

INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI

A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
313/761-4700 800/521-0600

**A STUDY OF FACTORS INFLUENCING ADOPTION OF A FIRST
PROGRAMMING LANGUAGE IN INTRODUCTORY COMPUTER SCIENCE
COURSES IN NORTH CAROLINA FOUR-YEAR COLLEGES AND
UNIVERSITIES**

A Dissertation Presented

by

LALCHAND T. SHIMPI

**Submitted to the Graduate School of the
University of Massachusetts Amherst in partial fulfillment
of the requirements for the degree of**

DOCTOR OF EDUCATION

FEBRUARY 1995

SCHOOL OF EDUCATION

UMI Number: 9524748

Copyright 1995 by
SHIMPI, LALCHAND TUKARAM
All rights reserved.

UMI Microform 9524748
Copyright 1995, by UMI Company. All rights reserved.

This microform edition is protected against unauthorized
copying under Title 17, United States Code.

UMI

300 North Zeeb Road
Ann Arbor, MI 48103

© Lalchand Tukaram Shimpi 1995
All Rights Reserved

A STUDY OF FACTORS INFLUENCING ADOPTION OF A FIRST
PROGRAMMING LANGUAGE IN INTRODUCTORY COMPUTER SCIENCE
COURSES IN NORTH CAROLINA FOUR-YEAR COLLEGES AND
UNIVERSITIES

A Dissertation Presented

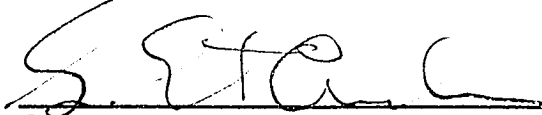
by

LALCHAND T. SHIMPI

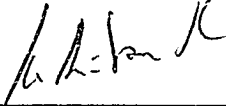
Approved as to style and content by:



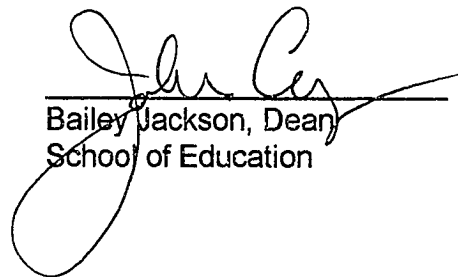
Howard A. Peelle, Chair



G. Ernest Anderson, Member



Krithi Ramamrithum, Member



Bailey Jackson, Dean
School of Education

ACKNOWLEDGMENTS

I am indebted to a number of people for assisting, encouraging, and inspiring me toward the completion of this project.

Dr. Howard A. Peelle was mainly the one who kept me going with his firm and gentle guidance from the time the idea for the dissertation was hatched to the time it was completed.

Dr. Ernest Anderson helped me with my data analysis.

Dr. Krithi Ramamrithum, with his background in Programming Languages, was an invaluable resource and support.

Dr. Ted Norton, with his ideas on Computer Science Education and Programming Languages, was very helpful in giving some critical but valuable information.

Furthermore, I wish to acknowledge the following:

Dr. Klaus Schultz for being a member on my comprehensive examination committee; Dr. Jim Kurose for being an outside member on my comprehensive examination committee; my colleagues and students from the twenty North Carolina four-year colleges and/or universities for being the research subjects; my friend and colleague from Keene State College, Ronald Tourgee, for pushing me on; my brother, Gopi Shimpi, for helping and encouraging me.

Finally, I want to thank my wife, Razieh, and my son, Anand, for giving me emotional support.

ABSTRACT

A STUDY OF FACTORS INFLUENCING ADOPTION OF A FIRST PROGRAMMING LANGUAGE IN INTRODUCTORY COMPUTER SCIENCE COURSES IN NORTH CAROLINA FOUR-YEAR COLLEGES AND UNIVERSITIES

FEBRUARY, 1995

LALCHAND T. SHIMPI,

B.S., UNIVERSITY OF POONA

M.S., UNIVERSITY OF POONA

M.A., UNIVERSITY OF MASSACHUSETTS BOSTON

M.S., UNIVERSITY OF MASSACHUSETTS AMHERST

Ed.D., UNIVERSITY OF MASSACHUSETTS AMHERST

Directed by: Dr. Howard A. Peelle

This study focused on the selection of programming languages in Computer Programming I classes (CS1) in four-year colleges and universities from North Carolina. The objectives were to identify differences in faculty and student views about the programming languages used in the first computer programming class and to see if demographic variables as well as type of school, job market in the region, quality and amount of experience with programming languages and/or computers correlated with the selection of the language. The study also solicited judgements about important factors for choosing a particular programming language and reasons which seemed to have

influenced this selection. The study also determined how well the students and faculty in these first computer programming classes agreed on the selection of the languages and the factors which led to the selection.

Three instruments were used to accomplish the above objectives. One was a survey questionnaire sent to twenty four-year colleges and universities in North Carolina in May 1993. Second was a survey questionnaire administered to 322 students from Computer Programming I from these schools in North Carolina during Spring and Fall semesters of 1993. Third was an open-ended interview of 20 faculty.

Results of the student survey questionnaire showed that Pascal was the language respondents had the most experience with, and it was the most heavily used language among them, followed by BASIC, COBOL, and C/C++. The top three reasons for learning these languages were: job market demands, someone's advice, and popularity of the language. If the students were given a chance of learning a first programming language all over again, their number one choice would be Pascal followed by C/C++. The top three reasons for this selection were that the language was used in the other computer science courses, they wanted to learn the language, and it was an easy language to learn.

Results of the faculty survey questionnaire showed that Pascal was the most widely taught first and Second programming language, and C/C++ would be their number one choice for a new first programming language when and if

they were going to make another selection. Job market requirements, design and structure of a language that implements modularity, concurrency, reusable code, and competition from other area schools were the top reasons in the selection process of a first programming language. Examination of some variables as possible predictors of these first programming languages revealed the following:

Strong correlation between the selection of a first programming language and such factors as compiler cost, compiler availability, teaching staff knowledge, hardware availability, and cost of a language;

Strong correlation between the type of a school and such factors as ability of a language to form good programming habits, availability of the language, modularity, parameters, ease of design and structure of the language, and a language which provides job related skills, and is usable in the real world.

The follow-up interviews seemed to show that a significant number of faculty had been thinking about changing to a new first programming language. In other words, the Pascal era was going to end soon, and a replacement for Pascal was going to be either C or C++. It was also clear that most of the faculty were trying to follow the ACM guidelines whether or not they agreed with them.

TABLE OF CONTENTS

	Page
ACKNOWLEDGEMENTS.....	iv
ABSTRACT	v
LIST OF TABLES	xii
LIST OF FIGURES.....	xxxi
CHAPTER	
I. INTRODUCTION	1
Statement of the Problem	1
Rationale	2
Objectives of an Initial Programming Course	5
Reasons for Choosing a Primary Programming Language	7
Problems with Choosing a Primary Programming Language	8
Advantages of Choosing a Primary Programming Language	11
Criteria for Choosing a Primary Programming Language	12
Which Primary Programming Language Should be Selected?	18
The Aim of a Primary Programming Language Course	20
Future of Programming Languages used in Introductory Computer Science classes	22
Which Language(s) to teach?	25
II. REVIEW OF LITERATURE	28
Programming Languages and Computer Science Curriculum	28
Comparison of Programming Languages in Introductory Computer Science Classes	33
Summary	38

III. METHODOLOGY	40
Purpose and Rationale	40
Research Questions	41
Design of the survey	43
Sampling and Administration of the Student and Faculty Survey	46
Administration of the Open-Ended Interviews	49
Methods of Analysis	51
The Student Survey	51
The Faculty Survey	52
The Open-Ended Faculty Interviews	52
Limitations of Study	53
IV. RESULTS, ANALYSIS AND DISCUSSION	54
Analysis of the Student Survey	54
Group I - Students Taking the First Programming Language Class	55
Group II - Experience with the Programming Languages	62
Group III - First and Second Programming Languages Learned with the Reasons	72
Cross Tabulations by Demographic Variables	86
School Type and Student's Choice of a First Programming Language	87
School Type and the Reasons for Selecting a First Programming Language	89
School Type and Student's Choice of a New First Programming Language	104
School Type and the Reasons for Selecting a New First Programming Language If Students were Given Another Chance for Selecting a First Programming Language	106
School Type and Student's Choice of a Second Programming Language	122
School Type and the Reasons for Selecting a Second Programming Language	125

Analysis of the Faculty Survey	144
Group I - Faculty Teaching the First Programming Language Class	147
Group II - Main Factors and Procedures Behind the Selection of the First Programming Language	153
Group III - Course Content of the First Programming Language Class	163
Group IV - Future of First Programming Languages	167
Cross Tabulations by Demographic Variables	174
Type of School and First Programming Language	175
Type of School and Second Programming Language	178
Number of Years of Service within the Department and The Selection of First and Second Programming Languages	181
Type of School and Programming Languages Used in Different Classes in the Curriculum	184
Type of School and Reasons for Selecting a First Programming Language	187
Type of School and Factors for Selecting a First Programming Language	197
Type of School by the Time Line for Evaluating a Choice of a New First Programming Language ...	229
School Type and Different Purposes for Using a First Programming Language	237
School Type by the New First Programming Language ..	252
Type of School and Student Involvement in the Decision of Selecting a First Programming Language	255
Type of School and Different Ways Students Will be Involved in the Decision of the Selection of a First Programming Language	258
Type of School and the Opinion That the Method of Selecting Programming Language is Effective	261
Analysis of the Interviews	264
Introduction	264
Introductory Computer Science Courses and Programming Language(s) Used	266
Introductory Computer Science Course and Its Content .	268

Future of Introductory Computer Science Courses	272
Factors Influencing the Selection of a New First Language	277
Selection Process of a First Programming Language	280
 V. CONCLUSIONS AND RECOMMENDATIONS	285
Summary of Findings	285
General Recommendations	291
Recommendations for Four-Year Colleges and Universities	292
Suggestions for Further Research	294
 APPENDICES	
A. LIST OF ALL 4-YEAR COLLEGES AND UNIVERSITIES IN NORTH CAROLINA	297
B. STUDENT SURVEY QUESTIONNAIRE	298
C. FACULTY SURVEY QUESTIONNAIRE	302
D. CONSENT FORM FOR FACULTY	307
E. CONSENT FORM FOR STUDENTS	309
F. INTERVIEW GUIDELINES AND QUESTIONS	310
G. SAMPLE INTERVIEW TRANSCRIPTS	312
 BIBLIOGRAPHY	331

LIST OF TABLES

	Page
1. V1 Classification of Students by Their Class	58
2. V2 Classification of Students by Age	58
3. V3 Classification of Students by Sex	59
4. V4 Classification of Students by the Number of Courses Taken ...	59
5. V5 Classification of Students by Majors	61
6. V6 Classification of Students by Experience with Programming Languages	69
7. V7 Classification of Students by the Most Used Programming Language	70
8. V8 Classification of Students by the First Programming Language Learned	77
9. V9 Classification of Students by the Number of Years Since They Learned Their First Programming Language	78
10. V10 Classification of Students by the Reasons for Learning the First Programming Language	79
11. V11 Classification of Students by the Second Programming Language Learned	80
12. V12 Classification of The Students by The Number of Years Since They Learned Their Second Programming Language	82
13. V13 Classification of Students by the Reasons for Learning the Second Programming Language	83
14. V14 Classification of Students by Their Choice of a First Programming Language, if They Were Given Another Chance to Learn a Language	84

15.	V15 Classification of Students by the Reasons for Learning this Language as the First Programming Language	85
16.	Cross Tabulation of School Type And First Programming Language Learned	88
17.	Cross Tabulation of School Type And First Programming Language Learned	88
18.	Cross Tabulation of School Type And First Programming Language Learned	88
19.	Cross Tabulation of School Type And the Language Was Used in the First Programming Language Class as a Reason in Learning the First Programming Language	91
20.	Cross Tabulation of School Type And Wanted to Learn the Language as a Reason in Learning the First Programming Language	91
21.	Cross Tabulation of School Type And Someone Told to Learn the Language as a Reason in Learning the First Programming Language	91
22.	Cross Tabulation of School Type And Only Language Available as a Reason in Learning the First Programming Language	92
23.	Cross Tabulation of School Type And the Language Was Popular as a Reason in Learning the First Programming Language	92
24.	Cross Tabulation of School Type And Job Market for the Language as a Reason in Learning the First Programming Language	92
25.	Cross Tabulation of School Type And Only Language Was Offered as a Reason in Learning the First Programming Language	93
26.	Cross Tabulation of School Type And the Language Was Required for the Major as a Reason in Learning the First Programming Language	93

27.	Cross Tabulation of School Type And it Was an Easy Language as a Reason in Learning the First Programming Language	93
28.	Cross Tabulation of School Type And the Language Was Used in the First Programming Language Class as a Reason in Learning the First Programming Language	96
29.	Cross Tabulation of School Type And Wanted to Learn The Language as a Reason in Learning The First Programming Language	96
30.	Cross Tabulation of School Type And Someone Told to Learn The Language as a Reason in Learning The First Programming Language	96
31.	Cross Tabulation of School Type And Only Language Available as a Reason in Learning The First Programming Language	97
32.	Cross Tabulation of School Type And The Language Was Popular as a Reason in Learning The First Programming Language	97
33.	Cross Tabulation of School Type And Job Market For The Language as a Reason in Learning The First Programming Language	97
34.	Cross Tabulation of School Type And Only Language Was Offered by The School as a Reason in Learning The First Programming Language	98
35.	Cross Tabulation of School Type And the Language Was Required For the Major as a Reason in Learning The First Programming Language	98
36.	Cross Tabulation of School Type And it Was an Easy Language as a Reason in Selecting The First Programming Language	98

37.	Cross Tabulation of School Type And the Language Was Used in the First Programming Language Class as a Reason in Learning the First programming language	101
38.	Cross Tabulation of School Type And Wanted to Learn The Language as a Reason in Learning The First Programming Language	101
39.	Cross Tabulation of School Type And Someone Told to Learn The Language as a Reason in Learning The First Programming Language	101
40.	Cross Tabulation of School Type And Only Language Available as a Reason in Learning The First Programming Language	102
41.	Cross Tabulation of School Type And The Language Was Popular as a Reason in Learning The First Programming Language	102
42.	Cross Tabulation of School Type And Job Market For The Language as a Reason in Learning The First Programming Language	102
43.	Cross Tabulation of School Type And Only Language Was Offered by The School as a Reason in Learning The First Programming Language	103
44.	Cross Tabulation of School Type And The Language Was Required For The Major as a Reason in Learning The First Programming Language	103
45.	Cross Tabulation of School Type And it Was an Easy Language as a Reason in Learning The First Programming Language	103
46.	Cross Tabulation of School Type And New First Programming Language Learned	105
47.	Cross Tabulation of School Type And New First Programming Language Learned	105

48.	Cross Tabulation of School Type And New First Programming Language Learned	105
49.	Cross Tabulation of School Type And The Language Was Used in The Second Programming Language Class as a Reason in Learning The New First Programming Language	108
50.	Cross Tabulation of School Type And Wanted to Learn The Language as a Reason in Learning The New First Programming Language	108
51.	Cross Tabulation of School Type And Someone Told to Learn The Language as a Reason in Learning The New First Programming Language	108
52.	Cross Tabulation of School Type And Only Language Available as a Reason in Learning The New First Programming Language	109
53.	Cross Tabulation of School Type And The Language Was Popular as a Reason in Learning The New First Programming Language	109
54.	Cross Tabulation of School Type And Job Market For The Language as a Reason in Learning The New First Programming Language	109
55.	Cross Tabulation of School Type And Only Language Was Offered by The School as a Reason in Learning The New First Programming Language	110
56.	Cross Tabulation of School Type And The Language Was Required For the Major as a Reason in Learning The New First Programming Language	110
57.	Cross Tabulation of School Type And it Was an Easy Language as a Reason in Selecting the New First Programming Language	110

58.	Cross Tabulation of School Type And The Language Was Used in the Second Programming Language Class as a Reason in Learning the New First Programming Language	114
59.	Cross Tabulation of School Type And Wanted to Learn the Language as a Reason in Learning the New First Programming Language	114
60.	Cross Tabulation of School Type And Someone Told to Learn The Language as a Reason in Learning the New First Programming Language	114
61.	Cross Tabulation of School Type And Only Language Available as a Reason in Learning The New First Programming Language	115
62.	Cross Tabulation of School Type And The Language Was Popular as a Reason in Learning The New First Programming Language	115
63.	Cross Tabulation of School Type And Job Market For The Language as a Reason in Learning The New First Programming Language	115
64.	Cross Tabulation of School Type And Only Language Was Offered by The School as a Reason in Learning The New First Programming Language	116
65.	Cross Tabulation of School Type And The Language Was Required For The Major Class as a Reason in Learning The New First Programming Language	116
66.	Cross Tabulation of School Type And it Was an Easy Language as a Reason in Selecting The New First Programming Language	116
67.	Cross Tabulation of School Type And The Language Was Used in The Second Programming Language Class as a Reason in Learning The New First Programming Language	119

68.	Cross Tabulation of School Type And Wanted to Learn The Language as a Reason in Learning The New First Programming Language	119
69.	Cross Tabulation of School Type And Someone Told to Learn The Language as a Reason in Learning The New First Programming Language	119
70.	Cross Tabulation of School Type And Only Language Available as a Reason in Learning The New First Programming Language	120
71.	Cross Tabulation of School Type And The Language Was Popular as a Reason in Learning The New First Programming Language	120
72.	Cross Tabulation of School Type And Job Market For The Language as a Reason in Learning The New First Programming Language	120
73.	Cross Tabulation of School Type And Only Language Was Offered by The School as a Reason in Learning The New First Programming Language	121
74.	Cross Tabulation of School Type And The Language Was Required For The Major as a Reason in Learning The New First Programming Language	121
75.	Cross Tabulation of School Type And it Was an Easy Language as a Reason in Selecting The New First Programming Language	121
76.	Cross Tabulation of School Type And Second Programming Language Learned	124
77.	Cross Tabulation of School Type And Second Programming Language Learned	124
78.	Cross Tabulation of School Type And Second Programming Language Learned	124

79.	Cross Tabulation of School Type And The Language Was Used in The Second Programming Language Class as a Reason in Learning The Second Programming Language	127
80.	Cross Tabulation of School Type And Wanted to Learn The Language as a Reason in Learning The Second Programming Language	127
81.	Cross Tabulation of School Type And Someone Told to Learn The Language as a Reason in Learning The Second Programming Language	127
82.	Cross Tabulation of School Type And Only Language Available as a Reason in Learning The Second Programming Language	128
83.	Cross Tabulation of School Type And The Language Was Popular as a Reason in Learning The Second Programming Language	128
84.	Cross Tabulation of School Type And Job Market For The Language as a Reason in Learning The Second Programming Language	128
85.	Cross Tabulation of School Type And Only Language Was Offered by The School as a Reason in Learning The Second Programming Language	129
86.	Cross Tabulation of School Type And The Language Was Required For The Major as a Reason in Learning The Second Programming Language	129
87.	Cross Tabulation of School Type And it Was an Easy Language as a Reason in Selecting The Second Programming Language	129
88.	Cross Tabulation of School Type And The Language Was Used in The Second Programming Language Class as a Reason in Learning The Second Programming Language	133

89.	Cross Tabulation of School Type And Wanted to Learn The Language as a Reason in Learning The Second Programming Language	133
90.	Cross Tabulation of School Type And Someone Told to Learn The Language as a Reason in Learning The Second Programming Language	133
91.	Cross Tabulation of School Type And Only Language Available as a Reason in Learning The Second Programming Language	134
92.	Cross Tabulation of School Type And The Language Was Popular as a Reason in Learning The Second Programming Language	134
93.	Cross Tabulation of School Type And Job Market For The Language as a Reason in Learning The Second Programming Language	134
94.	Cross Tabulation of School Type and Only Language Was Offered by the School as a Reason in Learning the Second Programming Language	137
95.	Cross Tabulation of School Type And The Language Was Required For The Major as a Reason in Learning The Second Programming Language	137
96.	Cross Tabulation of School Type And it Was an Easy Language as a Reason in Selecting The Second Programming Language	137
97.	Cross Tabulation of School Type And The Language Was Used in The Second Programming Language Class as a Reason in Learning The Second Programming Language	141
98.	Cross Tabulation of School Type And Wanted to Learn The Language as a Reason in Learning The Second Programming Language	141

99.	Cross Tabulation of School Type And Someone Told to Learn The Language as a Reason in Learning The Second Programming Language	141
100.	Cross Tabulation of School Type And Only Language Available as a Reason in Learning The Second Programming Language	142
101.	Cross Tabulation of School Type And The Language Was Popular as a Reason in Learning The Second Programming Language	142
102.	Cross Tabulation of School Type And Job Market For The Language as a Reason in Learning The Second Programming Language	142
103.	Cross Tabulation of School Type And Only Language Was Offered by The School as a Reason in Learning The Second Programming Language	143
104.	Cross Tabulation of School Type And The Language Was Required For The Major as a Reason in Learning The Second Programming Language	143
105.	Cross Tabulation of School Type And it Was an Easy Language as a Reason in Selecting The Second Programming Language	143
106.	Classification of Schools by Status of Minority/non-minority	146
107.	Classification of Schools by Status of The Type of School (Private/public)	146
108.	Classification of Schools by The Type of School (4-year College/university)	146
109.	Classification of Schools by Their Participation in The Survey	146
110.	V1 Classification of Faculty by the Departments in Which They Work	149

111.	V2 Classification of Faculty by the Number of Years of Affiliation with the Department	150
112.	V3 Classification of Faculty by The First Programming Language They Teach	151
113.	V4 Classification of Faculty by The Second Programming Language They Teach	152
114.	V5 Classification of First Programming Language And Other Computer Science Courses	155
115.	V6 Classification of Faculty by The Decision Process For Selecting The First Programming Language	155
116.	V7 Classification of Faculty by The Factors Playing Major Role in The Selection of The First Programming Language	157
117.	V8 Classification of Faculty by How Often Choice of a First Programming Language is Re-evaluated	157
118.	V10 Classification of Faculty by The Effectiveness of the Department in Selecting a First Programming Language	160
119.	V13 Classification of Faculty by The Most Important Factors For Choosing The First Programming Language	160
120.	V9 Classification of Faculty by The Purposes of Using This Language	162
121.	V11 Classification of Faculty by The Percentage of Course Content In Teaching The First Programming Language Class	165
122.	V13 Classification of Faculty by The Percentage of Time Spent In Each Delivery Method When Teaching The First Programming Language Class	166
123.	V14 Classification of Faculty by The Decision of Changing to Another First Programming Language	169

124.	V15 Classification of Faculty by The New First Programming Language	169
125.	V16 Classification of Faculty by The Reasons For Selecting This New First Programming Language	171
126.	V17 Classification of Faculty by The Time Deadline For Changing to a New First Programming Language	171
127.	V18 Classification of Faculty by The Fact Whether Students Will be Involved in The Decision of Selecting The New First Programming Language	173
128.	V19 Classification of Faculty by The Ways Students Will Be Involved in The Selection Process of Choosing a New First Programming Language	173
129.	Cross Tabulation of School Type by First Programming Language (Pascal or Other)	177
130.	Cross Tabulation of School Type by First Programming Language (Pascal or Other)	177
131.	Cross Tabulation of School Type by First Programming Language (Pascal or Other)	177
132.	Cross Tabulation of School Code by Second Programming Language (Pascal or Other)	180
133.	Cross Tabulation of School Type by Second Programming Language (Pascal or Other)	180
134.	Cross Tabulation of School Status by Second Programming Language (Pascal or Other)	180
135.	Cross Tabulation of Years of Service by First Programming Language (Pascal or Other)	183
136.	Cross Tabulation of Years of Service by Second Programming Language (Pascal or Other)	183
137.	Cross Tabulation of School Type by Programming Languages Used In Different Classes (Pascal or other)	186

138.	Cross Tabulation of School Status by Programming Languages Used In Different Classes (Pascal or Other)	186
139.	Cross Tabulation of School Code by Programming Languages Used In Different Classes (Pascal or Other)	186
140.	Cross Tabulation of School Type by Job Demands as a Reason for Selecting First Programming Language	188
141.	Cross Tabulation of School Type by Job Demands as a Reason for Selecting First Programming Language	188
142.	Cross Tabulation of School Type by Job Demands as a Reason for Selecting First Programming Language	188
143.	Cross Tabulation of School Type by Design of a Language and Ease of its Use as a Reason for Selecting First Programming Language	190
144.	Cross Tabulation of School Type by Design of a Language and Ease of its Use as a Reason for Selecting First Programming Language	190
145.	Cross Tabulation of School Type by Design of a Language and Ease of its Use as a Reason for Selecting First Programming Language	190
146.	Cross Tabulation of School Type by Software Limitations as a Reason for Selecting First Language	192
147.	Cross Tabulation of School Type by Software Limitations as a Reason for Selecting First Language	192
148.	Cross Tabulation of School Type by Software Limitations as a Reason for Selecting First Language	192
149.	Cross Tabulation of School Type by Opinion of Business Advisory Council of Teachers and Business Members as a Reason for Selecting First Programming Language	194

150.	Cross Tabulation of School Type by Structural Language That Implements Modularity, Concurrency, Reusable Code as a Reason for Selecting First Programming Language	196
151.	Cross Tabulation of School Type by Structural Language That Implements Modularity, Concurrency, Reusable Code as a Reason for Selecting First Programming Language	196
152.	Cross Tabulation of School Type by Job Demands as a Factor For Selecting First Programming Language	199
153.	Cross Tabulation of School Type by Job Demands as a Factor for Selecting First Programming Language	199
154.	Cross Tabulation of School Type by Job Demands as a Factor for Selecting First Programming Language	199
155.	Cross Tabulation of School Type by Potential For Data Structures as a Factor For Selecting First Programming Language	202
156.	Cross Tabulation of School Type by Potential For Data Structures as a Factor For Selecting First Programming Language	202
157.	Cross Tabulation of School Type by Potential For Data Structures as a Factor For Selecting First Programming Language	202
158.	Cross Tabulation of School Type by The Ability to Create True Abstract Data Types as a Factor For Selecting First Programming Language	205
159.	Cross Tabulation of School Type by Ability to Create True Abstract Data Types as a Factor For Selecting First Programming Language	205
160.	Cross Tabulation of School Type by the Ability to Create True Abstract Data Types (ADT) as a Factor For Selecting First Programming Language	205

161.	Cross Tabulation of School Type by the Ability of a Language to Form Good Programming Habits as a Factor for Selecting First Programming Language	208
162.	Cross Tabulation of School Type by the Ability of a Language to Form Good Programming Habits as a Factor for Selecting First Programming Language	208
163.	Cross Tabulation of School Type by the Ability of a Language to Form Good Programming Habits as a Factor for Selecting First Programming Language	208
164.	Cross Tabulation of School Type by the Compiler Cost as a Factor for Selecting First Programming Language	210
165.	Cross Tabulation of School Type by the Compiler Cost as a Factor for Selecting First Programming Language	210
166.	Cross Tabulation of School Type by Compiler Availability as a Factor for Selecting First Language	213
167.	Cross Tabulation of School Type by Compiler Availability as a Factor for Selecting First Language	213
168.	Cross Tabulation of School Type by Compiler Availability as a Factor for Selecting First Language	213
169.	Cross Tabulation of School Type by Teaching Staff Knowledge as a Factor for Selecting First Programming Language	216
170.	Cross Tabulation of School Type by Teaching Staff Knowledge as a Factor for Selecting First Programming Language	216
171.	Cross Tabulation of School Type by Teaching Staff Knowledge as a Factor for Selecting First Programming Language	216
172.	Cross Tabulation of School Type by Availability of Texts as a Factor for Selecting First Language	219
173.	Cross Tabulation of School Type by the Availability of Texts as a Factor for Selecting First Language	219

174.	Cross Tabulation of School Type by the Availability of Texts as a Factor for Selecting First Language	219
175.	Cross Tabulation of School Type by Hardware Availability as a Factor to Select a New First Programming Language	222
176.	Cross Tabulation of School Type by Hardware Availability as a Factor to Select a New First Programming Language	222
177.	Cross Tabulation of School Type by Hardware Availability as a Factor to Select a New First Programming Language	222
178.	Cross Tabulation of School Type by Cost of a Language as a Factor to Select a New First Language	225
179.	Cross Tabulation of School Type by Cost of a Language as a Factor to Select a New First Language	225
180.	Cross Tabulation of School Type by Cost of a Language as a Factor to Select a New First Language	225
181.	Cross Tabulation of School Type by Language Features as a Factor to Select a New First Language	228
182.	Cross Tabulation of School Type by Language Features as a Factor to Select a New First Language	228
183.	Cross Tabulation of School Type by Language Features as a Factor to Select a New First Language	228
184.	Cross Tabulation of School Type by Time Line of Every Year to Select a New First Programming Language	231
185.	Cross Tabulation of School Type by Time Line of Every Year to Select a New First Programming Language	231
186.	Cross Tabulation of School Type by Time Line of Every Year to Select a New First Programming Language	231

187	Cross Tabulation of School Type by 2-4 Year Time Period to Select a New First Programming Language (Pascal or Other)	234
188.	Cross Tabulation of School Type by 2-4 Year Time Period to Select a New First Programming Language (Pascal or Other)	234
189.	Cross Tabulation of School Type by 2-4 Year Time Period to Select a New First Programming Language (Pascal or Other)	234
190.	Cross Tabulation of School Type by as Need Arises to Select a New First Programming Language	236
191.	Cross Tabulation of School Type by as Need Arises to Select a New First Programming Language	236
192.	Cross Tabulation of School Type by as Need Arises to Select a New First programming Language	236
193.	Cross Tabulation of School Type by Language Availability as a Purpose to Select a New First Programming Language	239
194.	Cross Tabulation of School Type by Language Availability as a Purpose to Select a New First Programming Language	239
195.	Cross Tabulation of School Type by Language Availability as a Purpose to Select a New First Programming Language	239
196.	Cross Tabulation of School Type by Parameters and Modularity as a Purpose to Select a New First Language	242
197.	Cross Tabulation of School Type by Parameters and Modularity as a Purpose to Select a New First Language	242
198.	Cross Tabulation of School Type by Parameters and Modularity as a Purpose to Select a New First Language	242

199.	Cross Tabulation of School Type by Giving Students Better Understanding of Gui and Other Features of Work Stations as a Purpose to Select a New First Programming Language	245
200.	Cross Tabulation of School Type by Giving Students Better Understanding of Gui and Other Features of Work Stations as a Purpose for Using a New First Programming Language	245
201.	Cross Tabulation of School Type by Giving Students Better Understanding of Gui and Other Features of Work Stations as a Purpose for Using a New First Programming Language	245
202.	Cross Tabulation of School Type by Ease of Design, Structure as a Purpose to Select a New First Language	248
203.	Cross Tabulation of School Type by Ease of Design, Structure as a Purpose to Select a New First Language	248
204.	Cross Tabulation of School Type by Ease of Design, Structure as a Purpose to Select a New First Language	248
205.	Cross Tabulation of School Type by to Provide Job-related Skills, Usable in Real-world and Deals with Societal Issues as a Purpose for Using First Programming Language	251
206.	Cross Tabulation of School Type by to Provide Job-related Skills, Usable in Real-world and Deals with Societal Issues as a Purpose for Using First Programming Language	251
207.	Cross Tabulation of School Type by to Provide Job-related Skills, Usable in Real-world and Deals with Societal Issues as a Purpose for Using First Programming Language	251
208.	Cross Tabulation of School Type by Another Language to Select as a New First Programming Language	254

209.	Cross Tabulation of School Type by New First Programming Language (Pascal or C/C++)	254
210.	Cross Tabulation of School Type by New First Programming Language (Pascal or C/C++)	254
211.	Cross Tabulation of School Type by Whether Students Will be Involved in the Decision of Selecting First Language	257
212.	Cross Tabulation of School Type by Whether Students Will be Involved in the Decision of Selecting First Language	257
213.	Cross Tabulation of School Code by Whether Students Will be Involved in the Decision of Selecting First Language	257
214.	Cross Tabulation of School Status by Different Ways Students Will Be Involved in the Decision of Selecting First Programming Language	260
215.	Cross Tabulation of School Type by Different Ways Students Will Be Involved in the Decision of Selecting First Programming Language	260
216.	Cross Tabulation of School Status by the Opinion That the Method of Selecting First Programming Language Is Effective	263
217.	Cross Tabulation of School Status by the Opinion That the Method of Selecting First Programming Language Is Effective	263
218.	Cross Tabulation of School Status by the Opinion That the Method of Selecting First Programming Language Is Effective	263

LIST OF FIGURES

	Page
1. Classification of Students by Their Class	60
2. Classification of Students by Their Age Group	60
3. Classification of Students by Sex	60
4. Classification of Students by Number of Computer Courses Taken	60
5. Classification of Students by Their Major (If Declared)	64
6. Classification of Students by Their Experience With ADA	64
7. Classification of Students by Their Experience With BASIC	64
8. Classification of Students by Their Experience With C++	64
9. Classification of Students by Their Experience With C	65
10. Classification of Students by Their Experience With FORTH	65
11. Classification of Students by Their Experience With FORTRAN	65
12. Classification of Students by Their Experience With HYPERTALK	65
13. Classification of Students by Their Experience With LISP	66
14. Classification of Students by Their Experience With LOGO	66

15.	Classification of Students by Their Experience With ASSEMBLY	66
16.	Classification of Students by Their Experience With MODULA-2	66
17.	Classification of Students by Their Experience With Other Lang.	67
18.	Classification of Students by Their Experience With PASCAL	67
19.	Classification of Students by Their Experience With PL/I	67
20.	Classification of Students by Their Experience With PROLOG	67
21.	Classification of Students by Their Experience With SCHEME	68
22.	Classification of Students by Their Experience With SMALLTALK	68
23.	Classification of Students by Their Experience With TURING	68
24.	Classification of Students by Their Experience With COBOL	68
25.	Frequency Count of The Students By the Mostly Used Programming Language	71
26.	Frequency Count of The Students by the 1st Programming Language Learned	71
27.	Frequency Count of The Students by the No. of Years Since They Learned 1st Language	71
28.	Frequency Count of The Students by Reasons For Learning the 1st Prog. Language	71
29.	Frequency Count of The Students by Second Programming Language Learned	76

30.	Classification of Students by No. of Years Since They Learned 2nd Language	76
31.	Frequency Count of The Students by Reasons For Learning Second Language	76
32.	Frequency Count of The Students by the Choiced First Language	76
33.	Classi. of Students by Reasons For Learning Choiced First Language	81
34.	Cross Tabulation of School Type and First Programming Language Used	81
35.	Cross Tabulation of School Type by First Programming Language Used	81
36.	Cross Tabulation of School Type by First Programming Language Used	81
37.	Cross Tabulation of School Type and Second Programming Language Used	113
38.	Cross Tabulation of School Type and Second Programming Language Used	113
39.	Cross Tabulation of School Type and Second Programming Language Used	113
40.	Classification of Faculty by the First Programming Language	113
41.	Classification of Schools by Willingness to Participate	135
42.	Classification of Schools by the Minority/Non-Minority Status	135
43.	Classification of Schools by Their Type (Public/Private)	135
44.	Classification of Schools by Their Type (4-YR College/University)	135

45.	Classification of Faculty by the Second Programming Language	136
46.	Classification of Faculty by Decision Process For Selection	136
47.	Classification of Faculty by Major Factors in Selection	136
48.	Classification of Faculty by Reevaluation Schemes For Selection	136
49.	Classification of Faculty by Effectiveness of Selection	140
50.	Classification of Faculty by Purposes of Using First Language	140
51.	Classification of Faculty by Factors in the Selection Process	140
52.	Classification of Faculty by Time Line For Changing to New Language	140
53.	Classification of Faculty by Percentage of Course Content	159
54.	Classification of Faculty by the Decision to Change	159
55.	Classification of Faculty by the New First Programming Language	159
56.	Classification of Faculty by Reasons For New First Language	159
57.	Classification of Faculty by Allowing Students In the Decision	168
58.	Classification of Faculty by Different Ways of Involving Students	168
59.	Classification of Faculty by the Departments They Work For	168
60.	Classification of Faculty by the Number of Years of Affiliation	168

CHAPTER I

INTRODUCTION

Statement of the Problem

Decisions regarding selection of a first programming language for introductory computer science courses typically are not policy-driven, are not widely disseminated, exclude student input, and do not get input from other departments on campus. Further, it is not known to what extent faculty and students agree on the programming language used in introductory computer science courses.

Ever since the second high-level programming language course was introduced, the Computer Science educator has had to decide which language to use for instructional purposes. As high-level languages proliferated, the decision became more difficult (Evans, 1984; Tatar, 1986; Sparks, 1988; Reisman, 1982; Shirkhande, 1986; Morris, 1985; Maddux, 1984; Krus, 1987; Taylor, 1987). Although Curriculum 78 suggests features which should be available in a programming language, it does not recommend one language or address in detail the process of selecting one (Frank, 1990; Wileman, 1981; Wexeblat, 1981; Schneider, 1989; Ferchichi, 1987).

Some language developers and proponents claim that a particular language is suitable for all situations, but it does not appear that a consensus has been reached (Hill, 1980; Prather, 1983; Marcel, 1986; Lee, 1989; Leeper, 1984; Sointseff, 1984; Oman, 1986).

It appears that four-year colleges in the United States use several different programming languages in introductory computer science classes. Two publications, Computing Curricula (1991) and the Denning Report (1990), ignited a debate over the best approach to the introductory computer science course, dividing computer scientists into two camps: traditionalists who believe that structured programming, object oriented programming languages, and functional programming languages, problem solving, procedural abstraction, and data abstraction are the right knowledge to launch students on their careers; and "new wavers" who believe that a breadth of Computer Science approach - that students should sample a "dim sum" of topics - is appropriate.

Rationale

A study of factors which influence adoption of a first programming language in introductory computer science courses is needed so other Computer Science faculty can know results and can make better decisions based on actual research. Computer Science education would have a common ground if the faculty and students agree on a programming language to be used in introductory computer programming classes. If students and faculty do not agree on the choice of a programming language, at least the study of the reasons for choosing a particular programming language(s) between the students and faculty would help Computer Science education in designing curriculum. If male and female students have

different reasons for choosing a programming language used in introductory Computer Science courses, the study should help the faculty adapt their courses to the majority of males and/or females.

There is increasing discussion about the primary language used for undergraduate courses in Computer Science education. The language used for the first and the second programming course recommended by ACM is regarded as a crucial factor in students' subsequent progress in the discipline. The first course in Computer Science education has been the center of discussion for many years, as many students and educators have become dissatisfied with conventional teaching methods (Bauer, 1979; Bell, 1987; Blaisdell, 1985; Coombs, 1982; Dupras, 1984; Ellison, 1986; Gries, 1974; Hanson, 1975; Woodhouse, 1983; Winslow, 1989; Skublics, 1991; Motil, 1991; LaLonde, 1990; Koffman, 1988).

The selection of languages for use as pedagogical aids in the teaching of Computer Science is still a big issue at most colleges and universities. In any university or college environment, one is faced with demands for offering a variety of programming languages. The rebellion against FORTRAN has given rise to numerous heirs to the throne (e.g., PL/1, ALGOL, Pascal, C etc.) In deciding how to resolve this issue, departments have to consider the effect the decision might have on the productivity of their faculty. The literature (Chapter 2) reveals the existence of areas in which little agreement can be seen, the most important one being the programming language to be used in instruction.

The choice of a programming language to be used in a Computer Science curriculum is one of the most important decisions that Computer Science departments must make.

After agreeing on the importance of programming, the next question which comes to mind is the language to be learned first. As Blank (1981) noted a story from Bible: "Once upon a time the whole earth had one language and few words... Then (men) said, 'Come, let us build ourselves a city, and a tower with its top in the heavens... And the Lord came down to see the city and the tower, which the sons of men had built. And the Lord said, 'They are one people, and they have all one language; and this is only the beginning of what they will do; and nothing that they propose to do will now be impossible for them. Come, let us go down and confuse their language, that they may not understand one another's speech.' So the Lord scattered them abroad from there over the face of all the earth, and they left off building the city. Therefore its name was called Babel (meaning confusion), because the Lord confused the language of all the earth." (Genesis: Chapter 11)

According to this Biblical story, mankind was given different languages to prevent us from joining together and developing enough power to rival God. Perhaps we have developed so many computer languages to keep the computers from joining together and developing enough power to rival mankind? Every language has its own design objectives, which usually means that the particular language is ideal for some purposes and inappropriate for others. This proliferation is a further complication associated with the first Computer Science course.

The core of any Computer Science course is a thorough grounding in programming which must entail knowledge of at least one particular programming language.

To know a programming language is to have the tools to carry out many different tasks.

Objectives of an Initial Programming Course

The essential objectives of an initial programming course can be described as follows:

1. Direct initial discussion at teaching what constitutes a well-defined problem statement.
2. Concentrate on the introduction of the concept of an algorithm for solving a problem before discussing an actual coding of the problem using any particular programming language.
3. Introduce the relationship of data structures and algorithms in the process of choosing a 'right' data structure.
4. Choose a programming language that enhances the learning process. The choice of a general-purpose programming language is usually made on the basis of pragmatic, non-educational factors -- what is available and what is supported at one's installation, etc. This is an unfortunate fact of life that has serious educational repercussions, as stated by Edsger Dijkstra in the 1972 ACM Turing Award Lecture: "When I start to analyze the thinking habits of myself and my fellow

human beings I come, whether I like it or not, to a different conclusion, viz., that the tools we are trying to use and the language or notation we are using to express our thoughts are the major factors in determining what we think or express at all!... The analysis of the influence that programming languages have on the thinking habits of their users now gives us a yardstick for comparing the relative merits of various languages." Selection of a programming language should be based on which language meets two critical and apparently opposite criteria: richness and simplicity. The language should be rich in those constructs needed for introducing fundamental concepts in programming; and the language should be simple enough to be presented and grasped in a one semester course.

5. Concentrate on semantics and program characteristics -- not syntax.

6. Consider programming style as early as possible. The worst mistake an instructor can make is to initially teach programming quickly with the idea of coming back later and teaching it well. Bad habits die hard, so it is important to prevent them from ever developing.

7. Present the topic of debugging formally. Students should be presented with debugging techniques in the first programming course. This should help those who are just learning their beginning programming.

8. Teach program testing and verification. An important part of programming is to make sure that the programs run correctly. One must be taught the idea of testing a program and verifying it with some sample input before the actual run is done.

9. Include documentation. All programs written by any programmer should be documented properly, which in turn will help those who will use that program. Also, this being the first course, whatever habits the student picks up from this course will carry on to the next programming courses. So, it is very important that they are taught to provide documentation.

10. Provide an overall perspective of realistic programming and program environments. One has to be told about the limitations of any programming language and the environments under which each program will work.

Reasons for Choosing a Primary Programming Language

In 1974, Donald Knuth wrote: "At the present time I think we are on the verge of discovering at least what programming languages should really be like. I look forward to seeing many responsible experiments with language design during the next few years; and my dream is that by 1984 we will see a consensus developing for a really good programming language (or, more likely, a coherent family of languages). Furthermore, I'm guessing that people will become so disenchanted with the languages they are now using -- even COBOL & FORTRAN -- that this new language, UTOPIA 84, will have a chance to take over. At present we are far from that goal, yet there are indications that such a language is very slowly taking shape. (Knuth 1974)." Now, over a decade later, the number of languages is getting larger and larger. This multiplicity of languages poses a

problem for departments of Computer Science and for universities more generally.

Which languages should be taught to Computer Science students? To

Non-Computer Science students?

Students' first contact with programming is of prime importance and ought to be controlled carefully. This must be reflected in the choice of instructor, the choice of textbook, the choice of methodology, the choice of programming language, etc. With the recent advances of microcomputers, access to computing facilities is more and more common. This is causing a myriad of problems ranging from heterogeneity of students' backgrounds to ill-conceived first contacts with programming (Dijkstra 1982).

Problems with Choosing a Primary Programming Language

To answer the questions discussed above, we obviously have to know what languages are available and to evaluate them. But even here we encounter some problems. Teachers of FORTRAN typically know little about LISP; teachers of LISP may be strangers to PROLOG; and number of university people knowing anything about FORTH or SMALLTALK or APL can be counted on a very small number of fingers and toes. Do you choose to teach a language which is theoretically interesting, or one that will find extensive business and/or industrial use? Or, do you teach several different languages, hoping that the student will choose the best one?

Recognizing that the first programming course is no longer generating the desired results is not difficult. A problem arises, however, in determining a good replacement. In addition to the proper language to teach, developers of the first course are forced to consider:

1. The target population

In order to best satisfy the needs of the students in first programming language classes, we must make sure the backgrounds, majors, and needs of the students in the course. Students with a business background might be best served with a business language like COBOL as their first programming language as compared to the students with engineering and/or pure computer science majors will be happy with procedural or object oriented programming (OOP) languages.

2. The desired goals of the course

In order to attain the ten objectives of an initial programming language course, we have to specify the goals in the course. One may not be able to obtain all the ten goals. But depending upon the emphasis on the major goals, we will choose a particular language. If a major goal of a first programming course is to teach structured programming, one might be happy to go with a procedural language.

3. The course material

If the course material demands to cover all the basics plus a richness of data structures, one has to choose a language with all these features. For example, if the course material demands to cover recursion, one cannot be happy with FORTRAN as a language in the first programming language course.

4. The environment – class size, student background

If majority of the students in the first programming language course have a Pascal background as their programming language, one has to make sure that the language they are using in their first programming language course follows the same principles. It will be confusing for these students if the language they are learning is a non-structured language.

5. The availability of the faculty

The faculty availability has to be considered while making the choice of a first programming language. Since the faculty might have their own interests, and bias while teaching a programming language.

6. The availability of the compilers/interpreter

The availability of a compiler/interpreter might force some schools to choose a language in their first programming language course. For example, if one wants to teach ADA as their first programming language, it will be very difficult for a school with limited resources to offer ADA as their first programming language being the unavailability of full ADA compiler on a PC.

7. Trends in the industry and/or business world

The industry demands and business world will force schools to choose a particular language in their first programming language course whether the schools want or not. Also, competition from other schools will force them to fulfill industry and/or business world demands and choose a language otherwise they would not have chosen.

Various courses have been designed and implemented, based on the answers to the above questions. One solution is to establish several introductory courses – programming for engineers, programming for business students, programming for the social scientists, etc. Rather than using a single language in the first programming language course, a better choice will be to use different languages as first programming languages. This will satisfy the needs of all different backgrounds, majors, interests, and requirements for students learning the first programming language.

Advantages of Choosing a Primary Programming Language

Although a good Computer Science program will familiarize the student with a wide variety of programming languages, usually one language emerges as the de facto mode of expression for most Computer Science concepts. The use of a primary programming language throughout the curriculum also serves to give the entire program a cohesiveness that it might otherwise lack. Computer resources of the school can be optimized for the use of that language. Such a language facilitates the introduction of advanced Computer Science concepts. It reduces overlap among the various courses providing more student homogeneity. It is most cost-effective in terms of both computer and human resources.

Criteria for Choosing a Primary Programming Language

Many students are introduced to a first programming language in a college environment. The influence of that language on a student's subsequent thought processes and programming abilities (Wexebaltt, 1981) should be considered when selecting a language for introductory courses.

Several factors which influence the choice of the language to be used in supporting a Computer Science curriculum. What do we need to know in order to make curricular decisions about computer languages?

1. Overview of the modern languages available
2. Strengths and weaknesses
3. Some understanding of contemporary trends among existing computer languages
4. Projections about future trends in computing

Students are most likely to succeed in their first crucial programming experiences when programs containing a relatively few simple statement types can be composed and tested.

The use of FORTRAN in an introductory course is criticized for the effect it has on future programming. Heavy reliance on the "GOTO", for instance, is something students tend to stick with even when they are programming in a language where it is not necessary.

Factors which influence the choice of the language to be used in supporting a Computer Science curriculum:

A. Pedagogical factors

1. Availability of a subset for beginners

As a beginning programming language student, not only one wants to get a basic flavor of the language but also wants to get the feel of the language as a whole. To satisfy this need, the language should be such that by teaching a subset of the language a beginner not only receives the feel of the language but also gets interested in that language to learn it in full.

2. Support for upper level courses

When making a selection of a first programming language, one has to evaluate the requirements of the upper level courses and pay attention to the programming language(s) used in those courses. This might force one to go with a language which is used widely in the upper level courses.

3. Support for the process of teaching about programming

While teaching a first programming language course, one must remember that this is the initial contact of the students with a programming language, but most importantly, with the process of programming. Not only we are teaching a programming language, but, we are teaching the art of programming.

4. Text availability

The choice of a first programming language might also depend upon the wide availability of text books in the market. Since an enrollment in a first programming language might be the largest as compared to upper level computer science classes, one must take into consideration that there are enough number of texts available for the first programming language class.

B. Resource Constrains

5. Influence of a departmental computer and other computers

The availability of platforms of main-frame, mini-frame, or PCs might dictate the choice of a first programming language.

6. The time constraint of one three-year program

Some schools which are struggling with enrollments and fighting with budget constraints, might try to offer a program for a limited time on a trial basis. In those cases, one might choose a language on a trial basis and make sure that it is within the constraints of the resources for that school..

7. Cost efficiency and/or turnaround time

The choice of a first programming language might also depend upon the cost to use that language in this large enrollment course. Secondly, one must keep in mind that the decision of choosing a language is time-dependent and we might have to choose another first language within a time-limit.

C. Political Issues

8. Languages used in the "real world"

The language which is in great demand by the business world and is used heavily in industry might be a good selection as a first programming language as there will be a big demand for the graduates of that school.

9. Service courses

The demand of service courses might decide the selection of a first programming language at a particular school as one must maximize the use of available resources.

To make the selection of a primary language, one must take into account the above factors and also ask the following questions:

1. Is the language simple or complex?

A primary language should carefully balance power with simplicity. Each concept of programming should be easily explained using the features of the primary language and at the same time the language must be powerful enough to handle any complex programming structures.

2. Does the language promote good programming practice?

In a primary programming language course, we are cultivating the minds so that they have a good and strong basis for programming and they form good programming habits, we have to make sure that the first programming language must fulfill these requirements.

3. Is the language suitable for presenting advanced topics, such as pointers, recursion?

A primary programming language must be suitable for presenting advanced topics such as dynamic memory allocation, recursion.

4. Is the language suitable for a wide range of applications, such as scientific and business applications?

The target population in a primary programming language class might be students with different backgrounds, varied interests, and different majors. To satisfy the curiosity of this population, a primary programming language must be suitable for a wide range of applications, such as scientific and business applications.

5. Do well-defined, suitable subsets exist for the language?

The language's syntax and semantics must be free of ambiguity, and are complete. Also, a primary language must have suitable subsets so that one does not have to learn the whole language right away to start programming using the language.

6. Are compilers or interpreters widely available?

A primary programming language must have widely available compilers or interpreters. This will guarantee the use of the language outside the classroom by these students during and/or after they complete their first programming language course.

7. Is the language in widespread use?

A primary programming language must be in widespread use in the industry and by the business world. This will guaranty the big demand for the language and hence the people who use that language.

8. What is the cost of supporting use of the language?

When making a selection of a primary programming language, the cost of supporting the language will be an important factor. This cost will include software, hardware, and other necessary resources.

9. Does the language support the required core of upper level courses?

A primary programming language will be a good choice if it can support the required core of the upper level courses.

10. Does the language satisfy the criteria of generality?

The property that permits a language to handle a wide range of programming applications will be referred as the generality of the language.

11. Is the turnaround time short for the programs written in that language?

It has been my experience that good turnaround is important pedagogically and for maintaining high morale in beginning courses with large enrollments. Beginning students don't improve their debugging methods much when turnaround time increases because they don't have many debugging techniques that they understand how to use at an early stage of their development.

12. Is it necessary to teach the languages which are most widely used outside the classroom in order to keep the curriculum relevant to the real world?

Proponents of some of the newer languages claim that this should not be an issue and that by making it an issue Computer Science departments are dogmatically reinforcing a stagnant status quo.

Which Primary Programming Language Should be Selected?

The answer to the question, "Is this language a good primary language?", depends less upon the language than upon the application. The choice of a primary programming language will mainly depend upon the type of target population, the desired goals of the course, the trends in the business world and/or industry. The primary language syntax should be able to represent the algorithm at a very high level.

Various language designers have, from time to time, discussed their art itself. For instance, C. A. R. Hoare (1973) published his thoughts on the subject, and Niklaus Wirth (1974) did the same. In these and other similar papers, the overriding advice for language designers is to combine simplicity and functionality. The ultimate goal of a language should be to allow the programmer to think clearly about the complexity of the presented problem rather than the complexity of the programming language itself.

Programming languages have been and will continue to be an important instrument for the automation of a wide variety of functions within industry and for the Federal Government. Different programming languages tend to be developed for different application areas: principally "scientific," "data processing," "artificial intelligence," "text processing," and "systems programming" applications.

Given that conventional programming is the appropriate technique for a particular application, the choice among the various languages becomes an important issue. There are a great number of selection criteria, not all of which depend directly on the language itself.

The criteria are based on:

1. The language and its implementation;
2. The application to be programmed; and
3. The user's existing facilities and software.

When an application is to be implemented with conventional programming, the choice of language can have a major effect on its success. Moreover, the user must carefully consider the costs and benefits not only during development, but also throughout the life of the application. In many cases, maintenance costs exceed development costs.

Success of a language depends on its systematic approach to programming. Program code must be readable to facilitate maintenance and development. Strong typing leads to fewer run-time errors, at the relatively small expense of more compilation errors.

In the process of transformations from algorithm to implementation, the beginning student should encounter a minimum of coding concerns in order not to be distracted or diverted from the focus of the task: learning to design well engineered, well-structured solutions to data processing problems.

While the language may be rich, there must be an integral subset which, while limited in size and complexity, still provides sufficient power for the successful straight forward implementations of well-structured statements.

The language must be appropriate for the end-user programmer as well as for the potential professional. The overhead required for the solution of small problems should be correspondingly small. Additionally, the language should be widely available for quick, convenient, and casual use against small problems.

The program development cycle should be simple. The programming environment must be easy to master and remember. The language ought to be standardized -- independent of the operating systems and machine. The language ought to be interpreted, so that the novice programmer can deal with syntactic or typographic errors immediately.

The Aim of a Primary Programming Language Course

As long as the chosen programming language has certain essential features, then selection of any of a number of languages is satisfactory. The introductory programming course often must satisfy the need for computer literacy

as well as serve most advanced students who need to use the computer in their own discipline. Structured programming is almost a necessity for students with very limited background. Students in the first course have had little exposure to the sciences or mathematics. We want them to have a general appreciation of the skills required for work in Computer Science and some sense of what kind of problems are appropriate for computer solution. Since most of the students have little if any previous experience with the computer, it is essential that the first course have a significant amount of programming.

A primary goal of the first course is often the development of general problem-solving techniques. The use of structured programming assists in this task. The programming language chosen as a primary language for Computer Science majors should support problem abstraction and decomposition. Despite the attitude that a student's time is of little value, it is important to consider the time required for a student to develop a program when selecting a language.

Many students have had some previous experience with programming, hence preventing college and/or university faculty from controlling their first contact with programming. Their previous programming experience typically ranges anywhere from BASIC programming on a home computer to a formal Computer Science course in their pre-university education. In both cases this mean some amount of damage control and a lesser impact of our course design.

Often the choice of a programming language has been an emotional issue just as is the selection of a political candidate. Such subjectivity should be removed from decisions by providing an objective analysis of several languages.

Future of Programming Languages Used in Introductory Computer Science Classes

All these comparisons could not come up with one strong member as a primary language. In the light of current trends within the discipline, serious consideration must therefore be given to the adequacy of the department's "core" programming language, especially insofar as it meets the student's needs now and upon graduation, whether those needs focus on immediate employment or further formal education. Thus appropriate languages are those which, by their structures, are designed to support concepts currently viewed by the computing community as important, such as embedded systems, code encapsulation, abstract data types, symbol importing and exporting, separate compilation, concurrency, and object oriented programming.

Two famous forward-looking authors reach the same conclusion: The future belongs to those who can deal with reality in symbolic ways. Alvin Toffler (1991), shows that the new source of today's wealth is knowledge. But interestingly, access to knowledge alone is not enough. One also has to master the ability to analyze, evaluate and then organize these facts to achieve this new mantle power

in both the workplace and the marketplace. The most important economic development of our lifetime has been the rise of a new system for creating wealth, based no longer on muscle but on the power of mind.

The second author, Robert B. Reich (1991), addresses a different issue but reaches a remarkably similar conclusion. Reich states unequivocally that the future division of "haves" and "have-nots" will be determined by an individual's ability to work in symbolic terms.

Don't listen to those who tell you that "the era of programming is past. It should only be taught to those who decide to major in computer science." Or, "students only need to learn how to use these applications. Programming knowledge is no longer necessary."

Computer use has changed. In the early days, writing your own programs was often necessary because there were relatively few applications available. Today there are thousands of applications available, but very few programs work together easily. No single application can fill the needs and expectations of everyone. In the years ahead, companies and institutions will be searching for persons with the ability to take the results from one program and make it available to other programs. If our students master this life-changing skill, they will always be "better than average" and greatly in demand for their abilities!

It can hardly be emphasized too strongly that users should not ignore long-term costs and benefits. For small short-term projects, the total risk is low in any case. But for larger projects, many indirect criteria may become crucial. In

particular, it can be a decisive advantage when a language is supported by strong standardization. The casual user of the future will demand a language that is simple, powerful, and logical. Over the short term traditional procedural languages may continue to dominate in most personal computer applications. Experts expect that such languages will continue to evolve and that major new languages will be introduced occasionally.

Despite the enormous increase in the number of personal and small-business computers, it appears that development of programming languages will be much slower than in the past. This is because the rapid proliferation of PCs has drastically changed the environment in which programming languages are developed.

Demands will favor greater development of non-procedural fifth generation languages, such as PROLOG. The leaders of the industry will move toward the new languages, but individual organizations may resist. It seems unlikely that any single general purpose computing language is going to fill the needs of all the programmers.

Computer Science departments must devise curricula for students who wish to become professional computer scientists, and they must also create courses and opportunities for a broad spectrum of non-computer science students as well. Computer Science faculty should consider not only the languages they happened to grow up with, but the whole spectrum of languages now available.

Which Language(s) to Teach?

It is not yet and may never be clear which programming language will dominate our language culture of the future. One reason there is no such thing as The Best Computer Language is that there is no possible way to agree on what the best language should do.

We will need to go beyond the numbers game ("millions of people couldn't have been wrong about FORTRAN" or "everyone's teaching Pascal -- I guess we should too".) The choice of languages should depend upon what the future of languages seems to be, and upon the needs of students and what industry wants. There have been many attempts to construct a universal computer language, and anyone who has developed a good language might be tempted to wish that it could be universally available. However, it is doubtful that there will ever be such an ideal computer language.

Many students have been taught the technical aspects of a particular programming language (the syntax) with only minimal (if any) emphasis on the problem-solving process in particular, on the production of quality software, and on an introduction to Computer Science in general. We really should be asking more crucial questions:

- What we wish our beginning students to know.
- Why we want them to know it.
- How we wish them to acquire this knowledge.

This study seeks to identify the reasons for selecting a first programming language by students and faculty of fifty-eight four-year colleges and universities in North Carolina and hence by the department where the first programming language course is offered. This study will also seek to identify the major factors associated with the above selection and the process of selection of the first programming language. The study of these factors influencing the adoption of the first programming language(s) attempts to answer the following questions:

1. What are the main reasons faculty are using a language in the first programming language classes?
2. What are the major factors associated with a selection of first programming language(s)?
3. What are the main reasons students select a first programming language in their first programming language classes?
4. Are there significant differences in the reasons behind the language selection among different groups of faculty?
5. Are there significant differences in the reasons behind the selection of a first programming language among different groups of students who were taking the first programming language classes?
6. Do demographic factors such as type of school (four-year college vs university, private vs public, minority vs non-minority), and number of years of service with the department correlate with selection of first programming languages?

7. Are there critical differences in the reasons behind the selection of first programming languages among different groups of students and faculty?

In order to conduct this study, surveys were mailed to all faculty and students dealing with first programming language classes from fifty-eight four-year colleges and universities in North Carolina. From these, thirty three schools wanted to take part in the survey. But six did not offer programming language classes and hence did not participate in the study. Seven schools did not offer programming language classes during the Spring 1993 semester when the study was conducted and hence could not participate in the study. Twenty schools were selected because of their proximity to the researcher's work place and the sample represents a typical mix of four-year colleges and universities, public and private schools, and minority and non-minority schools in a large city in the United States with large student population.

The survey instruments were two sets of questionnaires: First was a 10-item questionnaire for students and, second was a 16-item questionnaire for faculty. Both sets of questionnaires were either mailed with a return postage-paid envelope or hand-delivered to the faculty and students in Spring 1993. A follow-up personal or telephone interview was conducted with the faculty of the survey population. The follow-up interview were done at the end of the Spring 1993 semester.

CHAPTER II

REVIEW OF LITERATURE

This review of the literature will focus on two areas. First, we will look at various studies expressing views about different programming languages used in Computer Science curriculum. Second, we will look at comparisons of various programming languages used in Computer Science curriculum.

Programming Languages and Computer Science Curriculum

After the first set of ACM (Association of Computing Machinery) guidelines in 1968, there was little attention paid by researchers to the choice of programming language(s) in Computer Science curriculum (Abbott, 1975; Lawrence, 1973; Cameron, 1975; Ruby, 1976; Nartker, 1977; Lopez, 1977; Austing, 1977). But, in 1978 ACM published a second set of guidelines for a Computer Science major. The first two courses were Computer Programming I (CS1) and Computer Programming II (CS2). After these guidelines appeared, serious discussion started (Beidler, 1985; Cohen, 1982; Fosberg, 1981; Goulet, 1982; Worland, 1978; Brookshear, 1985; Cunningham, 1978; Ellison, 1986; Gibbs, 1986; Mahoney, 1982; Powell, 1978; Wardle, 1982).

With the diversity of high-level programming languages available, selecting the "right" one for a Computer Science course can be a very difficult process (Smith, 1976; Cole, 1983; Furugori, 1977).

Expert opinions were expressed about each of the programming languages available at that time. Dijkstra (1972) claimed: " 'the infantile disorder', FORTRAN, by now nearly 20 years old, is hopelessly inadequate for whatever computer application you have in mind today: it is now too clumsy, too risky, and too expensive to use. It is practically impossible to teach good programming to students who had a prior exposure to BASIC: as potential programmers they are mentally mutilated beyond hope of regeneration."

Blaisdell & Burroughs (1985) claimed that "the use of COBOL cripples the mind; its teaching should, therefore, be regarded as criminal offense. It is practically impossible to teach good programming to students who have had prior exposure to BASIC. ADA is so large and contains so many constructs of limited use in a business environment that it would be a very constrained subset of ADA which would make a satisfactory introductory language."

James Martin recommended that "APL is the procedural language for end-users. Assembly Language seems inappropriate for business applications. COBOL is a large special-purpose language. It contains more syntax than BASIC, making it more difficult as an introductory language in a course where the primary objective ought to be the process of how one writes programs rather than the knowledge of the specific syntax.

COBOL requirements are so cumbersome as to distract the student from the focus of learning the principles and process of software engineering, and to mis-focus his attention on mere knowledge of the syntax. C and PL/I are both too sophisticated to be usable in a beginning business programming environment. BASIC is relatively small and simple. The syntax is straight forward and fairly mnemonic. BASIC is interpreted. Pascal is an excellent language with which to teach software engineering concepts."

Mallozzi (1985), commenting on the use of BASIC, said: "Students with only BASIC programming experience tend to over-emphasize syntax and have difficulty concentrating on larger programming issues such as top-down design and modular structure."

Sedlmeyer & Parman (1980) claimed: "Translation to code from algorithmic form is more difficult in BASIC because some control structures, such as variable declarations, are lacking in the language."

Most of the high schools were using BASIC as the language for their Computer Science classes (if they were offering). When these students came to college and were learning a structured language like Pascal in their first programming language classes, McGee, Wilson, and Polychronopoulos (1987) asked: "Does BASIC have a positive effect on performance in introductory Pascal courses?" The use of BASIC and FORTRAN in an introductory computer science course is criticized for the effect it has on future programming. Heavy reliance on the "GOTO", for instance, is something students tend to stick with even when they are programming in a language where it is not necessary.

Recently, the College Board has selected Pascal as the programming language required for the Advanced Placement Test in Computer Science. Braswell & Wadkins (1984) commented on the above decision by claiming "College Board chose Pascal over BASIC because Pascal encourages good programming habits."

Bauer (1979) recommended the use of Pascal in the first programming language class by making the claim that : "Pascal can be translated from algorithms much more directly." He felt that the procedural orientation and parameter passing in Pascal utilize the modular structure that was integral to the design of algorithms.

Fritz (1983) complained about teaching BASIC in the first programming language class by saying: "Many university instructors observe that an unstructured BASIC programming background causes problems for students in their introductory Computer Science courses." Bork (1983) supported the above claim by commenting on the use of BASIC as a programming language: "BASIC is 'junk food of computer programming', not nutritious enough for beginning programmers. BASIC is widely available as the built-in language of most PCS."

Of course BASIC did have its supporters like Braun (1983) who claimed that: "BASIC has simple syntax and interpretive structure." Blaisdell & Burroughs (1985); Wainwright (1980) also recommended BASIC over Pascal as a first programming language because of "BASIC's interpretive structure allows immediate debugging of syntax and typographical errors. Pascal requires waiting for a program to compile."

Merritt (1980) contested that "Presence or absence of certain features in programming languages can affect the quality of programs produced. It is my contention that teaching our students to program COBOL using "PERFORMs" rather than "GOTOs" is not enough. Pascal is small, well designed, available language. Pascal is strongly typed language; it is more strongly typed than any of its ancestors. Pascal does fall short of complete type safety with its variant record structure." Ever (1981) noted that: "Most PCs are programmed in BASIC. In many cases the language is built into the RAM of the machine. As a result most users of small computers learn BASIC as their first, and sometimes their only language". In the October 1980 survey of BYTE magazine almost 50% of the readers indicated that they use BASIC frequently, and close to 90% indicated occasional or intended use.

Wegner (1976) pointed out, the "introduction of a new programming language will inevitably involve some bloodletting." He pointed out further that external influences, courses, seminars and the like will not facilitate the introduction of new languages into an environment entrenched with FORTRAN and COBOL. Bauer (1979) claimed that "Pascal teaches students algorithm design and the creation of well-structured programs."

Mundie (1978) claimed : "If someone were to propose that the outdated Z-8088 CPU be retained in preference to the newer, faster, and more powerful 6868A, simply because every one was already familiar with the older machine, his sanity would probably be questioned."

Similarly, one can question for BASIC. In sum when it comes to the languages used on those machines, the personal computing community seems content to hobble along with a hopelessly inadequate language whose only excuse for existence is that it got there first. This contrast between compulsiveness with regards to machines and our fetish with regards to languages is surely one of the more interesting psychological aspects of the current computing scene. The market for BASIC is just about saturated and if personal computing is to attain its full potential, the many marginally interested members of the general public will have to be won over with a language more suited to their needs than BASIC. No wonder there are so many different versions of BASIC!!! Pascal offers a somewhat wider selection, but avoids the pitfall of trying to incorporate every feature known to man, as PL/1 seems to. Instead of trying to foresee every possible application which might arise, Pascal's designers chose just those features which allow the user to expand the language himself to suit his needs.

Comparison of Programming Languages in Introductory Computer Science Classes

Several comparisons of programming languages have been done using the same guidelines, outlined below:

Qualitative -- Comparison Criteria

-- existence of control structures

- data types and structures
- availability
- adequate diagnostic aids and other programming tools
- interfaces with existing software.
- existence of literature and program libraries
- interfaces with special equipment
- availability of adequate local and vendor support for the implementation

Quantitative -- Comparison criteria

- efficient machine usage - CPU time, memory, I/O requirements.
- Program development costs
- Direct costs - license fees, software maintenance contracts.

Using the above criteria Luker (1985) looked at several programming languages and concluded that "ALGOL 60 was more elegant, with its block structure and range of control statements. BASIC was proving to be very popular for service courses.

It was not the language but its environment that was attractive." Once Knuth (1981) had branded the GOTO as harmful, no self-respecting Computer Science department could count on any further use of BASIC. ALGOL 60 was getting a little long, and its paucity of data types was proving rather inconvenient. Horne and Wirth introduced ALGOL W, ALGOL 68 became something of a cult language in some universities. For most, though, it was too large, too powerful and too forgiving.

In Europe, SIMULA 67 had a strong following. SIMULA extended the range of types, cleaned up some of the less pleasant aspects of ALGOL 60, and added the CLASS, which at once provided coroutines and support for object-oriented programming. Pascal provided strong typing together with a sufficient set of types and control constructs that were being demanded in 1970s, and it is a small language in the sense that Pascal has a very small set of keywords and constructs.

Particular language comparisons and evaluations have also been extensively conducted in the past few years. Most significant among these was the comprehensive evaluation of twenty-two candidate languages for the DOD (Department Of Defense), using the STEELMAN requirements, which resulted in the conclusion that the new language Ada was needed. Other language evaluations and comparisons have been conducted in the past by several researchers. M. Shaw et. al. (1981) compared FORTRAN, COBOL, Jovial, and Ada for their support of good software engineering practices, Ada was the leader in this category followed by FORTRAN, COBOL, and Jovial. H. J. Bloom and E. De Jong (1980) used ALGOL 60, FORTRAN, Pascal, and ALGOL 68 for their comparison when they were implemented on a CDC CYBER 73. Pascal, ALGOL 68, Algol 60 were the leaders in this case whereas FORTRAN was behind ALGOL 60. Ernst and Wang (1977) used control structures as a key point when comparing ALGOL, Pascal, and FORTRAN, They claimed Pascal with the best control structures followed by ALGOL and FORTRAN.

Tharp (1977) compared COBOL, FORTRAN, PL/I and Spibol and came to the conclusion that each of these languages were good at different applications. FORTRAN was best suited for scientific applications whereas COBOL was at its best with business applications. Tucker (1986) compared Pascal, FORTRAN, and APL for scientific programming. He concluded that Pascal was the best defined and most portable among the three, APL had the best data structure support, and FORTRAN was the most efficient. When COBOL and PL/I were compared for data processing, he claimed that PL/I was superior to COBOL in data structure, modularity, pedagogy, and generality. On the other hand, COBOL excelled in portability and efficiency. In text processing area SNOBOL and C were compared. His conclusion was that C was clearly superior, overshadowing SNOBOL. For artificial intelligence, when LISP and PROLOG were compared, LISP was found to be slightly better in modularity and efficiency, and PROLOG was found to be slightly better in input-output facilities. Notably, neither rated particularly well in portability, pedagogy, or generality. Neither was particularly efficient because both were principally interpreted languages. Finally, while comparing Ada and Modula-2 for systems programming, he concluded that Ada's input-output facilities, pedagogy, and generality were superior to those of Modula-2. On the other hand, Modula-2 found to be superior in well-definedness, data types and structures, and efficiency..

Zeil & Scott (1987) on one hand complained that the syntax trees of (Pascal) for semi-colons are tortuous and error-prone. There is no access to random access files. And on the other hand defended the fact that an implementation of Pascal is efficient and in addition, Pascal has a very simple structure that allows a concise semantic definition. It has been used in connection with the verification of programs and it is available on an increasing number of computers." Alsbaugh (1972) demanded that "good programming principles and techniques should be taught in the first course, but the use of standard FORTRAN impedes learning them, because FORTRAN does not avoid non-structured programming; it does not have the capability of creating separate modules; the I/O is not user-friendly." Weare (1976) was also convinced that in order to teach good programming techniques, FORTRAN should not be used as a first language, even though it could be taught in a service course.

Ohler (1976) claimed that kids who program in BASIC develop so many bad programming habits that making the transition to the more structured languages that are used in the programming profession (such as C or Pascal) becomes extremely difficult. He believes that BASIC is a better language to use in order to get acquainted with programming, but it is not a good production language. Solntseff (1976) divided the users into the following groups so that appropriate language can be chosen for them: The "casual user", the "general user", and the "professional user." The "casual user" is the student for whom an acquaintance with computer-aided problem solving is part of his/her general

education. The "general user" is the student who will make extensive use of the computer during his/her stay at the university and during his/her subsequent career, but who will not be needing specialized knowledge about computer systems. The "professional user" is the student with a major in Management, Engineering, or Computer Science. According to Solntseff the casual user is best served by BASIC or SPEAKEASY or Logo since anything more than a very low "approach threshold" will have the effect of turning the students away from the computer.

Summary

The preceding survey of the literature has shown that there are a few studies focusing on the selection of programming languages in the first Computer Science course. Most studies dealing with programming languages consist of comparisons of those languages and/or survey of languages used in the computer science curricula in the 4-year colleges and/or universities.

There are quite a few studies of teaching first year programming dealing with the content of the course, ACM guidelines and comparison of different languages used in the first programming language course. However, there is a growing need of changing to a different programming language in the first Computer Science course. Some studies seek to establish the importance of structured programming in the first programming language classes and hence to

establish the importance of procedural programming languages. Then, what about Object-Oriented programming languages? What about the demands from the industry? What does the job market ask for?

This study surveys the views of faculty in selecting the first programming language and the reasons behind the selection of this language. The study also surveys students who are in these first programming classes and tries to find out their views about their first programming language and the reasons behind the selection of their first programming language. The study solicits the general attitudes of those students as well as those faculty on specific issues relating to the selection of the first programming language. The study also explores the relationship between general attitudes of the faculty and the students and certain demographic variables.

Finally, the follow-up interviews of those faculty should provide some detailed views about the first programming language, the selection procedure within their department, their views about the ACM guidelines for the first programming language class and see if their views and reasons for selecting a particular programming language coincide with those of the students in the first programming language classes.

CHAPTER III

METHODOLOGY

Purpose and Rationale

This exploratory study was designed to investigate faculty concerns and student opinions regarding the possible first programming language to be used in introductory Computer Science courses in North Carolina four-year colleges and universities. The ultimate goal of this research was

1. To provide useful information about first programming language for faculty and students.
2. To help those schools with very little or no resources who are thinking about introducing Computer Science courses in their curriculum for the first time.
3. To stipulate consensus in Computer Science Education regarding the selection of programming language in introductory Computer Science classes and in turn help reform Computer Science Education.
4. To involve students in the decision process of selecting the first programming language.

Research Questions

The general research questions motivating this study were:

1. What factors influence faculty members in choosing a particular programming language for their introductory Computer Science courses?
2. What are students' views about choice of a programming language for their future use?
3. What are the reasons students learn a particular programming language as their first programming language?
4. How do faculty and students' views of the particular programming language used in the first programming language class differ?
5. How do factors influencing faculty and students' choice of a first programming language correlate?

The specific research questions involved in the study were:

1. Which was the first programming language learned by the students?
2. What were the reasons behind learning the first language?
3. What was the second programming language learned by the students?
4. What were the reasons behind learning the second language?
5. If given another chance to study a first programming language, which language would the students choose to learn? Why?
6. Which programming language(s) were taught as the first programming language?

7. What were the reasons behind the selection?
8. Which programming language(s) were used in the second programming language course?
9. What was the decision process for faculty choosing the first programming language?
10. What was the purpose of using the language as the first programming language?
11. What were the most important factors for choosing a programming language in the first programming language class?
12. Were faculty choosing a new language? Why? When?
13. Were students involved in the above decision process?

Two methods were used to measure the factors involved in choosing a programming language for introductory Computer Science courses: (1) survey questionnaires, and (2) interviews. First, a survey questionnaire was administered to the students of introductory Computer Science classes from those four-year colleges and universities in North Carolina which consented to participate in the study; and a survey questionnaire was administered to the faculty teaching introductory Computer Science classes from those schools.

The second method consisted of open-ended interviews of the same faculty. The survey and the interviews were conducted during the Spring 1993 semester.

Design of the Survey

Two sets of questionnaires were designed by the researcher: one for students in introductory Computer Science classes from all four-year colleges and universities in North Carolina and the other for faculty who were teaching these classes.

The students' questionnaire consisted of five statements to be evaluated by the students plus five demographic questions. Questions 6 through 10 were directed at specific issues that are related to the selection of introductory programming language.

There was a practical reason for limiting the number of statements to ten so that it can be administered within fifteen minutes. Try-outs conducted among students at Meredith College showed that it takes nine minutes to evaluate five statements of the type used in the questionnaire and another three minutes to do a set of questions on demographic variables, for a total time of twelve minutes.

The faculty questionnaire consisted of fourteen statements plus two demographic questions. Questions 3 through 16 were directed at specific issues that are related to the selection of a introductory programming language.

The number of statements in the faculty questionnaire was also kept to a minimum so that the questionnaire could be completed within fifteen minutes. Several pretests conducted among adult friends at Meredith College showed that it takes an average of ten minutes and another minute to complete the

questions on demographic variables, for a total average time of eleven minutes. The open-ended questions were reserved for the faculty interview which were given about thirty minutes of time.

All the demographic questions were put at the beginning, the reason being that placing them at the very end might have created bias in the mind of the respondent, or even reluctance to respond. This also allowed more time at the end for the respondents to evaluate important questions in the study. Questionnaires for the students were either hand delivered to the faculty teaching these classes or mailed using first class mail and either were picked up later the same day or mailed by the faculty in the postage-paid envelope provided by the researcher. Questionnaires for the faculty were delivered via first class mail or hand-delivered the same day as the students' questionnaire and collected via postage-paid envelope provided by the researcher or on the same day of the interview in order to get prompt responses from the faculty teaching these classes.

The section of the questionnaire for students on demographic variables is preceded by a statement justifying gathering personal data on respondents, and a note of reassurance on anonymity of responses. The demographic variables collected for students were: college class, age, sex, number of computer courses taken, and major. The respondent was not asked to indicate the name of his/her institution. However, this information was obtained from either the faculty to whom the questionnaire was mailed or hand delivered. The section of the

questionnaire for faculty on demographic variables was preceded by a statement justifying gathering personal data on respondents, and a note of reassurance on anonymity of responses. The demographic variables collected for faculty were: department name, number of years with the department. The respondent was not asked to indicate the name of his/her institution. However, this information was obtained from either the faculty to whom the questionnaire was mailed or hand delivered or when he/she was interviewed.

Subjects for this study were undergraduate students from introductory Computer Science classes and the faculty who were teaching these classes from four-year colleges and universities in North Carolina. In case of multi-section classes, the sections whose faculty agreed to participate in the study were chosen. The list of twenty participating schools was as follows:

- Belmont Abbey College
- Brevard College
- Campbell University
- Chowan College
- Davidson College
- Elizabeth City State University
- Lenoir-Rhyne College
- Louisburg College
- Meredith College
- Mount Olive College
- North Carolina Agricultural And Technical State University
- North Carolina Central University
- North Carolina State University
- Pembroke State University
- Saint Augustine's College
- Shaw University
- University of North Carolina at Asheville
- University of North Carolina at Chapel Hill
- Wake-Forest University
- Western Carolina University

The final forms of students' and faculty questionnaires are shown in Appendix B and in Appendix C respectively.

Sampling and Administration of the Student and Faculty Survey

Preliminary try-outs of the student questionnaire were conducted with 22 students from Meredith College in Computer Programming I during Fall 1992 semester for the purpose of refining the format and language of the questionnaire and to check on response time. There were a few improvements in the phrasing of questions resulting from these try-outs. Average total response time was under ten minutes.

From the outset, a hand-delivered questionnaire approach to all the 58 schools was ruled out due to the traveling distance and time required to do this. Since mailed questionnaires generate a very low response rate, it was decided that the researcher would hand deliver the questionnaire for the local schools and mail the questionnaire for the schools which were more than one hour of driving time.

Before the mailing and/or delivering the questionnaire, the researcher mailed a letter explaining the purpose of the study. The letters were mailed to all the 58 schools in North Carolina. The letters were mailed to the department heads and/or chairs in the Computer Science related areas. The letters included 3 pages: first page was addressed to the department chair and/or head; second

page was addressed to the faculty explaining in details the purpose of the letter and the third page asked the faculty to send their names, telephone numbers, office hours, best time to call, home telephone number (if possible), the names of the courses currently being taught, and the number of students in the programming language courses. The letter also included a self-stamped envelope with the address of the researcher to return the third page with the information in that envelope. The respondents were asked to deliver the letter to the appropriate faculty teaching the first programming language classes.

Preliminary try-outs were conducted with eleven adults for the purpose of refining the format and language of the questionnaire and to check on response time for faculty. Three of the respondents were non-faculty computer science professionals who were working with different programming languages and eight were faculty members who were either currently teaching Computer Programming I classes or who had taught these classes before under the ACM guidelines. There were a few improvements in the phrasing of questions resulting from these try-outs. Average response time was under fifteen minutes.

Based on the number of students in the Computer Programming I classes, the students' questionnaire were either hand-delivered by the researcher or mailed using a first-class postage. All faculty were requested to mail back the completed student questionnaire in the self-addressed postage-paid envelope as soon as possible.

In case of the faculty questionnaire, researcher sent letters to the department chairs or deans along with a consent letter addressed to the faculty who was/were teaching Computer Programming I classes. These letters included a self-addressed, postage-paid envelope to get the consent letters back at the earliest time and to find out the name, address and telephone number of the responsible faculty taking part in the study. The telephone number was later used to contact the faculty for the open-ended interview and the administration of student questionnaire.

The student questionnaire and faculty questionnaire was administered during the last week of classes in April, 1993. Out of the 58 four-year colleges and/or universities in North Carolina, 20 responses were obtained out of 58 faculty members who wanted to take part in the study, six schools did not offer any programming language classes and hence did not want to participate in the study, seven schools did not offer programming language classes during Spring semester and hence could not participate in the study. So, 33 out of 58 (57%) schools responded to the researcher's request. Total number of student responses were 322 from 20 different schools in North Carolina.

Administration of the Open-Ended Interviews

In the Spring of 1993, open-ended interviews were conducted for the faculty members who had taken part in the survey. All the interviews were either conducted in person or by telephone. All interviews were audio-recorded after receiving permission of the subject. There were several purposes of the interview. One was to get to know the program of the school and the conditions these courses were offered. Another purpose was to gather a non-quantitative data.

Accordingly, 20 faculty took part in the interview process. Each of these people were approached by letter or in person or through a telephone call for permission to be part of the study. Wherever possible, some of the faculty were interviewed in person at their offices while others were interviewed by telephone. The plan was for the interviewer to engage the interviewee in a conversation with the purpose of getting information in the following areas:

1. The faculty's thoughts about the introductory Computer Science course, particularly with regards to its content and the programming language used.
2. What, if any, changes the faculty plans for introductory Computer Science course in the future?
3. The faculty's thoughts about which programming language(s) to use in introductory Computer Science course in the future.

4. What do the faculty feel about the current ACM guidelines with regards to introductory Computer Science (CS1) curriculum.

5. The faculty's thoughts about the selection process of a programming language used in introductory Computer Science (CS1) course at his/her school.

6. What does the faculty think about students' input in the decision process of selecting a programming language for CS1 course.

7. Does the faculty suggest any changes in the selection process?

In early April, 1993, along with the faculty questionnaire, a letter was mailed to the 58 faculty asking their help to participate in the telephone interview. In the letter they were asked to send their telephone number(s) and best time to do the interview.

The letter described the purpose of the study, the approximate length of the interview and stated that the interview would be audio taped. Respondents were asked to signify their assent by signing a consent form. The letter and consent form is Appendix D.

20 positive responses were received. The interviews were conducted between April 15, 1993 and May 31, 1993, with each interview lasting an average of 30 minutes. Transcripts of the interviews were made and edited to remove references to names and places so as to protect the anonymity of the respondents. Representative edited transcripts are found in Appendix J.

Methods of Analysis

The Student Survey

The student respondents were instructed to try to respond to all the questions. Of the 322 questionnaires that were returned, only 20 questionnaires revealed missing values for questions 1 to 10 after visual examination. On the average, one questionnaire was with missing values from each of the 20 schools who were participating in the study. So, these were eliminated from the pile.

The 302 valid returns were then coded. Excluding respondent number, 11 variables were coded. Of these, 10 came directly from the questionnaire. One was added by the researcher, namely, the school of the respondent. To keep track of the school, each questionnaire was assigned a case number which included 1 - 8 characters from their school name followed by an integer between one and the number of questionnaires from that school. The same label was used to keep track of each completed questionnaire as they came in.

The 302 questionnaires of the students were entered into the 386 (X20) PACK-MATE by Packard Bell personal computer owned by the researcher and were analyzed using the Minitab Statistical Package (MINITAB, Version 8.1). Each value of each of the variables was coded using integer values. Frequency tables were first obtained for each of the 10 variables for each of the student questionnaire.

Cross-tabulations were done using Minitab among different variables. Using the program Lotus 1-2-3 for Windows Version 3.1, frequency tables and frequency graphs were created for each of the variables.

The Faculty Survey

The 20 questionnaires for the faculty were entered into the same personal computer and were analyzed using MINITAB. Frequency tables were first obtained for each of the first 4 variables, and variables 10 through 14. For the remaining questions, first each of them were coded using integer values.

Using the Lotus 1-2-3 program for Windows Version 3.1, frequency tables and frequency graphs were created for each of the 16 variables. Cross-tabulations of these variables were done using Minitab.

The Open-Ended Faculty Interviews

An informal content analysis was made of the interviews to see how the interviewees responded to questions on such topics as ACM guidelines, future of programming languages, etc. Mostly Used language of the future, Student participation in the selection of the first programming language, if students are going to participate in the above decision, what are the best ways to deal with this, relation between the industry demands and choice of a language.

Limitations of Study

This study is primarily an investigation of the programming languages used in introductory Computer Science classes for a sample of students and faculty from four-year colleges and universities in North Carolina. Any inferences are, of course, limited to this sample.

Being descriptive, the data is naturally limited in practical value within a certain time. All the answers are valid for certain time. As the need changes, the language requirements, the language designs will change accordingly and hence might change the answers accordingly.

Every faculty had different objectives about introductory Computer Science classes. The course objectives differ from school to school and from faculty to faculty (one introductory course is not equal to another introductory course -- especially how much content is devoted to a programming language).

The researcher has his own views about the language selection in introductory Computer Science classes. His views were not expressed whenever personal contact with faculty involved in study was established. This attempted to avoid any researcher bias while the study was going on. Nonetheless, some bias may have been communicated informally, nonverbally -- albeit unintentionally.

CHAPTER IV

RESULTS, ANALYSIS AND DISCUSSION

This chapter analyzes student and faculty responses to survey questionnaires and faculty interviews administered in this study.

Analysis of the Student Survey

The first discussion will focus on the frequency tables for each of the variables in the student questionnaire. All the student questions will be divided into three groups: the first group of questions will include first five questions dealing with their class, age-group, sex, number of computer courses taken, and their major if declared, otherwise their prospective major. The second group will include next two questions which were dealing with their experience with programming languages and the mostly used programming languages. For this group, every question will be a contingency table showing the responses broken down into the seven categories of the Likert scale with the order of the categories set up so that expert counts come first. The other type of frequency table will be in the form of a bar graph wherein the seven categories are collapsed into three which are as follows: familiar, novice, never used.

The third group will involve the last three questions which mainly deal with the first and second programming languages learned and the reasons behind selecting those languages.

The second discussion will center on the cross tabulations of demographic variables against the four variables, namely, first programming language, reasons for selecting the first programming language, second programming language, and the reasons for selecting the second programming language. After all the questions in a group are presented this way, a composite table for the responses to all the questions will be presented and discussed, followed by a corresponding graph. This pattern will be repeated for all three groups of questions.

Group I - Students Taking the First Programming Language Class

Five questions to count the number of respondents taking the first programming language class were asked as follows:

1. What is your classification depending upon the number of total credits?
2. What is your age group?
3. What is your sex?
4. How many computer courses already have you taken?
5. What is your major?

Table 1 shows the breakdown of responses to the first question and the same data is graphically depicted in Figure 1. 13% of the students were freshman, 32% were juniors, 21% were sophomores, 27% were seniors, and remaining 7% were either graduate students or non-degree students. Since this is supposed to be the first programming language class, the number of freshman taking this class was small (13%) as compared to juniors (32%). So, this suggests that the freshman students are taking some other computer related classes before signing up for this programming language class. By looking at the number of sophomores (21%) and the number of seniors (27%), 48% of students are waiting to take this programming language class for at least two years. This suggests that more non-computer science related major students are taking the first programming language class before they graduate which in turn suggests that they are making themselves marketable for the job market.

Table 2 shows that the majority of students who were taking the first programming language class are between the ages of 23 and 27 (73%) and only 8% of the students who were taking the first programming language class were between the ages of 43 and 62 which suggest that either the age of these students is at least one year higher than a average age of a college student (18 through 22) or there are a lot of re-entry students in these first programming language classes. Same data is graphically depicted in Figure 2.

Table 3 shows that the male (52%) and female (48%) students who were taking the first programming language class were equally distributed and Figure 3 is graphically depicting the same data.

From Table 1 we have seen that more students are taking the first programming language classes during their last two years of college and hence Table 4 and Figure 4 show that 49% of students who were taking the first programming language class had already taken 4 or more than 4 computer courses and the students who had taken between 1 and 3 computer courses before they took the first programming language classes were 51%.

Table 5 shows that almost every student (98%) who was taking the first programming language class had declared his/her major. From this group, 70% students had computer science related major, 12% were Mathematics majors and the remaining 16% came from all other majors. Table 4 also shows that more than 96% of the students in the programming language class were hard-core science majors and only 4% were from humanities. Same data is graphically depicted in Figure 5.

Table 1

V1 Classification of Students by Their Class

CLASS	COUNT
Freshman	38
Junior	97
Sophomore	63
Senior	81
Graduate	4
Non-Degree	4
Total	302

Table 2

V2 Classification of Students by Age

AGE	COUNT
18 - 22	5
23 - 27	219
28 - 32	37
33 - 47	18
38 - 42	14
43 - 47	5
48 - 52	2
53 - 57	1
58 - 62	1
63 - ABOVE	0
TOTAL	302

Table 3

V3 Classification of Students by Sex

SEX	COUNT
Male	157
Female	145
Total	302

Table 4

V4 Classification of Students by the Number of Courses Taken

NO. OF COMPUTER COURSES	COUNT
1	55
2	50
3	47
4	31
More than 4	119
Total	302

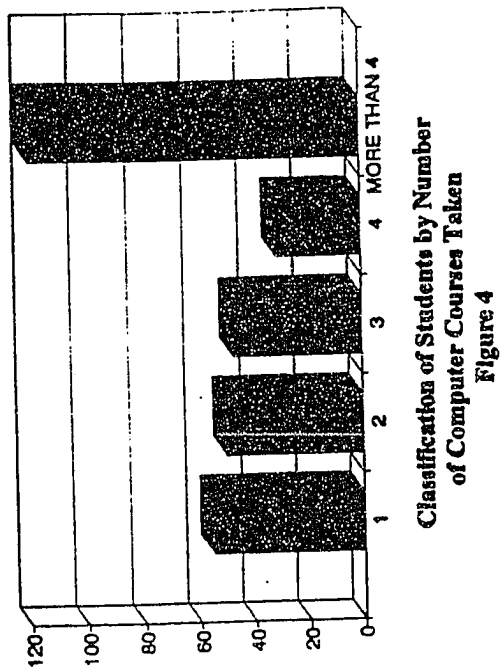
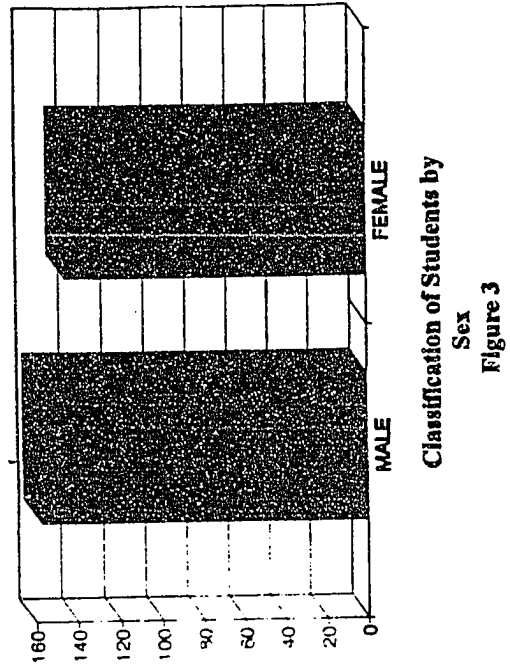
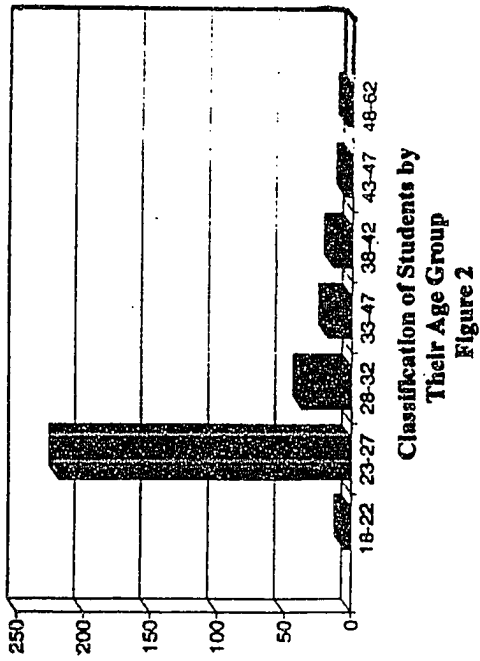
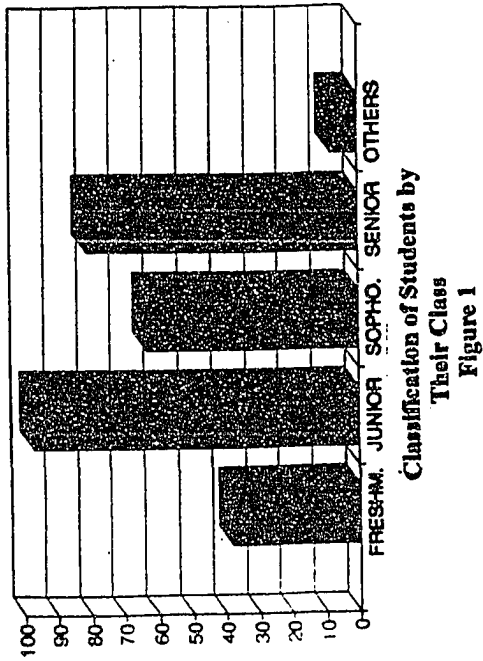


Table 5

V5 Classification of Students by Majors

MAJOR	COUNT
Major Is Not Declared	5
Aerospace Engineering	4
Biology	1
Business	14
Chemistry	5
Civil Engineering	1
Communications	1
Computer Information Systems	56
Computer Science	150
Computer Systems Engineering	4
Economics	1
Electrical Engineering	10
English	1
Environmental Science	1
Government	1
History	3
Mathematics	33
Mathematics Education	1
Mechanical Engineering	2
Mathematics & Computer Science	0
Music	1
Political Science	1
Philosophy	2
Religion/Christian Ministries	1
Spanish	1
Statistics	3
Total	302

Group II - Experience with the Programming Languages

Two questions to count the number of respondents taking the first programming language class were asked as follows:

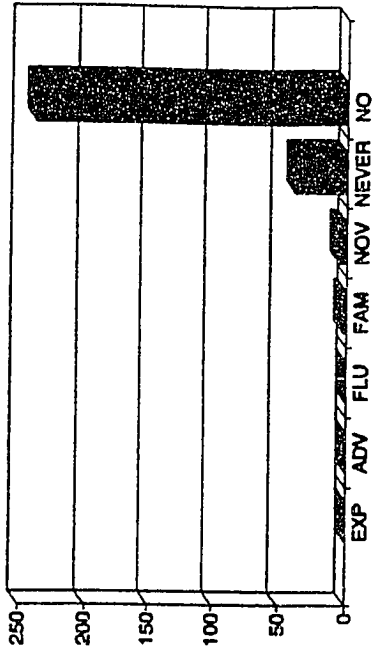
6. Rate your experience with the programming languages.
7. Which programming language do you use most in doing your programming?

Table 6 shows the breakdown of responses for Question 6 with the collapsed version graphically shown in Figures 6 through 24. This is one issue where respondents show a lot of ambivalence, as shown by the high number of 7 responses. More than 94% of the students either had no knowledge or never used languages like ADA, APL, SCHEME, FORTH, HYPERTALK, LISP, LOGO, MODULA-2, PL/I, PROLOG, SMALLTALK, TURING for their programming purposes. Almost 70% of the students either had never used or had no knowledge of Assembly/Machine language. So, only 30% of the students had experience with this language. Almost 70% of the respondents either were experts and/or familiar with BASIC programming. More than 30% of the respondents were programming using C language. C++ was far behind C in this matter (only 17% had shown experience with C++). There were almost 32% experts COBOL programmers in the survey. The share of FORTRAN was around 22% in this category. PASCAL had the largest share amongst all the languages.

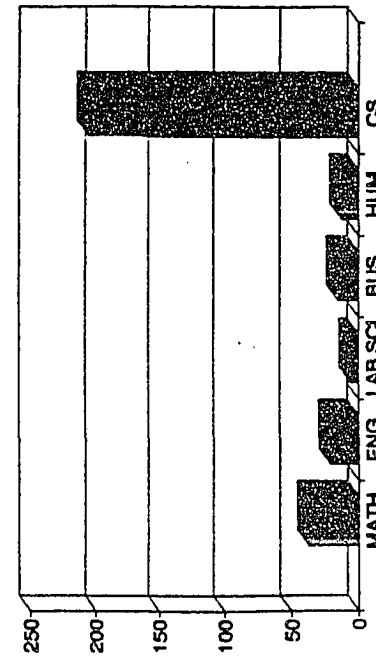
More than 80% of the respondents were programming using PASCAL. The languages which were not listed had almost 16% of the share of the programmers. In this case, the widely mentioned language was dBASE.

Table 7 shows the breakdown of responses for Question 7 with the collapsed version graphically shown in Figure 25. From the data in Table 7, it is clear that the least used programming languages were ADA (1%), APL (0%), Assembly/Machine (7%), LISP (2%), HYPERTALK (0%), SCHEME (0%), LOGO (2%), MODULA-2 (4%), PL/I (0%), PROLOG (0%), SMALLTALK (0%), TURING (1%). The languages which were heavily used by the respondents in their programming were PASCAL (70%), BASIC (29%), C (24%), COBOL (20%), FORTRAN (12%), and C++ had a share of 10%. The languages which were not in the list but were used heavily in the programming by the respondents had their share of 12% and most mentioned language in this category was dBASE.

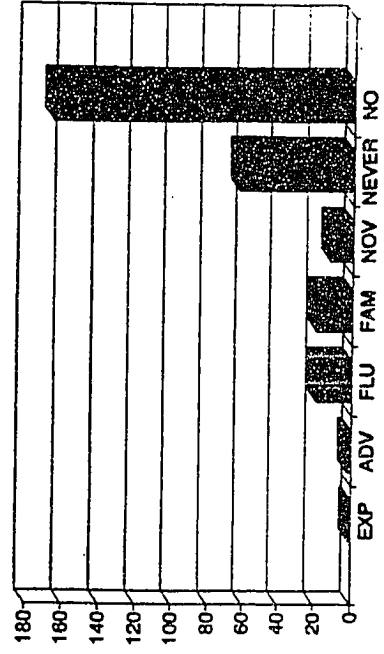
Data from these two tables clearly shows that Pascal was one of the widely used language by the respondents. Languages like BASIC, FORTRAN, COBOL were close second to PASCAL. Languages like C and C++ are trying to catch the number two spot among the most used languages by the respondents.



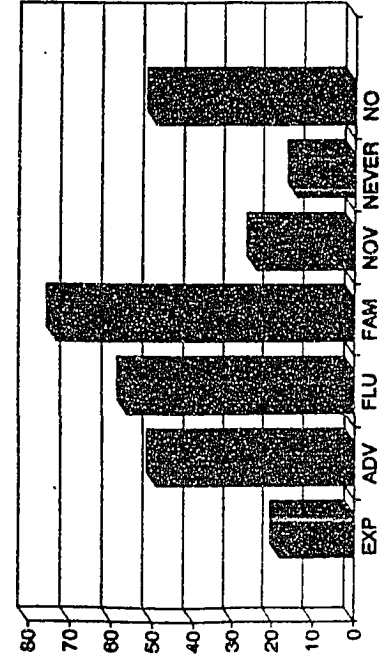
Classification of Students by Their Experience With ADA
Figure 6



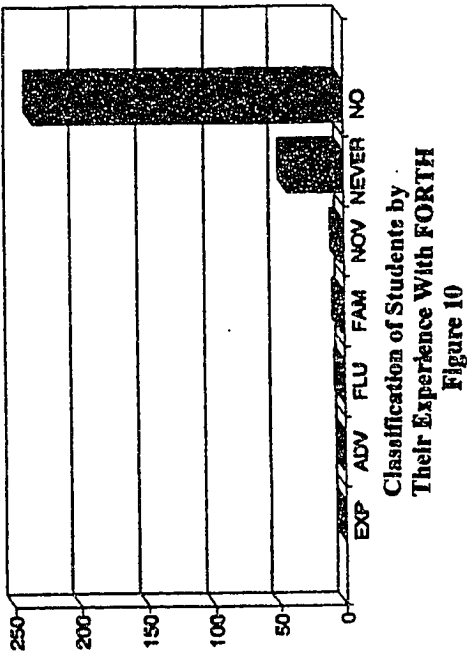
Classification of Students by Their Major (If Declared)
Figure 5



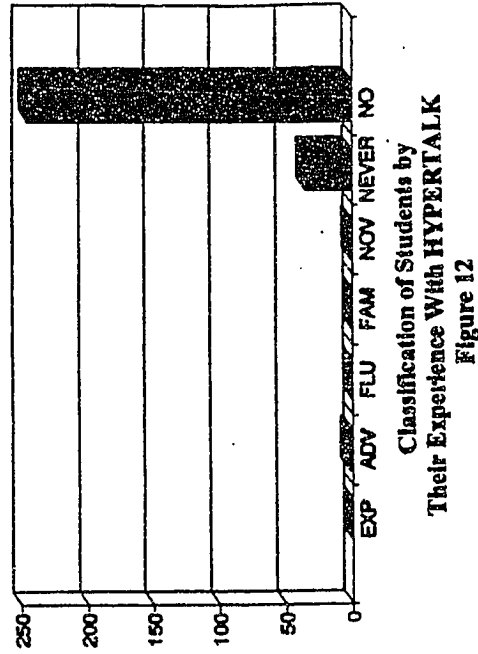
Classification of Students by Their Experience With C++
Figure 8



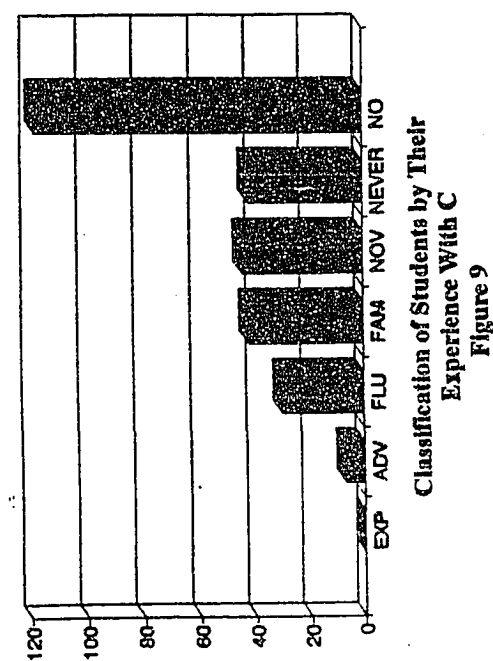
Classification of Students by Their Experience With BASIC
Figure 7



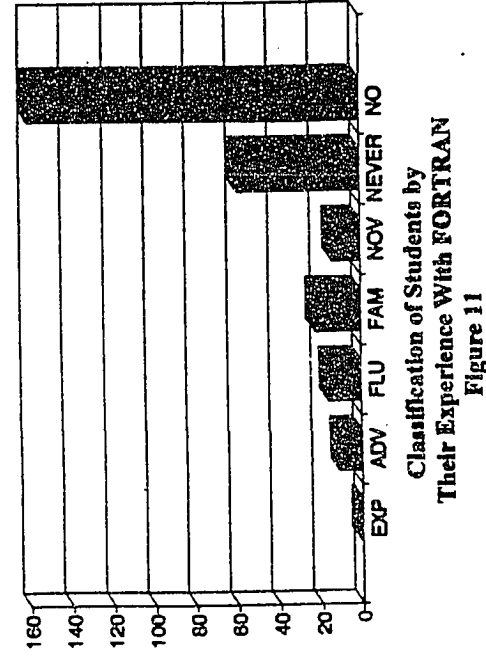
Classification of Students by Their Experience With FORTH
Figure 10



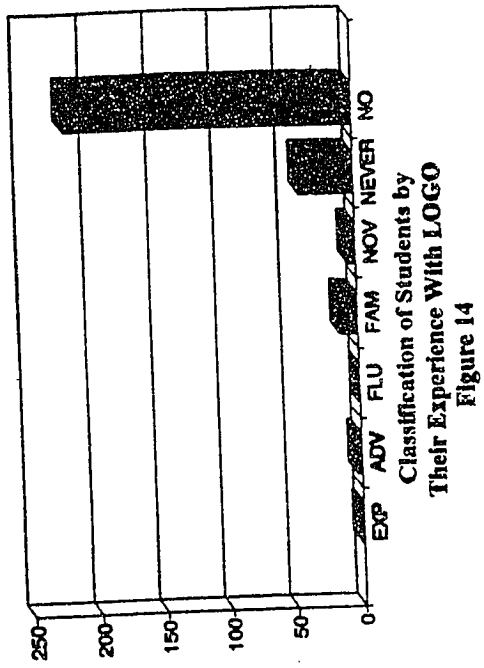
Classification of Students by Their Experience With HYPERTALK
Figure 12



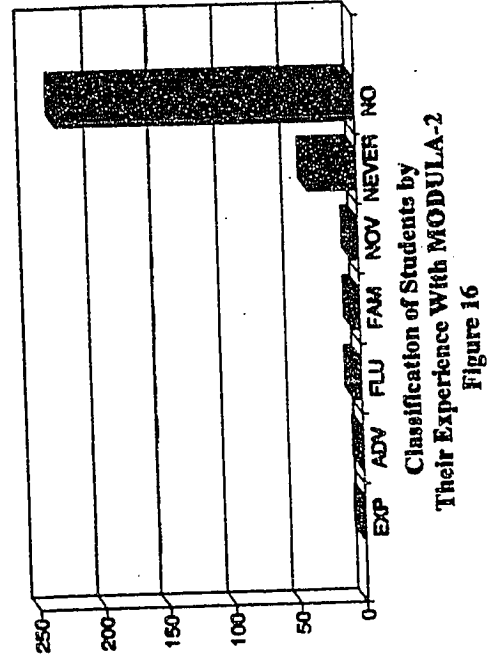
Classification of Students by Their Experience With C
Figure 9



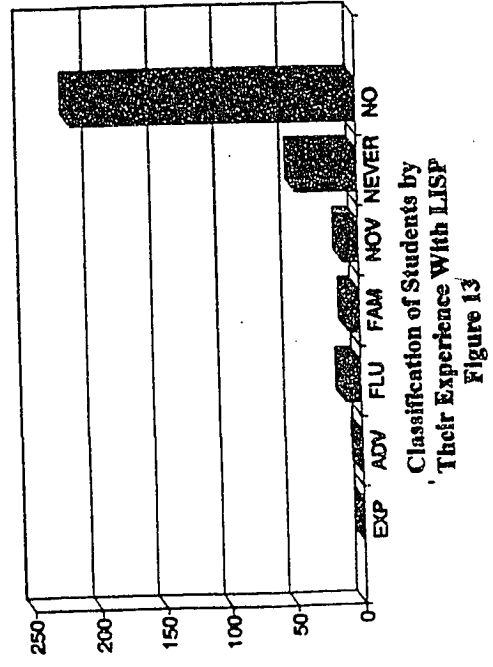
Classification of Students by Their Experience With FORTRAN
Figure 11



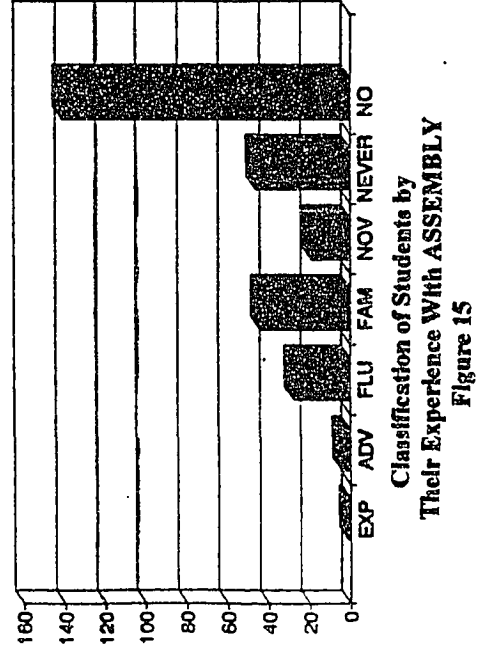
Classification of Students by
Their Experience With LOGO
Figure 14



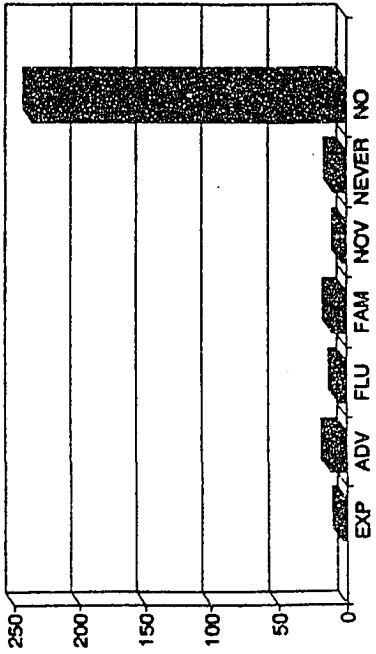
Classification of Students by
Their Experience With MODULA-2
Figure 16



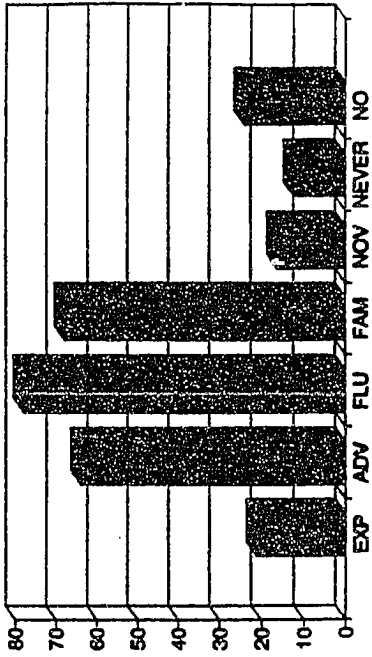
Classification of Students by
Their Experience With LISP
Figure 13



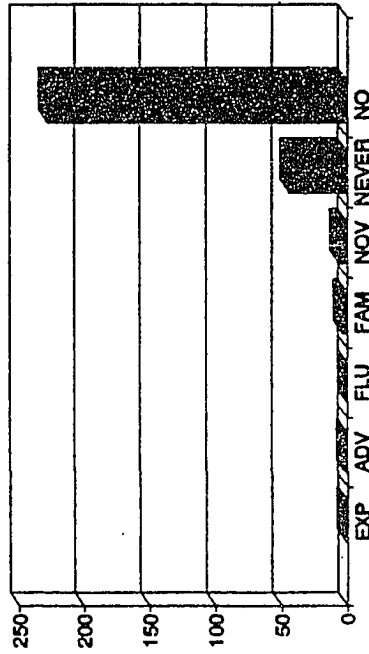
Classification of Students by
Their Experience With ASSEMBLY
Figure 15



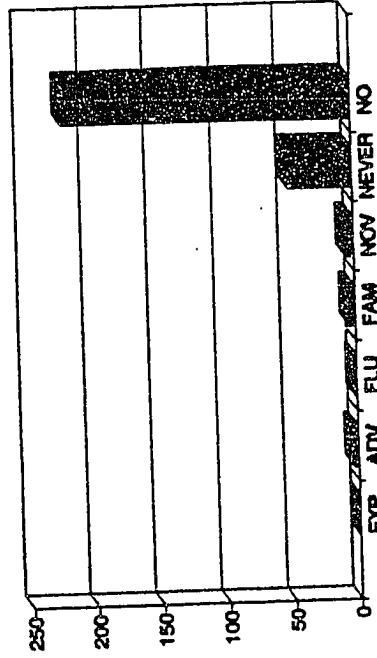
Classification of Students by Their Experience With Other Lang.
Figure 17



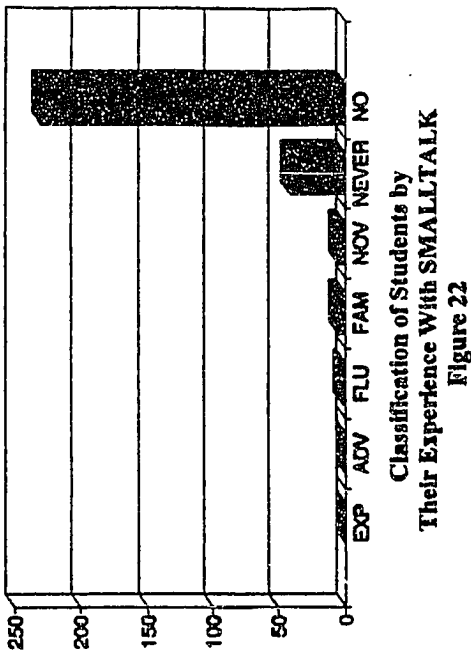
Classification of Students by Their Experience With PASCAL
Figure 18



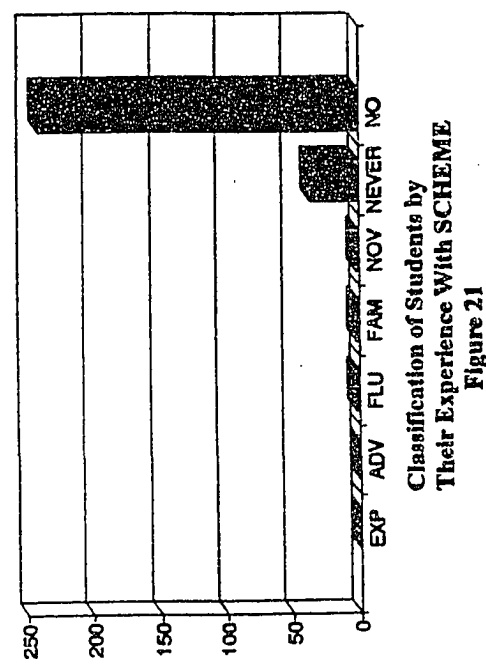
Classification of Students by Their Experience With PL/I
Figure 19



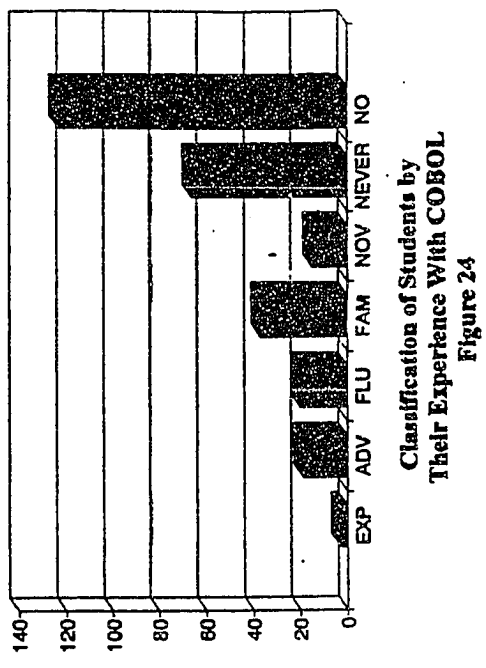
Classification of Students by Their Experience With PROLOG
Figure 20



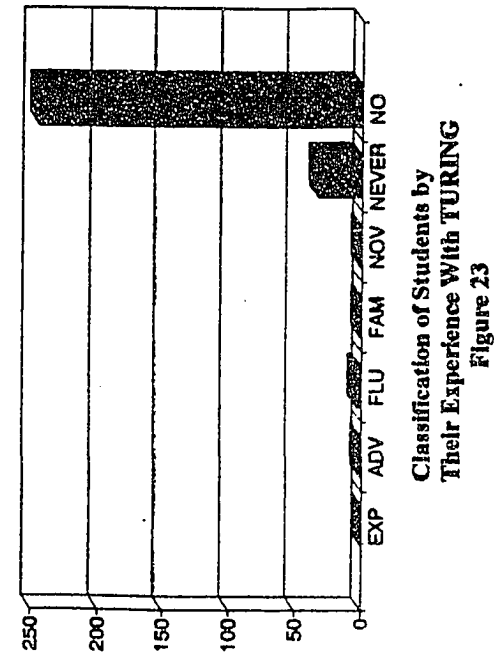
Classification of Students by Their Experience With SMALLTALK
Figure 22



Classification of Students by Their Experience With SCHEME
Figure 21



Classification of Students by Their Experience With COBOL
Figure 24



Classification of Students by Their Experience With TURING
Figure 23

Table 6

V6 Classification of Students by Experience with Programming Languages

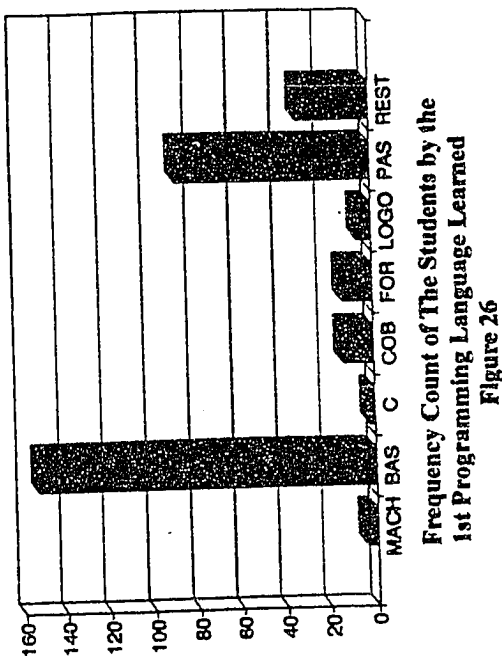
LANGUAGE	1	2	3	4	5	6	7	TOTAL
ADA	0	0	0	6	16	59	221	302
APL	0	0	0	2	5	57	238	302
ASSEMBLY/MACHINE	1	4	33	49	19	55	141	302
BASIC	18	54	61	78	28	14	49	302
C	0	7	35	47	49	46	118	302
C++	1	2	24	25	17	67	166	302
COBOL	3	24	25	42	19	66	123	302
SCHEME	0	0	2	2	2	45	251	302
FORTH	0	0	1	2	3	53	243	302
FORTRAN	1	16	21	27	18	59	160	302
HYPERTALK	0	2	0	0	1	40	259	302
LISP	0	1	1	8	11	56	225	302
LOGO	1	3	0	13	6	50	229	302
MODULA-2	1	1	6	5	9	42	238	302
PASCAL	21	69	83	73	20	12	24	302
PL/I	1	1	0	3	6	49	242	302
PROLOG	0	3	1	4	6	53	235	302
SMALLTALK	0	0	2	5	5	45	245	302
TURING	0	1	3	1	1	37	259	302
OTHER	3	21	11	14	8	9	236	302

Where 1 = Expert, 2 = Advanced, 3 = Fluent, 4 = Familiar, 5 = Novice, 6 = Never Use, 7 = No Knowledge whatsoever

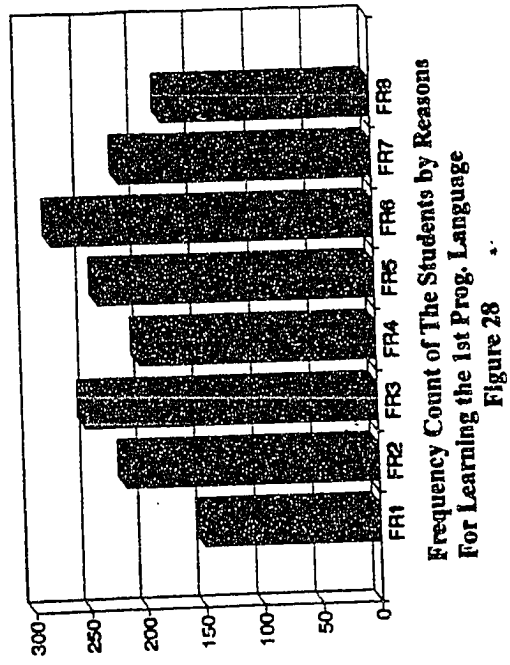
Table 7

V7 Classification of Students by the Most Used of Programming Language

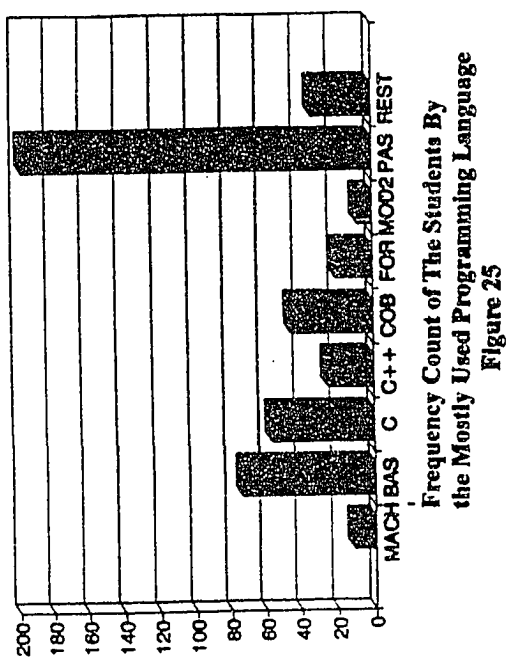
PROGRAMMING LANGUAGE	USE	DO NOT USE	TOTAL
ADA	2	300	302
APL	0	302	302
ASSEMBLY/MACHINE	21	281	302
BASIC	88	214	302
C	71	231	302
C++	29	273	302
COBOL	60	242	302
SCHEME	0	302	302
FORTH	0	302	302
FORTRAN	35	267	302
HYPERTALK	0	302	302
LISP	5	297	302
LOGO	6	296	302
MODULA-2	12	290	302
PASCAL	211	91	302
PL/I	0	302	302
PROLOG	0	302	302
SMALLTALK	0	302	302
TURING	2	300	302
OTHER	37	265	302



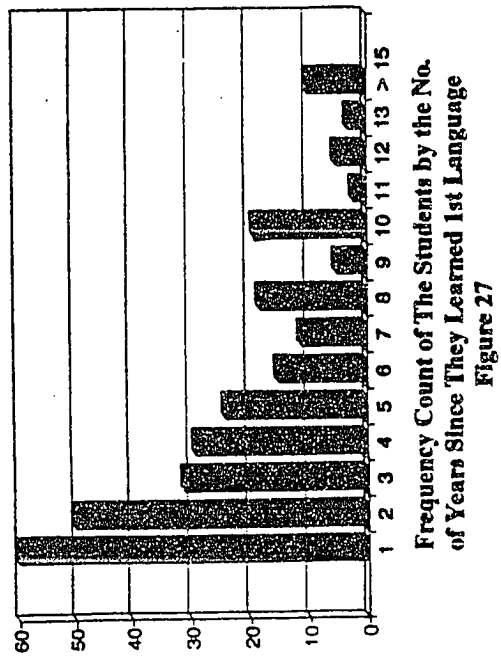
Frequency Count of The Students by the 1st Programming Language Learned
Figure 26



Frequency Count of The Students by Reasons For Learning the 1st Prog. Language
Figure 28



Frequency Count of The Students By the Mostly Used Programming Language
Figure 25



Frequency Count of The Students by the No. of Years Since They Learned 1st Language
Figure 27

Group III - First and Second Programming Languages Learned with the Reasons

Eight questions to count the number of respondents taking the first programming language class were asked as follows:

8. Which was the first programming language you learned?
 - 8(a). How long ago?
 - 8(b). Why did you learn this language?
9. Which was the second programming language you learned?
 - 9(a). How long ago?
 - 9(b). Why did you learn this language?
10. Which programming language would you learn first if you were given another chance of selecting the first programming language?
 - 10(a). What are the reasons behind this selection?

Table 8 shows the breakdown of responses to the first part of Question 8 with the collapsed version graphically shown in Figure 26. BASIC and Pascal were the most popular languages amongst the respondents. BASIC was the first programming language learned by 54% of the respondents. Pascal was selected by 32% as their first programming language. COBOL (4%), FORTRAN (4%), LOGO (2%), C (1%), and Assembly (1%) were the other languages learned by the respondents as their first programming language. Other languages in the list either were not chosen or not too many respondents decided to learn them as their first programming language.

Table 9 shows that more than 62% of the respondents had learned their first programming language within last four years. From the remaining respondents, 30% had learned their first programming language within past 10 years. The graphical version is shown in Figure 27.

Table 10 and Figure 28 tabulate the responses for choosing the first programming language from Table 8. Out of the 9 Reasons listed, obviously, 92% of the students learned their first programming language since the job market demand for the people who knew the language was great. The other reasons they learned the language were that they were told to learn that language (83%) and it was an easy language (82%) to learn. Only 50% learned the language because it was used in the first course in the department. 80% learned the language because of its popularity and only 59% learned the language because it was required for the major. The reason more than 70% learned the language was that they really wanted to learn the language.

Table 11 shows the breakdown of responses to Question 11 with the collapsed version graphically shown in Figure 29. 43% learned Pascal as their second programming language, 18% learned COBOL as their second programming language. C became the third most popular second programming language. Only 6% of the respondents learned Assembly and BASIC as their second programming languages.

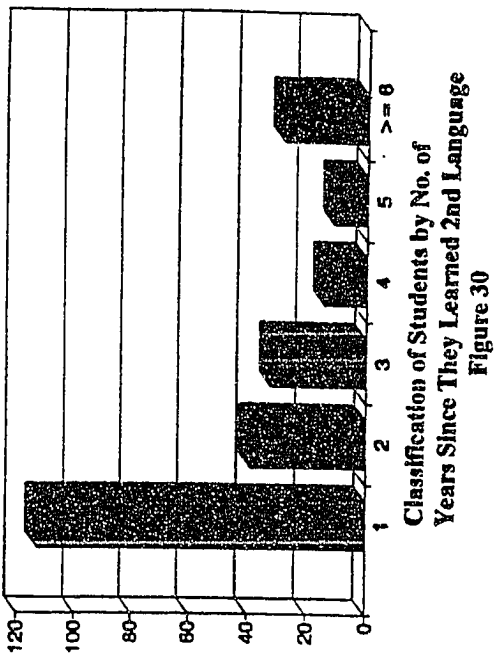
FORTRAN lost its popularity and had only 4% of the respondents who learned it as their second programming language. Nearly 4% of the respondents learned other languages as their second programming language. dBASE came out as a leader in this category.

Table 12 and Figure 30 show that 47% of the respondents took their second programming language class one year ago and more than 87% of the respondents took their second programming language class within the 4 years time.

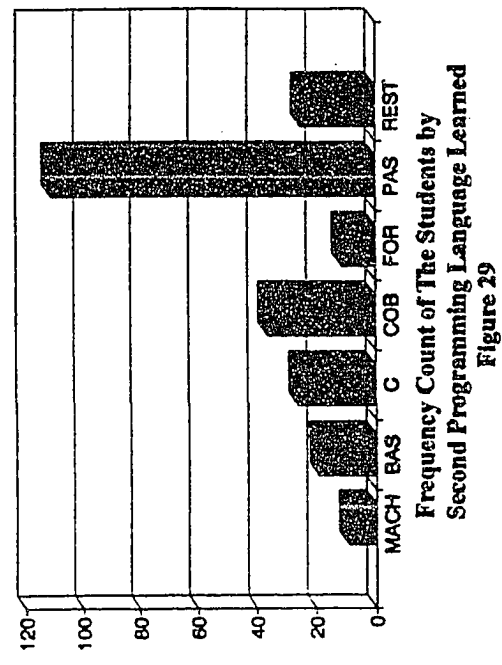
Table 13 tabulates the responses for choosing the second programming language from Table 11 and the same is graphically depicted in Figure 31. Out of the 9 Reasons listed, 73% of the students learned their second programming language since the job market demand for the people who knew the language was great. The other reasons they learned the language were that they were told to learn that language (63%) and it was an easy language (74%) to learn. Only 49% learned the language because it was used in a second course in the department. 65% learned the language because of its popularity and only 36% learned the language because it was required for the major. The reason more than 52% learned the language was that they really wanted to learn the language. Table 14 shows the breakdown of responses to Question 11 with the collapsed version graphically shown in Figure 32. If the respondents were given a chance of taking the first programming language class all over again, 33% wanted to start with Pascal.

Almost 27% would like to start with C or C++ as their first programming language. Only 19% wanted to learn BASIC as their first programming language. 3% of the respondents had no response one way or the other.

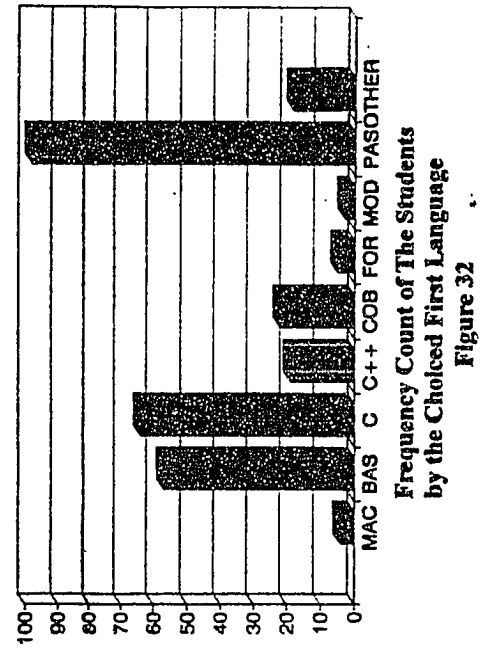
Table 15 tabulates the responses for choosing the new first programming language, if they were given another chance, from Table 14 and the same is graphically depicted in Figure 33. Out of the 10 reasons listed, only 55% of the students wanted to learn their first programming language due to the job market demand for the people who knew the language. The other reasons they wanted to learn the language were that they were told to learn that language (67%) and it was an easy language (80%) to learn. Almost 80% wanted to learn the language because it was the used in most computer science courses in the department. 51% wanted to learn the language because of its popularity and only 34% wanted to learn the language because it was required for the major. The reason more than 78% wanted to start with the new first programming language was that they really wanted to learn the language.



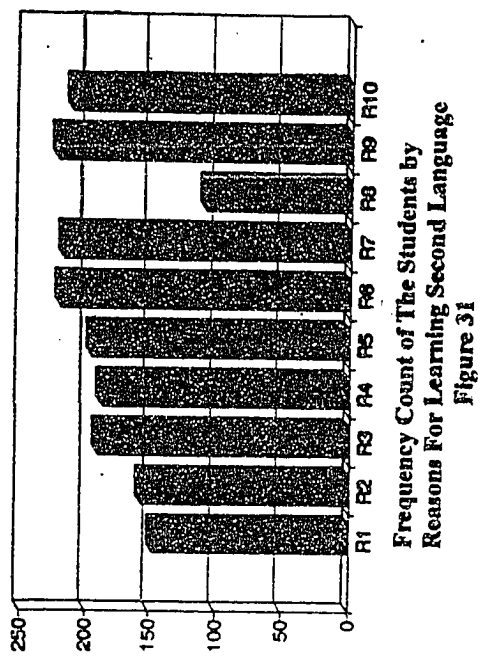
Classification of Students by No. of Years Since They Learned 2nd Language
Figure 30



Frequency Count of The Students by Second Programming Language Learned
Figure 29



Frequency Count of The Students by the Chosen First Language
Figure 32



Frequency Count of The Students by Reasons For Learning Second Language
Figure 31



Table 8

V8 Classification of Students by the First Programming Language Learned

FIRST LANGUAGE LEARNED	COUNT
ADA	0
APL	0
ASSEMBLY/MACHINE	4
BASIC	163
C	3
C++	0
COBOL	13
SCHEME	0
FORTH	0
FORTRAN	13
HYPERTALK	0
LISP	0
LOGO	6
MODULA-2	1
PASCAL	95
PL/I	1
PROLOG	0
SMALLTALK	0
TURING	0
OTHER	3
TOTAL	302

Table 9

V9 Classification of Students by the Number of Years Since They Learned Their First Programming Language

NUMBER OF YEARS SINCE THEY LEARNED FIRST LANGUAGE	COUNT
0	1
1	64
2	55
3	36
4	33
5	24
6	15
7	11
8	18
9	5
10	19
11	2
12	5
13	3
14	0
15	6
16	1
17	0
18	1
19	0
20	1
21	0
22	1
TOTAL	302

Table 10

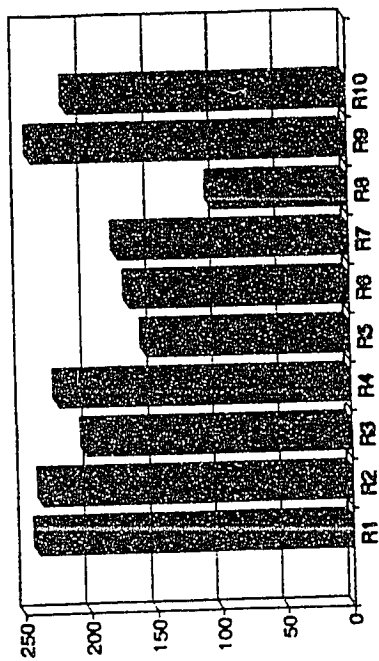
V10 Classification of Students by the Reasons for Learning the First Programming Language

CODE	REASON	COUNT
FR1	THE LANGUAGE WAS USED IN THE FIRST COURSE IN THE DEPARTMENT.	150
FR2	WANTED TO LEARN THAT LANGUAGE.	216
FR3	SOMEONE TOLD YOU TO TAKE THAT LANGUAGE COURSE	252
FR4	ONLY LANGUAGE COURSE AVAILABLE FOR YOU AT THAT TIME.	202
FR5	IT WAS A POPULAR LANGUAGE AT THAT TIME.	238
FR6	MORE JOBS AVAILABLE AT THAT TIME FOR PEOPLE WHO KNEW THE LANGUAGE.	278
FR7	THAT WAS THE ONLY LANGUAGE OFFERED BY THE SCHOOL.	216
FR8	IT WAS A REQUIRED LANGUAGE FOR THE MAJOR.	178
FR9	IT WAS THE EASY LANGUAGE.	247
FR10	OTHER	246

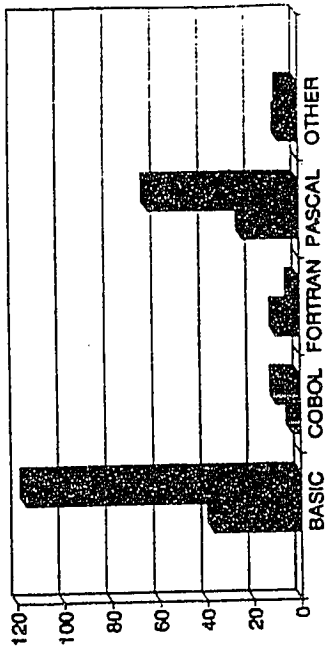
Table 11

V11 Classification of Students by the Second Programming Language Learned

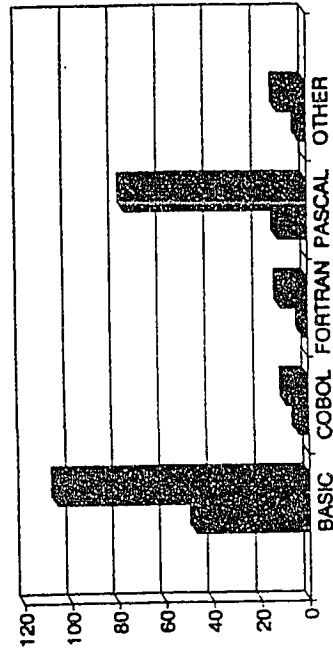
SECOND LANGUAGE	COUNT
ADA	0
APL	1
ASSEMBLY/MACHINE	17
BASIC	19
C	41
C++	4
COBOL	55
SCHEME	2
FORTH	0
FORTRAN	11
HYPERTALK	0
LISP	1
LOGO	2
MODULA-2	5
PASCAL	130
PL/I	1
PROLOG	0
SMALLTALK	0
TURING	2
OTHER	11
TOTAL	302



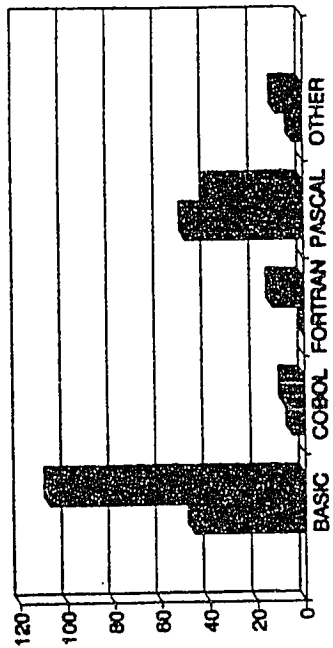
Class. of Students by Reasons For Learning Chioiced First Language
Figure 33



Cross Tabulation of School Type and First Programming Language Used
Figure 34



Cross Tabulation of School Type by First Programming Language Used
Figure 35



Cross Tabulation of School Type by First Programming Language Used
Figure 36

Table 12

V12 Classification of The Students by The Number of Years Since They Learned Their Second Programming Language

YEARS	COUNT
0	1
1	142
2	49
3	42
4	29
5	11
6	4
7	7
8	2
9	3
10	1
11	2
12	1
13	3
14	2
15	1
16	0
17	0
18	1
19	0
20	1
Total	302

Table 13

V13 Classification of the Students by the Reasons for Learning the Second Programming Language

CODE	REASON	COUNT
R1	LANGUAGE WAS USED IN THE SECOND COURSE IN DEPARTMENT.	147
R2	WANTED TO LEARN THAT LANGUAGE.	156
R3	SOMEONE TOLD YOU TO TAKE THAT LANGUAGE COURSE.	190
R4	ONLY LANGUAGE COURSE AVAILABLE FOR YOU AT THAT TIME.	186
R5	IT WAS POPULAR LANGUAGE AT THAT TIME.	195
R6	LOTS OF JOBS AVAILABLE FOR THOSE WHO KNEW THAT LANGUAGE.	220
R7	THAT WAS THE ONLY LANGUAGE OFFERED BY THE SCHOOL.	218
R8	IT WAS A REQUIRED LANGUAGE FOR THE MAJOR.	109
R9	IT WAS THE EASY LANGUAGE.	223
R10	OTHER	212

Table 14

V14 Classification of Students by Their Choice of a First Programming Language If They Were Given Another Chance to Learn a Language

LANGUAGE	COUNT
ADA	0
APL	0
ASSEMBLY/MACHINE	4
BASIC	57
C	64
C++	19
COBOL	22
SCHEME	0
FORTH	0
FORTRAN	5
HYPERTALK	1
LISP	0
LOGO	1
MODULA-2	3
PASCAL	98
PL/I	0
PROLOG	0
SMALLTALK	0
TURING	1
OTHER	18
TOTAL	293

MISSING CASES = 9

Table 15

V15 Classification of Students by the Reasons for Learning this Language as the First Programming Language

CODE	REASON	COUNT
R1	LANGUAGE WAS USED IN MOST COMPUTER SCIENCE COURSES IN THE DEPARTMENT	238
R2	WANTED TO LEARN THAT LANGUAGE	235
R3	SOMEONE TOLD YOU TO TAKE THAT LANGUAGE COURSE	202
R4	ONLY LANGUAGE COURSE AVAILABLE FOR YOU AT THAT TIME	222
R5	IT WAS POPULAR LANGUAGE AT THAT TIME	155
R6	LOTS OF JOBS AVAILABLE FOR THOSE WHO KNEW THAT LANGUAGE	167
R7	THAT WAS THE ONLY LANGUAGE OFFERED BY THE SCHOOL	175
R8	IT WAS A REQUIRED LANGUAGE FOR THE MAJOR	103
R9	IT WAS THE EASY LANGUAGE	240
R10	OTHER	212

Crosstabulations by Demographic Variables

To determine if any significant differences exist between demographic groups within the survey population of students, cross tabulations were obtained using as independent variables the results of Questions 1 to 10 and type of school (public/private), school code (4-year college/university), and status of school (minority/non-minority).

There were six dependent variables available. Among these variables four of them were: the first programming language learned; reasons for learning this language; the second programming language learned; the reasons for learning this language. The other two dependent variables were the responses to Question 10, which measured the respondent's choice of a new first programming language if they were given another chance to learn a first programming language and the reasons behind the selection of this language as their first programming language.

As the population was broken into subgroups, the frequencies in some individual cells were so small that it became necessary to collapse categories. The categories in first programming languages were collapsed into five, namely, BASIC, COBOL, FORTRAN, PASCAL and OTHER languages.

School Type and Student's Choice of a First Programming Language

The purpose of this analysis was to determine if the type of a school (public/private, 4-year college/university, minority/non-minority) was a predictor of the selection of a first programming language used in first programming language classes. As shown in Tables 16, 17, and 18, type of a school could not be a predictor of a first programming language used in the first programming language classes. Chi square value for public/private type of school was 20.853, 4-year college/university type of school was 24.698, and minority/non-minority type of school was 9.922. Same data is graphically depicted in Figures 34, 35, and 36 respectively.

Table 16
Cross Tabulation of
School Type and First Programming Language Learned

SCHOOL TYPE	BASIC	COBOL	FORTRAN	PASCAL	OTHER	ALL
PUBLIC	37	3	10	24	8	82
PRIVATE	116	10	3	64	7	200
ALL	153	13	13	88	15	282

CHI SQUARE = 20.853

MISSING CASES = 3

Table 17
Cross Tabulation of
School Type And First Programming Language Learned

SCHOOL TYPE	BASIC	COBOL	FORTRAN	PASCAL	OTHER	ALL
4-YEAR COLLEGE	46	5	0	49	4	104
UNIVERSITY	107	8	13	39	11	178
ALL	153	13	13	88	15	282

CHI SQUARE = 24.698

MISSING CASES = 3

Table 18
Cross Tabulation of
School Type and First Programming Language Learned

SCHOOL TYPE	BASIC	COBOL	FORTRAN	PASCAL	OTHER	ALL
MINORITY	47	4	2	12	3	68
NON-MINORITY	106	9	11	76	12	214
ALL	153	13	13	88	15	282

CHI SQUARE = 9.922

MISSING CASES = 3

School Type and the Reasons for Selecting a First Programming Language

The purpose of this analysis was to determine if the type of a school was a predictor of a reason for selecting a particular programming language. There were ten different reasons given in the original student questionnaire.

In this first group, school type (public/private) was cross tabulated against each of the reasons listed in the selection process of a first programming language. Only reason that the language was required for the major had no statistical significance over the type of school when making the selection of a first programming language. All the other reasons listed were significantly correlated with the type of school when making the selection of a first programming language. Of course, for some of these reasons, it was difficult to say whether the correlation had any meaning. Since some of the cell values were too small to compute a meaningful Chi square value. For example, the reason that someone told to learn the language, the reason that it was a popular language, the reason that it was an easy language, and the reason that there was a good job market for the language, the Chi square values showed that there might be a significant relationship between these reasons and the type of school when making the selection of a first programming language.

Surprisingly, Table 24 shows that more than 97% of the students surveyed learned the first programming language not because of the job market demand for that language. This percentage is 84% for public schools and 74% for private schools. Chi square value is equal to 0.066.

Table 22 shows that the reason 34% of the students took the first programming language class because that was the only language available while for private school students this percentage was 26%.

Table 19

Cross Tabulation of
School Type and the Language Was Used in the First Programming
Language Class as a Reason in Learning the First Programming Language

SCHOOL TYPE	NO	YES	ALL
PUBLIC	50	32	82
PRIVATE	100	100	200
ALL	150	132	282

CHI SQUARE = 2.814

MISSING CASES = 3

Table 20

Cross Tabulation of
School Type and Wanted to Learn the Language as a Reason in Learning
the First Programming Language

SCHOOL TYPE	NO	YES	ALL
PUBLIC	69	13	82
PRIVATE	147	53	200
ALL	216	66	282

CHI SQUARE = 3.677

MISSING CASES = 3

Table 21

Cross Tabulation of
School Type and Someone Told to Learn the Language as a Reason in
Learning the First Programming Language

SCHOOL TYPE	NO	YES	ALL
PUBLIC	74	8	82
PRIVATE	178	22	200
ALL	252	30	282

CHI SQUARE = 0.095

MISSING CASES = 3

Table 22

Cross Tabulation of
School Type and Only Language Available as a Reason in Learning the First
Programming Language

SCHOOL TYPE	NO	YES	ALL
PUBLIC	54	28	82
PRIVATE	148	52	200
ALL	202	80	282

CHI SQUARE = 1.899

MISSING CASES = 3

Table 23

Cross Tabulation of
School Type and the Language Was Popular as a Reason in Learning the
First Programming Language

SCHOOL TYPE	NO	YES	ALL
PUBLIC	69	13	82
PRIVATE	169	31	200
ALL	238	44	282

CHI SQUARE = 0.006

MISSING CASES = 3

Table 24

Cross Tabulation of
School Type and Job Market for the Language as a Reason in Learning the
First Programming Language

SCHOOL TYPE	NO	YES	ALL
PUBLIC	80	2	82
PRIVATE	194	6	200
ALL	274	8	282

CHI SQUARE = 0.066

MISSING CASES = 3

Table 25

Cross Tabulation of
School Type and Only Language Was Offered by the School as a Reason in
Learning the First Programming Language

SCHOOL TYPE	NO	YES	ALL
PUBLIC	62	20	82
PRIVATE	154	46	200
ALL	216	66	282

CHI SQUARE = 0.063

MISSING CASES = 3

Table 26

Cross Tabulation of
School Type and the Language Was Required for the Major as a Reason in
Learning the First Programming Language

SCHOOL TYPE	NO	YES	ALL
PUBLIC	64	18	82
PRIVATE	114	86	200
ALL	178	104	282

CHI SQUARE = 11.069

MISSING CASES = 3

Table 27

Cross Tabulation of
School Type and it Was an Easy Language as a Reason in Selecting the
First Programming Language

SCHOOL TYPE	NO	YES	ALL
PUBLIC	72	10	82
PRIVATE	175	25	200
ALL	247	35	282

CHI SQUARE = 0.005

MISSING CASES = 3

In this second group, school type (4-year college/University) was cross tabulated against each of the reasons listed in the selection process of a first programming language. All the reasons listed were significantly correlated with the type of school when making the selection of a first programming language. Of course, for some of these reasons, it was difficult to say whether the correlation had any meaning. Since some of the cell values were too small to compute a meaningful chi square value. For example, the reason that someone told to learn the language, the reason that it was a popular language, the reason that it was an easy language, and the reason that there was a good job market for the language, the Chi square values show that there might be a significant relationship between these reasons and the type of school when making the selection of a first programming language.

Surprisingly, Table 33 shows that more than 97% of the students surveyed learned the first programming language not because of the job market demand for that language. Chi square value is equal to 0.001.

Table 28 shows that the reason 51% of the students from four-year colleges took the first programming language class because that was used in a first programming language class while for university students this percentage was 44%. The chi square value was 1.141.

Table 31 shows that the reason 30% of the students from four-year colleges took the first programming language class because that was the only programming language available while for university students this percentage was 28%. The chi square value was 0.168.

Table 29 shows that the reason only 19% of the students from four-year colleges took the first programming language class because they wanted to learn the programming language while for university students this percentage was 26%. The Chi square value was 1.601.

Table 28

Cross Tabulation of
School Type And The Language Was Used in The First Programming
Language Class as a Reason in Learning The First Programming Language

SCHOOL TYPE	NO	YES	ALL
4-YEAR COLLEGE	51	53	104
UNIVERSITY	99	79	178
ALL	150	132	282

CHI SQUARE = 1.141

MISSING CASES = 3

Table 29

Cross Tabulation of
School Type And Wanted to Learn The Language as a Reason in Learning
The First Programming Language

SCHOOL TYPE	NO	YES	ALL
4-YEAR COLLEGE	84	20	104
UNIVERSITY	132	46	178
ALL	216	66	282

CHI SQUARE = 1.601

MISSING CASES = 3

Table 30

Cross Tabulation of
School Type And Someone Told to Learn The Language as a Reason in
Learning The First Programming Language

SCHOOL TYPE	NO	YES	ALL
4-YEAR COLLEGE	93	11	104
UNIVERSITY	159	19	178
ALL	252	30	282

CHI SQUARE = 0.001

MISSING CASES = 3

Table 31

Cross Tabulation of
School Type And Only Language Available as a Reason in Learning The
First Programming Language

SCHOOL TYPE	NO	YES	ALL
4-YEAR COLLEGE	73	31	104
UNIVERSITY	129	49	178
ALL	202	80	282

CHI SQUARE = 0.168

MISSING CASES = 3

Table 32

Cross Tabulation of
School Type And The Language Was Popular as a Reason in Learning The
First Programming Language

SCHOOL TYPE	NO	YES	ALL
4-YEAR COLLEGE	86	18	104
UNIVERSITY	152	26	178
ALL	238	44	282

CHI SQUARE = 0.364

MISSING CASES = 3

Table 33

Cross Tabulation of
School Type And Job Market For The Language as a Reason in Learning
The First Programming Language

SCHOOL TYPE	NO	YES	ALL
4-YEAR COLLEGE	101	3	104
UNIVERSITY	173	5	178
ALL	274	8	282

CHI SQUARE = 0.001

MISSING CASES = 3

Table 34

Cross Tabulation of
School Type And Only Language Was Offered by The School as a Reason
in Learning The First Programming Language

SCHOOL TYPE	NO	YES	ALL
4-YEAR COLLEGE	79	25	104
UNIVERSITY	137	41	178
ALL	216	66	282

CHI SQUARE = 0.037

MISSING CASES = 3

Table 35

Cross Tabulation of
School Type And The Language Was Required For The Major as a Reason
in Learning The First Programming Language

SCHOOL TYPE	NO	YES	ALL
4-YEAR COLLEGE	66	38	104
UNIVERSITY	112	66	178
ALL	178	104	282

CHI SQUARE = 0.008

MISSING CASES = 3

Table 36

Cross Tabulation of
School Type And it Was an Easy Language as a Reason in Selecting The
First Programming Language

SCHOOL TYPE	NO	YES	ALL
4-YEAR COLLEGE	90	14	104
UNIVERSITY	157	21	178
ALL	247	35	282

CHI SQUARE = 0.167

MISSING CASES = 3

In this third group, school type (Minority/Non-minority) was cross tabulated against each of the reasons listed in the selection process of a first programming language. All but two of the reasons listed were significantly correlated with the type of school when making the selection of a first programming language. The reasons that the language was required for the major and that they wanted to learn the language as a reason for selecting a first programming language had no statistical significance over the type of school. Of course, for some of these reasons, it was difficult to say whether the correlation had any meaning. Since some of the cell values were too small to compute a meaningful Chi square value. For example, the reason that someone told to learn the language, the reason that it was a popular language, the reason that it was an easy language, and the reason that there was a good job market for the language, the Chi square values showed that there might be a significant relationship between these reasons and the type of school when making the selection of a first programming language. To see that if this significance was real or not, it had to be studied further for some more analysis.

Table 37 shows that the reason 51% of the students from minority schools took the first programming language class because that was used in a first programming language class while for non-minority students this percentage was 45%. The Chi square value was 0.782.

Table 40 shows that the reason 28% of the students from minority schools took the first programming language class because that was the only programming language available while for non-minority students this percentage was 29%. The Chi square value was 0.008.

Table 43 shows that the reason only 15% of the students from minority schools took the first programming language class because that was the only language offered by the school while for non-minority school students this percentage was 26%. The Chi square value was 3.782.

Table 37

Cross Tabulation of
School Type And The Language Was Used in The First Programming
Language Class as a Reason in Learning The First Programming Language

SCHOOL TYPE	NO	YES	ALL
MINORITY	33	35	68
NON-MINORITY	117	97	214
ALL	150	132	282

CHI SQUARE = 0.782

MISSING CASES = 3

Table 38

Cross Tabulation of
School Type And Wanted to Learn The Language as a Reason in Learning
The First Programming Language

SCHOOL TYPE	NO	YES	ALL
MINORITY	40	28	68
NON-MINORITY	176	38	214
ALL	216	66	282

CHI SQUARE = 15.788

MISSING CASES = 3

Table 39

Cross Tabulation of
School Type And Someone Told to Learn The Language as a Reason in
Learning The First Programming Language

SCHOOL TYPE	NO	YES	ALL
MINORITY	58	10	68
NON-MINORITY	194	20	214
ALL	252	30	282

CHI SQUARE = 1.560

MISSING CASES = 3

Table 40

Cross Tabulation of
School Type And Only Language Available as a Reason in Learning The
First Programming Language

SCHOOL TYPE	NO	YES	ALL
MINORITY	49	19	68
NON-MINORITY	153	61	214
ALL	202	80	282

CHI SQUARE = 0.008

MISSING CASES = 3

Table 41

Cross Tabulation of
School Type And The Language Was Popular as a Reason in Learning The
First Programming Language

SCHOOL TYPE	NO	YES	ALL
MINORITY	56	12	68
NON-MINORITY	182	32	214
ALL	238	44	282

CHI SQUARE = 0.284

MISSING CASES = 3

Table 42

Cross Tabulation of
School Type And Job Market For The Language as a Reason in Learning
The First Programming Language

SCHOOL TYPE	NO	YES	ALL
MINORITY	65	3	68
NON-MINORITY	209	5	214
ALL	274	8	282

CHI SQUARE = 0.806

MISSING CASES = 3

Table 43

Cross Tabulation of
School Type And Only Language Was Offered by The School as a Reason
in Learning The First Programming Language

SCHOOL TYPE	NO	YES	ALL
MINORITY	58	10	68
NON-MINORITY	158	56	214
ALL	216	66	282

CHI SQUARE = 3.782

MISSING CASES = 3

Table 44

Cross Tabulation of
School Type And The Language Was Required For The Major as a Reason
in Learning The First Programming Language

SCHOOL TYPE	NO	YES	ALL
MINORITY	32	36	68
NON-MINORITY	146	68	214
ALL	178	104	282

CHI SQUARE = 9.931

MISSING CASES = 3

Table 45

Cross Tabulation of
School Type And it Was an Easy Language as a Reason in Selecting The
First Programming Language

SCHOOL TYPE	NO	YES	ALL
MINORITY	58	10	68
NON-MINORITY	189	26	214
ALL	247	36	282

CHI SQUARE = 0.434

MISSING CASES = 3

School Type and Student's Choice of a New First Programming Language

The purpose of this analysis was to determine if the type of a school (public/private, 4-year college/university, minority/non-minority) was a predictor of a new programming language a student would choose if they were given another chance. As the population was broken into subgroups, the frequencies in some individual cells were so small that it became necessary to collapse categories. The categories in first programming languages were collapsed into five, namely, BASIC, C, COBOL, PASCAL and OTHER languages.

As shown in Tables 46 and 48, school type (public/private) and school type (minority/non-minority) could not be a predictor of selection of a new first programming language. Chi squares for selecting a new first programming language were 15.554 (public/private) and 15.058 (minority/non-minority).

But, surprisingly, when school type (4-year college/university) was considered, we saw a strong positive correlation as shown in Table 47 where the chi square value was 6.404. More students (45%) from four-year colleges as compared to only 30% of the students from universities were going with PASCAL as their new first programming language. For BASIC this ratio was 22% to 24%, for C it was 21% to 28%, for COBOL it was 5% to 8% and 7% of the four-year college students and 10% of university students were going to choose some other languages as their new first programming language.

Table 46

**Cross Tabulation of
School Type And New First Programming Language Learned**

SCHOOL TYPE	BASIC	C	COBOL	PASCAL	OTHER	ALL
PUBLIC	8	28	6	30	4	76
PRIVATE	49	34	11	56	17	167
ALL	57	62	17	86	21	243

CHI SQUARE = 15.554

MISSING CASES = 39

Table 47

**Cross Tabulation of
School Type And New First Programming Language Learned**

SCHOOL TYPE	BASIC	C	COBOL	PASCAL	OTHER	ALL
4-YEAR COLLEGE	19	18	4	39	6	86
UNIVERSITY	38	44	13	47	15	157
ALL	57	62	17	86	21	243

CHI SQUARE = 6.404

MISSING CASES = 39

Table 48

**Cross Tabulation of
School Type And New First Programming Language Learned**

SCHOOL TYPE	BASIC	C	COBOL	PASCAL	OTHER	ALL
MINORITY	24	12	5	14	8	63
NON-MINORITY	33	50	12	72	13	180
ALL	57	62	17	86	21	243

CHI SQUARE = 15.058

MISSING CASES = 39

School Type and the Reasons for Selecting a New First Programming Language if Students Were Given Another Chance for Selecting a First Programming Language

The purpose of this analysis was to determine if the type of a school was a predictor of a reason for selecting a new first programming language if the students were given another chance of learning a first programming language. There were ten different reasons given in the original student questionnaire.

In this first group, school type (public/private) was crossstabulated against each of the reasons listed in the selection process of a first programming language. Out of these ten reasons only three of the following reasons seemed to have positive correlation. These reasons were : The language was very popular (Table 53, Chi square value = 0.019), it was the only language offered by the school (Table 55, Chi square value = 3.732) and the language was required for the major (Table 56, Chi square value = 3.610). All the other reasons (Table 51, 52, 54) either did not show any significant relationship. Of Course, for some of these reasons, it was difficult to say whether the correlation had any meaning. Since some of the cell values were too small to compute a meaningful chi square value. For example, the reason that the language was used in the second programming language class (Table 49), the reason that the participant wanted to learn the language (Table 50), and the reason (Table 57) that the reason that it was an easy language, the Chi square values showed that there might be a significant relationship between these reasons and the type of

school when making the selection of a first programming language. Table 54 showed that job market was not one of those reasons where there was any correlation between the type of a school and the choice of a new first programming language.

Table 49

Cross Tabulation of
School Type And The Language Was Used in The Second Programming
Language Class as a Reason in Learning The New First Programming
Language

SCHOOL TYPE	NO	YES	ALL
PUBLIC	73	3	76
PRIVATE	165	2	167
ALL	238	5	243

CHI SQUARE = 1.960

MISSING CASES = 39

Table 50

Cross Tabulation of
School Type And Wanted to Learn The Language as a Reason in Learning
The New First Programming Language

SCHOOL TYPE	NO	YES	ALL
PUBLIC	74	2	76
PRIVATE	161	6	167
ALL	235	8	243

CHI SQUARE = 0.152

MISSING CASES = 39

Table 51

Cross Tabulation of
School Type And Someone Told to Learn The Language as a Reason in
Learning The New First Programming Language

SCHOOL TYPE	NO	YES	ALL
PUBLIC	56	20	76
PRIVATE	146	21	167
ALL	202	41	243

CHI SQUARE = 7.031

MISSING CASES = 39

Table 52

**Cross Tabulation of
School Type And Only Language Available as a Reason in Learning The
New First Programming Language**

SCHOOL TYPE	NO	YES	ALL
PUBLIC	65	11	76
PRIVATE	157	10	167
ALL	222	21	243

CHI SQUARE = 4.764

MISSING CASES = 39

Table 53

**Cross Tabulation of
School Type And The Language Was Popular as a Reason in Learning The
New First Programming Language**

SCHOOL TYPE	NO	YES	ALL
PUBLIC	48	28	76
PRIVATE	107	60	167
ALL	155	88	243

CHI SQUARE = 0.019

MISSING CASES = 39

Table 54

**Cross Tabulation of
School Type And Job Market For The Language as a Reason in Learning
The New First Programming Language**

SCHOOL TYPE	NO	YES	ALL
PUBLIC	45	31	76
PRIVATE	122	45	167
ALL	167	76	243

CHI SQUARE = 4.657

MISSING CASES = 39

Table 55

Cross Tabulation of
School Type And Only Language Was Offered by The School as a Reason in
Learning The New First Programming Language

SCHOOL TYPE	NO	YES	ALL
PUBLIC	61	15	76
PRIVATE	114	53	167
ALL	175	68	243

CHI SQUARE = 3.732

MISSING CASES = 39

Table 56

Cross Tabulation of
School Type And The Language Was Required For The Major as a Reason
in Learning the New First Programming Language

SCHOOL TYPE	NO	YES	ALL
PUBLIC	39	37	76
PRIVATE	64	103	167
ALL	103	140	243

CHI SQUARE = 3.610

MISSING CASES = 39

Table 57

Cross Tabulation of
School Type And it Was an Easy Language as a Reason in Selecting the New
First Programming Language

SCHOOL TYPE	NO	YES	ALL
PUBLIC	74	2	76
PRIVATE	166	1	167
ALL	240	3	243

CHI SQUARE = 1.770

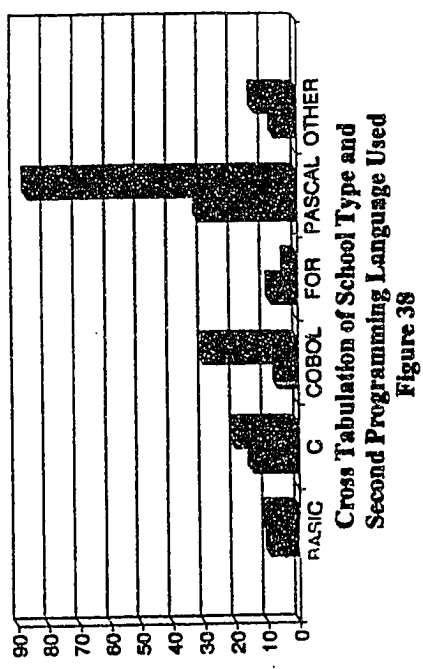
MISSING CASES = 39

In this second group, school type (4-year college/University) was cross tabulated against each of the reasons listed in the selection process of a new first programming language if students were given another chance of selecting a new first programming language. Out of the ten reasons listed in the original questionnaire, the following reasons showed some positive correlation: the reason that the language was popular (Table 62, Chi square value = 0.268), and the reason that the language had a very good job market (Table 63, Chi square value = 0.067). All the other reasons listed did not show any significant correlation with the type of school when making the selection of a new first programming language. Of course, for some of these reasons, it was difficult to say whether the correlation had any meaning. Since some of the cell values were too small to compute a meaningful Chi square value. For example, the reason that it was used in the first programming language class (Table 58), the reason that the participant wanted to learn the language (Table 59), the reason that it was the only available language (Table 61), and the reason that it was an easy language (Table 66), the Chi square values showed that there might be a significant relationship between these reasons and the type of school when making the selection of a first programming language. But being the contents of some of the cells too small to come up with a conclusion. Hence, one can say that these reasons needed to be considered for further analysis.

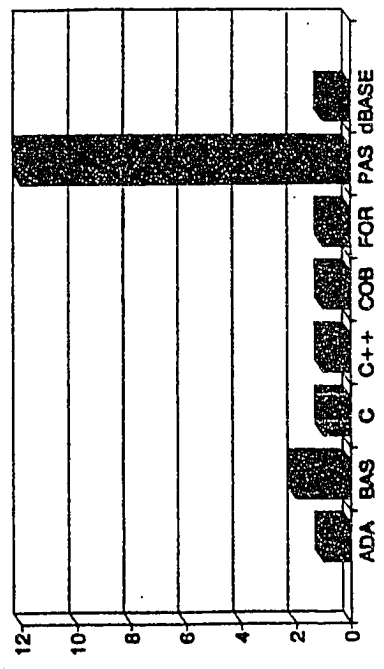
Surprisingly, Table 65 showed that no more than 58% of the students surveyed wanted to learn this new first programming language because it was required in the major. Chi square value was equal to 4.093.

Table 63 shows that the only 31% of the students wanted to learn a new first programming language because of the job market demand for this new first programming language. The Chi square value was 0.067.

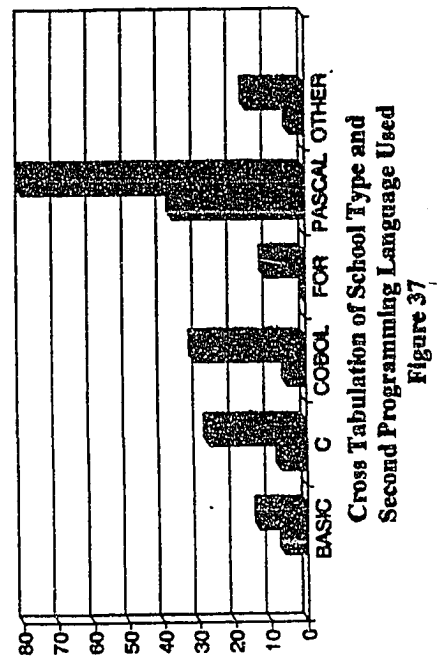
Table 64, and 65 show that school type (four-year college/university) was not correlated with the reasons that it was the only language offered by the school and the language was required for the major in selecting the first programming language.



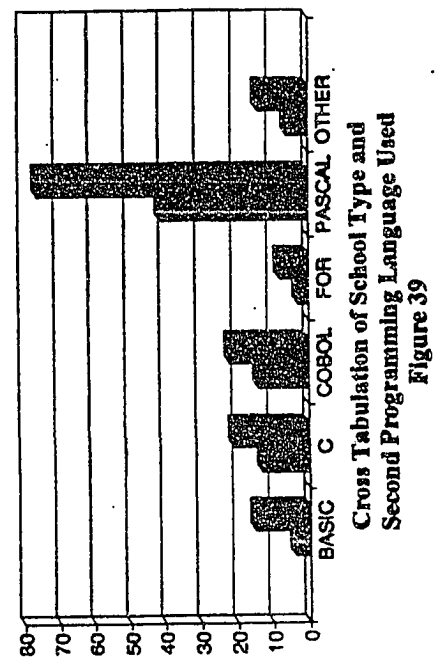
Cross Tabulation of School Type and Second Programming Language Used
Figure 38



Classification of Faculty by the First Programming Language
Figure 40



Cross Tabulation of School Type and Second Programming Language Used
Figure 37



Cross Tabulation of School Type and Second Programming Language Used
Figure 39

Table 58

Cross Tabulation of
School Type And The Language Was Used in the Second Programming
Language Class as a Reason in Learning the New First Programming Language

SCHOOL TYPE	NO	YES	ALL
4-YEAR COLLEGE	84	2	86
UNIVERSITY	154	3	157
ALL	238	5	243

CHI SQUARE = 0.047

MISSING CASES = 39

Table 59

Cross Tabulation of
School Type And Wanted to Learn the Language as a Reason in Learning the
New First Programming Language

SCHOOL TYPE	NO	YES	ALL
4-YEAR COLLEGE	86	0	86
UNIVERSITY	149	8	157
ALL	235	8	243

CHI SQUARE = 4.531

MISSING CASES = 39

Table 60

Cross Tabulation of
School Type And Someone Told to Learn The Language as a Reason in
Learning the New First Programming Language

SCHOOL TYPE	NO	YES	ALL
4-YEAR COLLEGE	81	5	86
UNIVERSITY	121	36	157
ALL	202	41	243

CHI SQUARE = 11.606

MISSING CASES = 39

Table 61

Cross Tabulation of
School Type And Only Language Available as a Reason in Learning The New
First Programming Language

SCHOOL TYPE	NO	YES	ALL
4-YEAR COLLEGE	78	8	86
UNIVERSITY	144	13	157
ALL	222	21	243

CHI SQUARE = 0.074

MISSING CASES = 39

Table 62

Cross Tabulation of
School Type And The Language Was Popular as a Reason in Learning The
New First Programming Language

SCHOOL TYPE	NO	YES	ALL
4-YEAR COLLEGE	53	33	86
UNIVERSITY	102	55	157
ALL	155	88	243

CHI SQUARE = 0.268

MISSING CASES = 39

Table 63

Cross Tabulation of
School Type And Job Market For The Language as a Reason in Learning
The New First Programming Language

SCHOOL TYPE	NO	YES	ALL
4-YEAR COLLEGE	60	26	86
UNIVERSITY	107	50	157
ALL	167	76	243

CHI SQUARE = 0.067

MISSING CASES = 39

Table 64

**Cross Tabulation of
School Type And Only Language Was Offered by The School as a Reason
in Learning The New First Programming Language**

SCHOOL TYPE	NO	YES	ALL
4-YEAR COLLEGE	53	33	86
UNIVERSITY	122	35	157
ALL	175	68	243

CHI SQUARE = 7.128

MISSING CASES = 39

Table 65

**Cross Tabulation of
School Type And The Language Was Required For The Major as a Reason
in Learning The New First Programming Language**

SCHOOL TYPE	NO	YES	ALL
4-YEAR COLLEGE	29	57	86
UNIVERSITY	74	83	157
ALL	103	140	243

CHI SQUARE = 4.093

MISSING CASES = 39

Table 66

**Cross Tabulation of
School Type And it Was an Easy Language as a Reason in Selecting The
New First Programming Language**

SCHOOL TYPE	NO	YES	ALL
4-YEAR COLLEGE	86	0	86
UNIVERSITY	154	3	157
ALL	240	3	243

CHI SQUARE = 1.664

MISSING CASES = 39

In this third group, school type (Minority/Non-minority) was cross tabulated against each of the reasons listed in the selection process of a new first programming language. All but one of the reasons listed did not show any significant correlation with the type of school when making the selection of a new first programming language if the students were given another chance of selecting a new first programming language.

Table 71, 72, and 74 show out of the ten reasons listed on the questionnaire only three of them show a significant positive correlation. The reason that it was a popular language was the reason in selecting the first programming language and the Chi square value = 0.062. The reasons that there was a great job market demand for the language and the reason that it was required for the major were the other two reasons for selecting a new first programming language which show some statistical significance over the type of school. Of course, for some of the other reasons, it was difficult to say whether the correlation had any meaning. Since some of the cell values were too small to compute a meaningful Chi square value. For example, the reason that someone told to learn the language, the reason that they wanted to learn the language, the reason that it was an easy language, the reason that the language was used in the second programming language class, and the reason that it was the only language available, the Chi square values show that there might be a

significant relationship between these reasons and the type of school when making the selection of a first programming language. To see that if this significance was real or not, it had to be studied further for some more analysis.

Table 73 shows that it was the only language offered by the school and the school type did not show any significant relationship in selecting this new first programming language. The chi square value was 6.190.

Table 67

**Cross Tabulation of
School Type And The Language Was Used in The Second Programming
Language Class as a Reason in Learning The New First Programming
Language**

SCHOOL TYPE	NO	YES	ALL
MINORITY	63	0	63
NON-MINORITY	175	5	180
ALL	238	5	243

CHI SQUARE = 1.787

MISSING CASES = 39

Table 68

**Cross Tabulation of
School Type And Wanted to Learn The Language as a Reason in Learning
The New First Programming Language**

SCHOOL TYPE	NO	YES	ALL
MINORITY	57	6	63
NON-MINORITY	178	2	180
ALL	235	8	243

CHI SQUARE = 10.374

MISSING CASES = 39

Table 69

**Cross Tabulation of
School Type And Someone Told to Learn The Language as a Reason in
Learning The New First Programming Language**

SCHOOL TYPE	NO	YES	ALL
MINORITY	55	8	63
NON-MINORITY	147	33	180
ALL	202	41	243

CHI SQUARE = 1.056

MISSING CASES = 39

Table 70

Cross Tabulation of
School Type And Only Language Available as a Reason in Learning The
New First Programming Language

SCHOOL TYPE	NO	YES	ALL
MINORITY	63	0	63
NON-MINORITY	159	21	180
ALL	222	21	243

CHI SQUARE = 8.045

MISSING CASES = 39

Table 71

Cross Tabulation of
School Type And The Language Was Popular as a Reason in Learning The
New First Programming Language

SCHOOL TYPE	NO	YES	ALL
MINORITY	41	22	63
NON-MINORITY	114	66	180
ALL	155	88	243

CHI SQUARE = 0.062

MISSING CASES = 39

Table 72

Cross Tabulation of
School Type And Job Market For The Language as a Reason in Learning
The New First Programming Language

SCHOOL TYPE	NO	YES	ALL
MINORITY	48	17	63
NON-MINORITY	121	59	180
ALL	167	76	243

CHI SQUARE = 0.729

MISSING CASES = 39

Table 73

Cross Tabulation of
School Type And Only Language Was Offered by The School as a Reason
in Learning The New First Programming Language

SCHOOL TYPE	NO	YES	ALL
MINORITY	53	10	63
NON-MINORITY	122	58	180
ALL	175	68	243

CHI SQUARE = 6.190

MISSING CASES = 39

Table 74

Cross Tabulation of
School Type And The Language Was Required For The Major as a Reason
in Learning The New First Programming Language

SCHOOL TYPE	NO	YES	ALL
MINORITY	29	34	63
NON-MINORITY	74	106	180
ALL	103	140	243

CHI SQUARE = 0.463

MISSING CASES = 39

Table 75

Cross Tabulation of
School Type And it Was an Easy Language as a Reason in Selecting The
New First Programming Language

SCHOOL TYPE	NO	YES	ALL
MINORITY	63	0	63
NON-MINORITY	177	3	180
ALL	240	3	243

CHI SQUARE = 1.063

MISSING CASES = 39

School Type and Student's Choice of a Second Programming Language

The purpose of this analysis was to determine if the type of a school (public/private, 4-year college/university, minority/non-minority) was a predictor of a second programming language choice. As the population was broken into subgroups, the frequencies in some individual cells were so small that it became necessary to collapse categories. The categories in second programming languages were collapsed into six, namely, BASIC, C, COBOL, FORTRAN, PASCAL and other languages.

As shown in Tables 76 and 77 and Figures 37, 38, school type (public/private) and school type (minority/non-minority) cannot be a predictor of selection of a second programming language. Chi square value for selecting a second programming language is 17.454 when school type is public/private whereas the Chi square value changes to 9.860 when school type considered is minority/non-minority. But, surprisingly, when school type (4-year college / university) is considered, we see a strong positive correlation as shown in Table 77 where the chi square value is 2.486. Same data is graphically depicted in Figure 39. More students (63%) from four-year colleges as compared to only 45% of the students from universities were going with PASCAL as their second programming language.

For BASIC this ratio is 10% to 7%, for C it is 12% to 15%, for COBOL it is 8% to 17%, for FORTRAN this ratio is 0% to 6% and 7% of the four-year college students and 9% of university students were using some other languages in their second programming language class.

Table 76

Cross Tabulation of
School Type And Second Programming Language Learned

SCHOOL TYPE	BASIC	C	COBOL	FORTRAN	PASCAL	OTHER	ALL
PUBLIC	9	14	6	8	31	7	75
PRIVATE	10	20	30	3	86	13	162
ALL	19	34	36	11	117	20	237

CHI SQUARE = 17.454

MISSING CASES = 45

Table 77

Cross Tabulation of
School Type And Second Programming Language Learned

SCHOOL TYPE	BASIC	C	COBOL	FORTRAN	PASCAL	OTHER	ALL
4-YEAR COLLEGE	4	13	14	3	41	6	81
UNIVERSITY	15	21	22	8	76	14	156
ALL	19	34	36	11	117	20	237

CHI SQUARE = 2.486

MISSING CASES = 45

Table 78

Cross Tabulation of
School Type And Second Programming Language Learned

SCHOOL TYPE	BASIC	C	COBOL	FORTRAN	PASCAL	OTHER	ALL
MINORITY	6	7	5	0	37	4	59
NON-MINORITY	13	27	31	11	80	16	178
ALL	19	34	36	11	117	20	237

CHI SQUARE = 9.860

MISSING CASES = 45

School Type and the Reasons For Selecting a Second Programming Language

The purpose of this analysis was to determine if the type of a school was a predictor of a reason for selecting a second programming language. There were ten different reasons given in the original student questionnaire.

In this first group, school type (public/private) was cross tabulated against each of the reasons listed in the selection process of a second programming language. Only two of the ten reasons listed in the questionnaire show positive correlation between the type of school and the selection of a second programming language. The reason that the language was used in the second programming language class did not show any significant relationship between the type of school and the selection of a second programming language: Of course, for some of these reasons, it was difficult to say whether the correlation had any meaning. Since some of the cell values were too small to compute a meaningful Chi square value. For example, the reason that there was a good job market for the language, the reason that it was the only language offered by the school, the reason that someone told students to learn that language, the reason that it was a popular language and the reason that it was an easy language, the Chi square values show that there might be a significant relationship between these reasons and the type of school when making the selection of a second programming language. But since the some of the cell values were too small to come with a concrete conclusion. These reasons were

left for further analysis. Table 80 shows that 28% of the students from public schools surveyed learned the second programming language because of their interests in learning the language. This percentage is 37% for private schools. Chi square value is equal to 1.861.

Table 81 shows that the reason 20% of the students took the second programming language class because of the advice of someone while for private school students this percentage is 23% and for private schools it is 19%. Chi square value is equal to 0.555.

Table 82 shows that 22% of the students took the second programming language class because it was the only class available at that time. This ratio is 23% for public schools to 21% for private schools. Chi square value is 0.086.

Table 86 shows that 54% of the students learned the second programming language because it was required for their major. This percentage goes down to 46% for public schools and goes up to 57% for private schools. The Chi square value is 2.381.

Table 79

Cross Tabulation of
School Type And The Language Was Used In The Second Programming
Language Class as a Reason in Learning The Second Programming
Language

SCHOOL TYPE	NO	YES	ALL
PUBLIC	54	21	75
PRIVATE	93	69	162
ALL	147	90	237

CHI SQUARE = 4.635

MISSING CASES = 45

Table 80

Cross Tabulation of
School Type And Wanted to Learn The Language as a Reason in Learning
The Second Programming Language

SCHOOL TYPE	NO	YES	ALL
PUBLIC	54	21	75
PRIVATE	102	60	162
ALL	156	81	237

CHI SQUARE = 1.861

MISSING CASES = 45

Table 81

Cross Tabulation of
School Type And Someone Told to Learn The Language as a Reason in
Learning The Second Programming Language

SCHOOL TYPE	NO	YES	ALL
PUBLIC	58	17	75
PRIVATE	132	30	162
ALL	190	47	237

CHI SQUARE = 0.555

MISSING CASES = 45

Table 82

Cross Tabulation of
School Type And Only Language Available as a Reason in Learning The
Second Programming Language

SCHOOL TYPE	NO	YES	ALL
PUBLIC	58	17	75
PRIVATE	128	34	162
ALL	186	51	237

CHI SQUARE = 0.086

MISSING CASES = 45

Table 83

Cross Tabulation of
School Type And The Language Was Popular as a Reason in Learning The
Second Programming Language

SCHOOL TYPE	NO	YES	ALL
PUBLIC	56	19	75
PRIVATE	139	23	162
ALL	195	42	237

CHI SQUARE = 4.360

MISSING CASES = 45

Table 84

Cross Tabulation of
School Type And Job Market For The Language as a Reason in Learning
The Second Programming Language

SCHOOL TYPE	NO	YES	ALL
PUBLIC	69	6	75
PRIVATE	151	11	162
ALL	220	17	237

CHI SQUARE = 0.113

MISSING CASES = 45

Table 85

Cross Tabulation of School Type And Only Language Was Offered by The School as a Reason in Learning The Second Programming Language

SCHOOL TYPE	NO	YES	ALL
PUBLIC	69	6	75
PRIVATE	149	13	162
ALL	220	19	237

CHI SQUARE = 0.000

MISSING CASES = 45

Table 86

Cross Tabulation of School Type And The Language Was Required For The Major as a Reason in Learning The Second Programming Language

SCHOOL TYPE	NO	YES	ALL
PUBLIC	40	35	75
PRIVATE	69	93	162
ALL	109	128	237

CHI SQUARE = 2.381

MISSING CASES = 45

Table 87

Cross Tabulation of School Type And it Was an Easy Language as a Reason in Selecting The Second Programming Language

SCHOOL TYPE	NO	YES	ALL
PUBLIC	72	3	75
PRIVATE	151	11	162
ALL	223	14	237

CHI SQUARE = 0.718

MISSING CASES = 45

In this second group, school type (4-year college/university) was cross tabulated against each of the reasons listed in the selection process of a second programming language. Out of the ten reasons listed in the questionnaire only three reasons listed were significantly correlated with the type of school when making the selection of a second programming language. Also, for other six reasons, it was difficult to say whether the correlation had any meaning. Since some of the cell values were too small to compute a meaningful Chi square value. For example, the reason that there was a good job market for the language, the reason that it was the only language offered by the school, the reason that someone told them to learn the language, the reason that it was a popular language, and the reason that it was an easy language, the Chi square values show that there might be a significant relationship between these reasons and the type of school when making the selection of a second programming language. But since the some of the cell values re too small to come with a concrete conclusion. These reasons were left for further analysis.

Table 88 shows that only 38% of the students surveyed learned the second programming language because it was used in the second programming language class. This percentage is 43% to 35% for four-year college students to university students. The Chi square value is = 1.432.

Table 89 shows that 36% of the students from four-year colleges surveyed learned the second programming language because of their interests in learning the language. This percentage is 34% for universities. Chi square value is equal to 0.039.

Table 90 shows that the reason 20% of the students took the second programming language class because of the advice of someone while for four-year college students this percentage is 17% and for universities it is 21%. Chi square value is equal to 0.502

Table 91 shows that 22% of the students took the second programming language class because it was the only class available at that time. This ratio is 28% for four-year colleges to 18% for universities. Chi square value is 3.445.

Table 95 shows that 54% of the students learned the second programming language because it was required for their major. This percentage goes down to 52% for four-year colleges and goes up to 55% for universities. The Chi square value is 0.230.

Table 88 shows that only 38% of the students learned the second programming language because it was used in the second programming language class. This percentage goes up to 43% for students from Four-year colleges and it goes down to 35% for university students. The Chi square value is 1.432.

Only 18% of the students learned the second programming language because of its popularity. This percentage is 19% for four-year college students and 17% for university students. Table 92 shows the Chi square value = 0.054.

Table 88

Cross Tabulation of
School Type And The Language Was Used in The Second Programming
Language Class as a Reason in Learning The Second Programming Language

SCHOOL TYPE	NO	YES	ALL
4-YEAR COLLEGE	46	35	81
UNIVERSITY	101	55	156
ALL	147	90	237

CHI SQUARE = 1.432

MISSING CASES = 45

Table 89

Cross Tabulation of
School Type And Wanted to Learn The Language as a Reason in Learning
The Second Programming Language

SCHOOL TYPE	NO	YES	ALL
4-YEAR COLLEGE	54	27	75
UNIVERSITY	102	54	156
ALL	156	81	237

CHI SQUARE = 0.039

MISSING CASES = 45

Table 90

Cross Tabulation of
School Type And Someone Told to Learn The Language as a Reason in
Learning The Second Programming Language

SCHOOL TYPE	NO	YES	ALL
4-YEAR COLLEGE	67	14	81
UNIVERSITY	123	33	156
ALL	190	47	237

CHI SQUARE = 0.502

MISSING CASES = 45

Table 91

**Cross Tabulation of
School Type And Only Language Available as a Reason in Learning The
Second Programming Language**

SCHOOL TYPE	NO	YES	ALL
4-YEAR COLLEGE	58	23	81
UNIVERSITY	128	28	156
ALL	186	51	237

CHI SQUARE = 3.445

MISSING CASES = 45

Table 92

**Cross Tabulation of
School Type And The Language Was Popular as a Reason in Learning The
Second Programming Language**

SCHOOL TYPE	NO	YES	ALL
4-YEAR COLLEGE	66	15	81
UNIVERSITY	129	27	156
ALL	195	42	237

CHI SQUARE = 0.054

MISSING CASES = 45

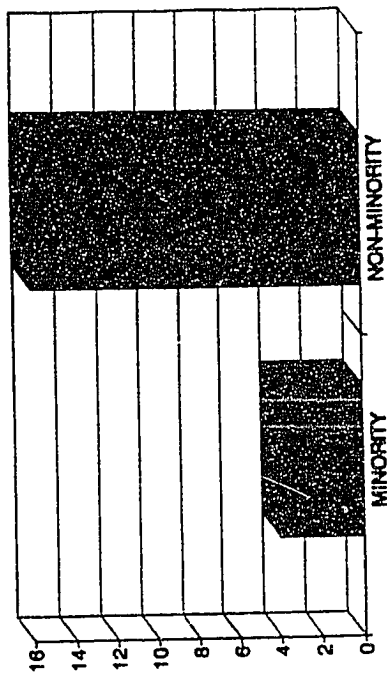
Table 93

**Cross Tabulation of
School Type And Job Market For The Language as a Reason in Learning
The Second Programming Language**

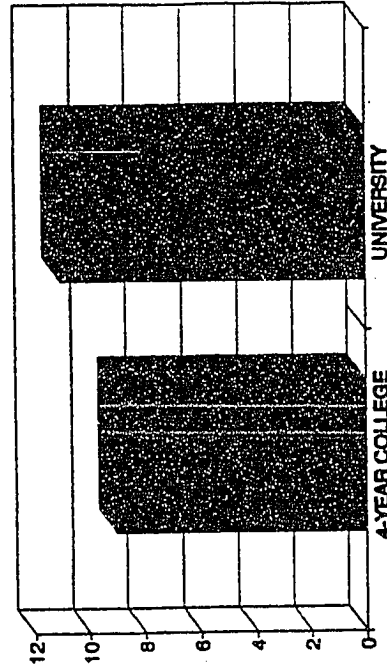
SCHOOL TYPE	NO	YES	ALL
4-YEAR COLLEGE	76	5	81
UNIVERSITY	144	12	156
ALL	220	17	237

CHI SQUARE = 0.185

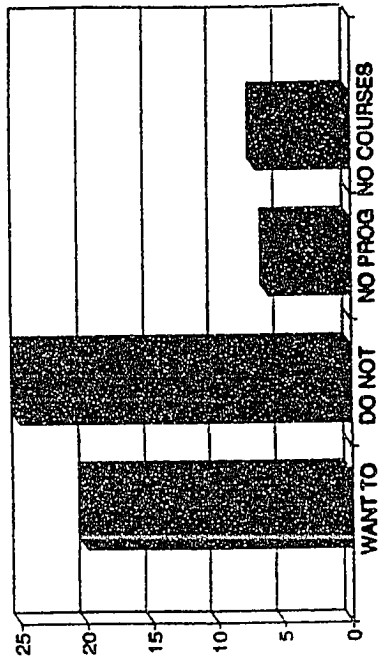
MISSING CASES = 45



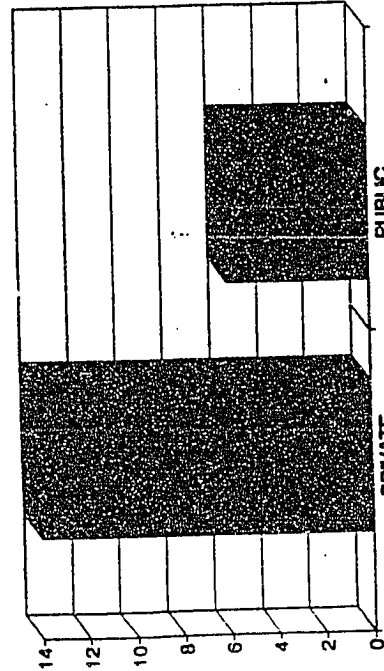
Classification of Schools by the Minority/Non-Minority Status
Figure 42



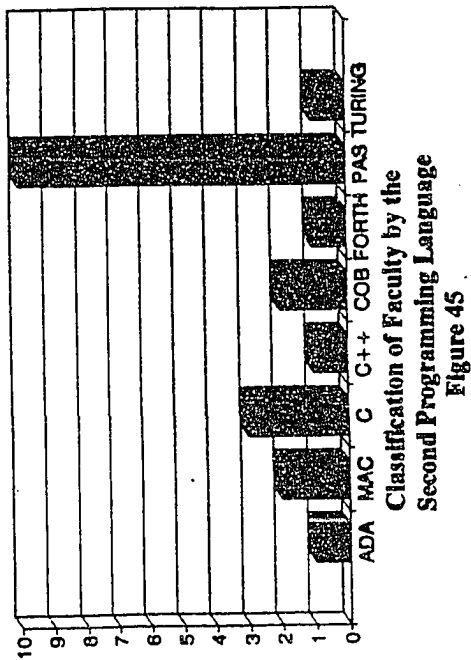
Classification of Schools by Their Type (4-YR College/University)
Figure 44



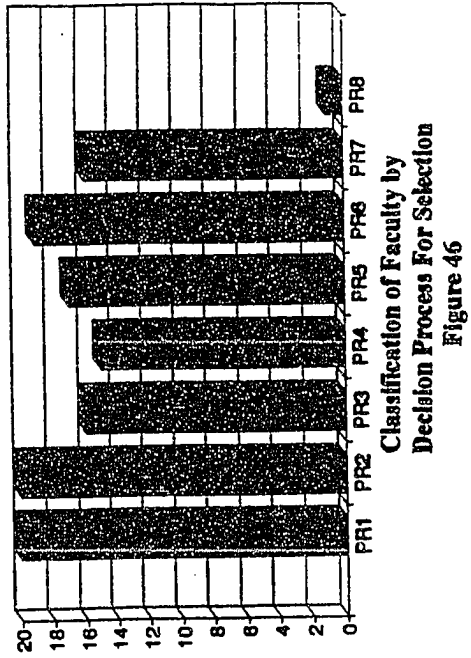
Classification of Schools by Willingness to Participate
Figure 41



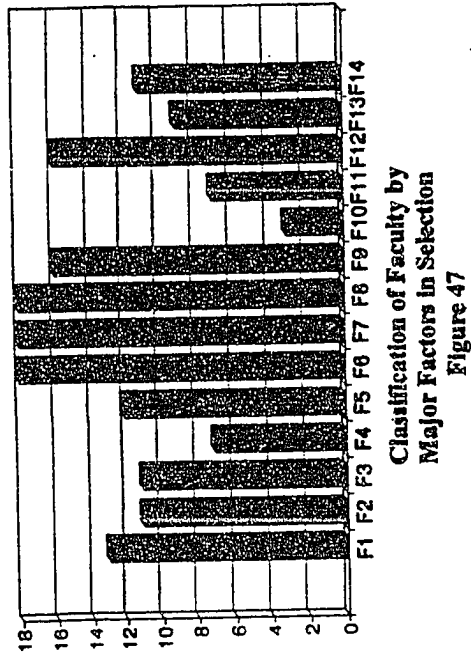
Classification of Schools by Their Type (Public/Private)
Figure 43



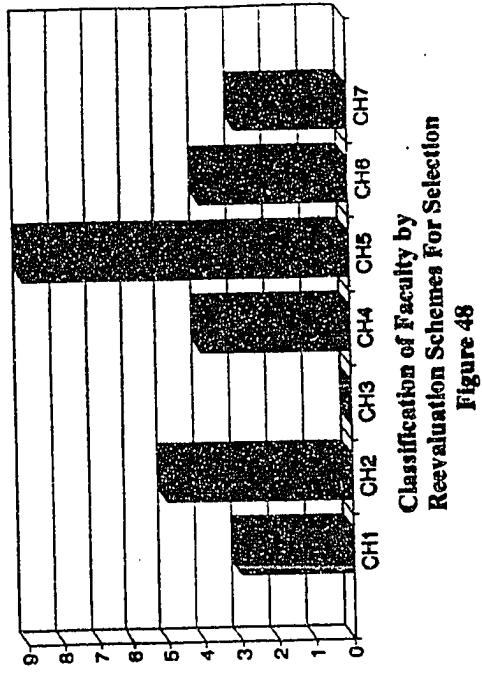
Classification of Faculty by the Second Programming Language
Figure 45



Classification of Faculty by Decision Process For Selection
Figure 46



Classification of Faculty by Major Factors in Selection
Figure 47



Classification of Faculty by Reevaluation Schemes For Selection
Figure 48

Table 94

Cross Tabulation of
School Type And Only Language Was Offered by The School as a Reason
in Learning The Second Programming Language

SCHOOL TYPE	NO	YES	ALL
4-YEAR COLLEGE	75	6	81
UNIVERSITY	143	13	156
ALL	218	19	237

CHI SQUARE = 0.062

MISSING CASES = 45

Table 95

Cross Tabulation of
School Type And The Language Was Required For The Major as a Reason
in Learning The Second Programming Language

SCHOOL TYPE	NO	YES	ALL
4-YEAR COLLEGE	39	42	81
UNIVERSITY	70	86	156
ALL	109	128	237

CHI SQUARE = 0.230

MISSING CASES = 45

Table 96

Cross Tabulation of
School Type And it Was an Easy Language as a Reason in Selecting The
Second Programming Language

SCHOOL TYPE	NO	YES	ALL
4-YEAR COLLEGE	75	6	81
UNIVERSITY	148	8	156
ALL	223	14	237

CHI SQUARE = 0.498

MISSING CASES = 45

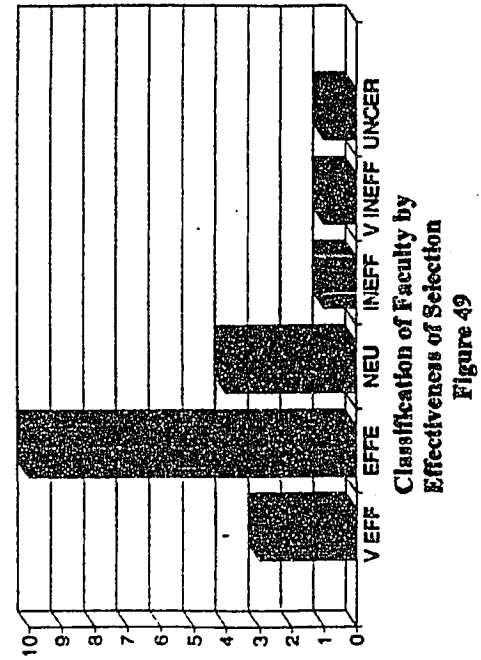
In this third group, school type (minority/non-minority) was cross tabulated against each of the reasons listed in the selection process of a second programming language. Only one of the ten reasons listed were significantly correlated with the type of school when making the selection of a second programming language. The reason that students had interest in learning the language shows a positive correlation between the type of school with a Chi square value is = 0.807. The reason that the language was used in the second programming language class had no statistical significance over the type of school. (Table 97, Chi square value = 4.167) and the reason that the language was required for the major also did not show any significant relationship over the type of school. (Table 104, Chi square value = 4.625). Of course, for five of these reasons, it was difficult to say whether the correlation had any meaning. Since some of the cell values were too small to compute a meaningful chi square value. For example, the reason that there was a good job market for the language, the reason that it was the only language offered by the school, the reason that it was an easy language, the reason that someone told the student to learn the language, the reason that it was the only language available, and the reason that it was a popular language, the Chi square values show that there might be a significant relationship between these reasons and the type of school when making the selection of a second programming language. But since the some of the cell values were too small to come with a concrete conclusion. These reasons were left for further analysis.

Table 98 shows that 40% of the students from minority schools surveyed learned the second programming language because of their interests in learning the language. This percentage is 33% for non-minority schools. Chi square value is equal to 0.807.

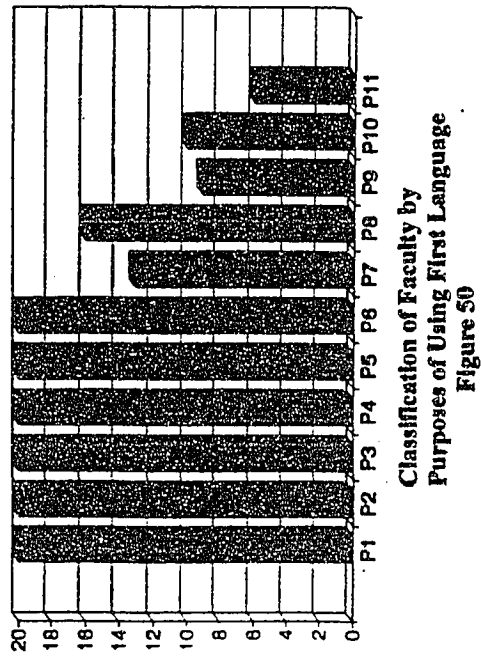
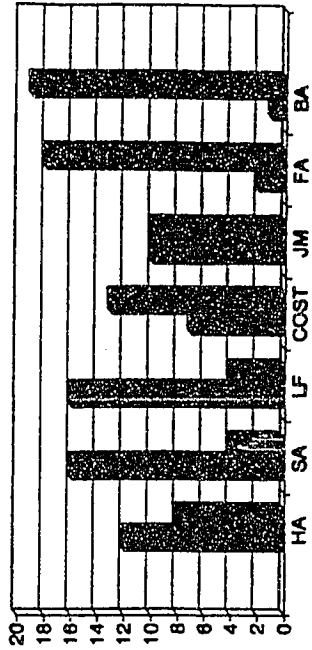
Table 99 shows that the reason 20% of the students took the second programming language class because of the advice of someone while for minority school students this percentage is 31% and for non-minorities it is 17%. Chi square value is equal to 2.624.

Table 100 shows that 22% of the students took the second programming language class because it was the only class available at that time. This ratio is 20% for minority schools to 2% for non-minorities. Chi square value is 0.065.

Only 18% of the students learned the second programming language because of its popularity. This percentage is 15% for minority school students and 19% for non-minority students. Table 101 shows the Chi square value = 0.328.



Classification of Faculty by Factors in the Selection Process
Figure 51



Classification of Faculty by Time Line For Changing to New Language
Figure 52

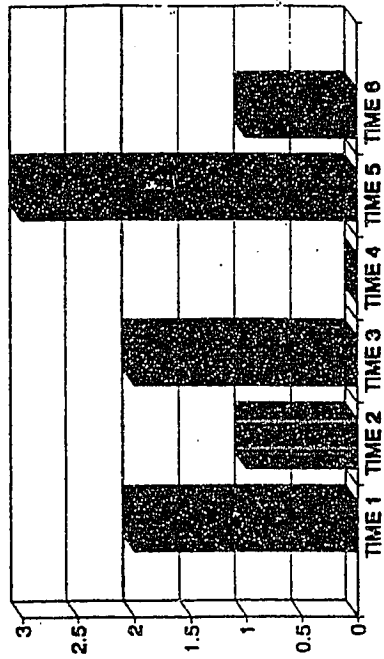


Table 97

Cross Tabulation of
School Type And The Language Was Used in The Second Programming
Language Class as a Reason in Learning The Second Programming Language

SCHOOL TYPE	NO	YES	ALL
MINORITY	30	29	59
NON-MINORITY	117	61	178
ALL	147	90	237

CHI SQUARE = 4.167

MISSING CASES = 45

Table 98

Cross Tabulation of
School Type And Wanted to Learn The Language as a Reason in Learning
The Second Programming Language

SCHOOL TYPE	NO	YES	ALL
MINORITY	36	23	59
NON-MINORITY	120	58	178
ALL	156	81	237

CHI SQUARE = 0.807

MISSING CASES = 45

Table 99

Cross Tabulation of
School Type And Someone Told to Learn The Language as a Reason in
Learning The Second Programming Language

SCHOOL TYPE	NO	YES	ALL
MINORITY	43	16	59
NON-MINORITY	147	31	178
ALL	190	47	237

CHI SQUARE = 2.624

MISSING CASES = 45

Table 100

Cross Tabulation of
School Type And Only Language Available as a Reason in Learning The
Second Programming Language

SCHOOL TYPE	NO	YES	ALL
MINORITY	47	12	59
NON-MINORITY	139	39	178
ALL	286	51	237

CHI SQUARE = 0.065

MISSING CASES = 45

Table 101

Cross Tabulation of
School Type And The Language Was Popular as a Reason in Learning The
Second Programming Language

SCHOOL TYPE	NO	YES	ALL
MINORITY	50	9	59
NON-MINORITY	145	33	178
ALL	195	42	237

CHI SQUARE = 0.328

MISSING CASES = 45

Table 102

Cross Tabulation of
School Type And Job Market For The Language as a Reason in Learning
The Second Programming Language

SCHOOL TYPE	NO	YES	ALL
MINORITY	55	4	59
NON-MINORITY	165	13	178
ALL	220	17	237

CHI SQUARE = 0.018

MISSING CASES = 45

Table 103

Cross Tabulation of
School Type And Only Language Was Offered by The School as a Reason
in Learning The Second Programming Language

SCHOOL TYPE	NO	YES	ALL
MINORITY	52	7	59
NON-MINORITY	166	12	178
ALL	218	19	237

CHI SQUARE = 1.577

MISSING CASES = 45

Table 104

Cross Tabulation of
School Type And The Language Was Required For The Major as a Reason
in Learning The Second Programming Language

SCHOOL TYPE	NO	YES	ALL
MINORITY	39	42	81
NON-MINORITY	70	86	156
ALL	109	128	237

CHI SQUARE = 4.625

MISSING CASES = 45

Table 105

Cross Tabulation of
School Type And it Was an Easy Language as a Reason in Selecting The
Second Programming Language

SCHOOL TYPE	NO	YES	ALL
MINORITY	54	5	59
NON-MINORITY	169	9	178
ALL	223	14	237

CHI SQUARE = 0.932

MISSING CASES = 45

Analysis of the Faculty Survey

The first discussion will focus on the frequency tables for each of the variables in the faculty questionnaire. All the faculty questions will be divided into four groups: first group of questions will include the first 4 questions dealing with their affiliation with the department, number of years of affiliation, language(s) used in the first programming language classes, and the language used in the second programming language classes. The second group will consists of next six questions and Question 13 which were dealing with programming languages used in the first programming language classes, the reasons behind the selection of this language as the first programming language, major factors which played a role in the above selection, how often this choice gets re-evaluated, and how effective is this evaluation process. The third group consists of the next two questions dealing with the course content of the first programming language class. For this group, every question will be a contingency table showing the responses broken down into the percentages of each of the categories. The other type of frequency table will be in the form of a bar graph. The fourth group will be consisting of the last two questions which mainly deal with the future of the next first programming language to be used and the reasons behind selecting these languages.

The second discussion will center on the crosstabulations of demographic variables against the remaining variables, namely, first programming language, reasons for selecting the first programming language, second programming language, and the reasons for selecting the second programming language, selection procedure of the first language, major factors involved in making the decision of selecting the first programming language, course content, effectiveness of the selection procedure, looking for a new first programming language, the reasons behind this new selection, and whether students will be involved in the decision process.

Table 106

Classification of Schools by Status of Minority/non-minority

TYPE OF SCHOOL	COUNT
MINORITY	4
NON-MINORITY	16
TOTAL	20

Table 107

Classification of Schools by The Type of School (Private/public)

TYPE OF SCHOOL	COUNT
PRIVATE	14
PUBLIC	6
TOTAL	20

Table 108

Classification of Schools by The Type of School (4-year College/university)

TYPE OF SCHOOL	COUNT
4-YEAR COLLEGE	9
UNIVERSITY	11
TOTAL	20

Table 109

Classification of Schools by Their Participation in The Survey

CODE	TYPE OF RESPONSE FROM THE SCHOOL	COUNT
WANT TO	WANTED TO PARTICIPATE IN THE STUDY	20
DO NOT	NO RESPONSE OR NO PARTICIPATION	25
NO PROG	NO PROGRAMMING COURSES OFFERED IN THE SCHOOL	6
NO COURSES	NO PROGRAMMING COURSE OFFERED DURING THAT SEMESTER	7

Group I - Faculty Teaching the First Programming Language Class

Four questions to count the number of respondents teaching the first programming language classes were asked as follows:

1. Provide the name of the department in which you work.
2. How many years you have been affiliated with the department?
3. Which programming language is taught/used in the first programming language course?
4. Which programming language is taught/used in the second programming language course?

Table 106 shows the breakdown of schools who participated by the status of minority/non-minority and the same data is graphically depicted in Figure 42.

Table 107 shows the breakdown of schools who participated by the status of public/private and the same data is graphically depicted in Figure 43. Table 108 shows the breakdown of schools who participated by the status of four-year college/university and the same data is graphically depicted in Figure 44. Table 109 shows the breakdown of schools by their participation and the same data is graphically depicted in Figure 41.

Table 110 shows the breakdown of faculty responses to the first question and the same data is graphically depicted in Figure 59. Obviously, 70% of the faculty teaching the first programming language classes were affiliated either with Computer Science department or with Mathematics and Computer Science

department. 20% of the faculty came from the Business department and the remaining 10% of the respondents were working with the Mathematics department.

Table 111 and Figure 60 show that 35% of the faculty have been affiliated with the department for between 1 and 4 years. Another 35% of them working with the department between 6 and 10 years. Remaining 30% have been involved in teaching the first programming language between 14 and 20 years.

After looking at the responses in Table 112 and Figure 40, Pascal seems to be the winner as the mostly used first programming language since 65% of the faculty were using Pascal in their first programming language classes. 10% of them were using BASIC, ADA, C, C++ COBOL, FORTRAN, and dBASE had a share of 5% each as first programming languages.

Table 113 shows the breakdown of faculty responses to Question 4 and the same data is graphically depicted in Figure 45. Pascal came out to be the winner in the category of the mostly used second programming language, since 45% of the faculty were using Pascal in their second programming language class. 20% of the faculty were convinced in using C/C++ as their second programming language. 10% of them were using COBOL and Assembly languages in their second programming language classes. Only 5% them had decided to use ADA, FORTH, and TURING in each of their second programming classes.

Table 110

V1 Classification of Faculty by the Departments in Which They Work

DEPARTMENT NAME	COUNT
Art	0
Biology	0
Business	4
Chemistry	0
Computer Information Systems	0
Computer Science	7
Computer Systems Engineering	0
Electrical Engineering	0
English	0
History	0
Mathematics	2
Mathematics & Computer Science	7
Music	0
Physics	0
Small Systems Computing	0
Others	0
Total	20

Table 111

V2 Classification of Faculty by the Number of Years of Affiliation with the Department

NO. OF YEARS	COUNT
1	2
2	2
3	2
4	1
5	0
6	1
7	2
8	2
9	1
10	1
11	0
12	0
13	0
14	1
15	0
16	0
17	0
18	0
19	0
20	1

Table 112

V3 Classification of Faculty by The First Programming Language They Teach

FIRST PROGRAMMING LANGUAGE USED	COUNT
ADA	1
APL	0
ASSEMBLY/MACHINE	0
BASIC	2
C	1
C++	1
COBOL	1
SCHEME	0
FORTH	0
FORTRAN	1
HYPERTALK	0
LISP	0
LOGO	0
MODULA-2	0
PASCAL	13
PL/I	0
PROLOG	0
SMALLTALK	0
TURING	0
OTHER (dBASE)	1
TOTAL	20

Table 113

V4 Classification of Faculty by The Second Programming Language They Teach

SECOND PROGRAMMING LANGUAGE USED	COUNT
ADA	1
APL	0
ASSEMBLY/MACHINE	2
BASIC	0
C	3
C++	1
COBOL	2
SCHEME	0
FORTH	1
FORTRAN	0
HYPERTALK	0
LISP	0
LOGO	0
MODULA-2	0
PASCAL	10
PL/I	0
PROLOG	0
SMALLTALK	0
TURING	1
OTHER	0
TOTAL	20

Group II - Main Factors and Procedures Behind the Selection of the First Programming Language

Six questions to count the number of respondents teaching the first programming language classes were asked as follows:

5. How does the programming language chosen in introductory computer science courses relate to programming languages used in other computer science courses?
6. What is the decision process for choosing the first programming language to be used in the first programming language courses? Who is involved in making this decision?
7. What factors played a major role in the above decision?
8. How often is the decision process regarding the choice of a first programming language reevaluated?
9. What, in your opinion, is the purpose of using this programming language in the first programming language class?
10. In your opinion, how would you rate the effectiveness of your department's ability to accurately select first programming language?
13. In your opinion, which factors are most important for choosing a first programming language?

Table 114 shows the breakdown of faculty responses to Question 5. 60% of the respondents use the same language in their second programming language class while 40% use a different language. Also, 60% of the respondents claimed that they use the same language in their data structures and other computer science related courses. Remaining 40% of the respondents use different programming languages in these courses.

Table 115 displays the breakdown of the respondents by the decision process used to select the first programming language in their departments. The same data is graphically depicted in Figure 46. 100% of the respondents agreed that either they make the selection of the language to be used in the first programming language class by themselves or with the consultation from the Computer Science committee. 95% of the respondents look for a structural language that implements modularity, concurrency, reusable code when making their selection for the first programming language. 85% of the respondents claimed that they look for what other schools do in this regards and also the transfer needs of the students who want to transfer from their schools to other schools. Ease of use of a language and job market demands were used by 80% of the faculty to make the selection of the first programming language. Only 5% of them consult to the business advisory council of teachers and business members when deciding the language to used in their first programming language class.

Table 114

V5 Classification of First Programming Language And Other Computer Science Courses

COURSES	SAME LANGUAGE	DIFFERENT LANGUAGE	TOTAL
PROGRAMMING II	12	8	20
DATA STRUCTURES	12	8	20
OTHER COURSES	12	8	20

Table 115

V6 Classification of Faculty by The Decision Process For Selecting The First Programming Language

CODE	DECISION PROCESS	TOTAL
PR1	FACULTY THAT TEACH CS CLASSES DECIDE	20
PR2	INDIVIDUAL FACULTY DECIDES WITH GUIDANCE FROM CS COMMITTEE	20
PR3	JOB MARKET REQUIREMENTS	16
PR4	SOFTWARE LIMITATIONS	15
PR5	WHAT OTHER SCHOOLS DO AND TRANSFER NEEDS OF STUDENTS	17
PR6	STRUCTURAL LANGUAGE THAT IMPLEMENTS MODULARITY, CONCURRENCY, REUSABLE CODE	19
PR7	DESIGN AND STRUCTURE OF A LANGUAGE, EASE OF USE OF LANGUAGE	16
PR8	BUSINESS ADVISORY COUNCIL OF TEACHERS AND BUSINESS MEMBERS	1

Table 116 shows the breakdown of faculty responses to the factors which play a major role in the selection of the first programming language with the collapsed version graphically shown in Figure 47. 90% of them claimed that the syntax, logic and structure of a language were the most important factors for them when they make the selection of the first programming language. 80% of the respondents claimed to look for the potential for data structures and ability to create true abstract data types when making this selection. 65% wanted to make sure that the language had the capability of teaching good programming habits. Compiler cost and its availability was the concern of 55% of the faculty. 60% of the faculty watch the job market and make the selection. Only 35% of them worry about the knowledge of teaching staff and availability of the texts and only 15% of them worry about whether the first programming language course will be transferable into the approved curriculum when making the selection. Finally, only 10% of them claim that they consult with other schools selection while making their selection of the first programming language.

Table 117 shows the breakdown of faculty responses to Question 8 and the same data is graphically depicted in Figure 48. 45% of the respondents claim that they re-evaluate the selection of the first programming language every 2-4 years. 25% re-evaluate the selection as often as the job market and industry demands. 20% of the faculty make the selection every year. 15% of them make the changes as often as the demands of the faculty teaching the first programming language classes.

Table 116

V7 Classification of Faculty by The Factors Playing Major Role in The Selection of The First Programming Language

CODE	MAJOR FACTORS IN SELECTING FIRST LANGUAGE	TOTAL
F1	FORM GOOD PROGRAMMING HABITS	13
F2	COMPILER COST	11
F3	COMPILER AVAILABILITY	11
F4	TEACHING STAFF KNOWLEDGE	7
F5	JOB MARKET	21
F6	THE SYNTAX OF LANGUAGE	18
F7	THE LOGIC OF LANGUAGE	18
F8	STRUCTURE OF LANGUAGE	18
F9	POTENTIAL FOR DATA STRUCTURES	16
F10	TRANSFERABILITY OF COURSES INTO APPROVED CURRICULUM	3
F11	AVAILABILITY OF TEXTS	7
F12	ABILITY TO CREATE TRUE ABSTRACT DATA TYPES	16
F13	PRODUCT INTERFACE	11
F14	OTHER SCHOOLS CHOOSE THE LANGUAGE	2

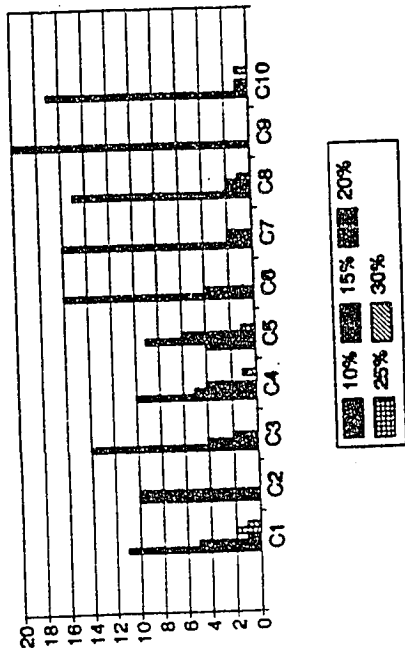
Table 117

V8 Classification of Faculty by How Often Choice of a First Programming Language is Re-evaluated

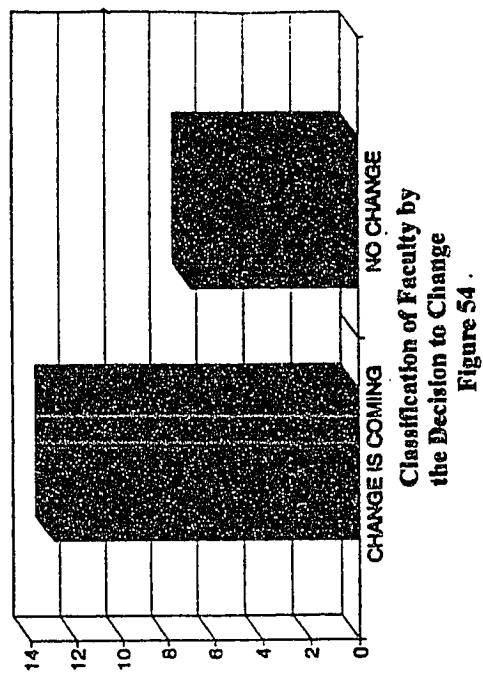
CODE	RE-EVALUATION TIME	TOTAL
CH1	OFTEN AS FACULTY REQUEST	3
CH2	OFTEN AS JOB MARKET DEMANDS	5
CH3	WAIT FOR NEW ACM GUIDELINES	0
CH4	EVERY YEAR	4
CH5	EVERY TWO - FOUR YEARS	9
CH6	EVERY FIVE - TEN YEARS	4
CH7	SELDOM	3

Table 118 shows the breakdown of faculty responses to the effectiveness of the department's selection process and the same is graphically depicted in Figure 49. 50% of the faculty agree with the fact that their department was effective in making the selection of a first programming language. Only 15% thought that the department was very effective in making this selection. Remaining 35% of the respondents either were neutral in making this claim or were not happy with the effectiveness of the department in making the selection of the first programming language.

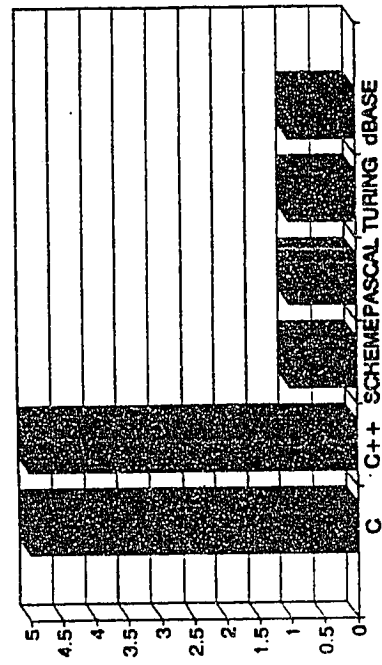
Table 119 shows the breakdown of faculty responses to Question 13 and the same data is graphically depicted in Figure 51. 80% of the faculty claim that software availability and language features were equally important factors in making the selection of the first programming language. 60% were concerned about the hardware availability when making their selection. Only 35% were worried about the cost when making the selection of the language and 10% were worried about the qualified faculty to teach the class when making their selection.



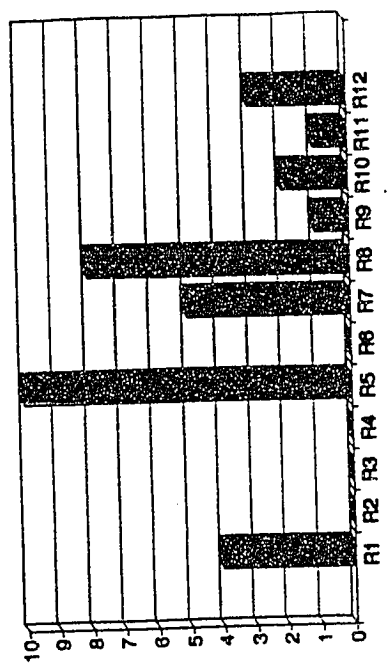
Classification of Faculty by Percentage of Course Content
Figure 53



Classification of Faculty by the Decision to Change
Figure 54



Classification of Faculty by the New First Programming Language
Figure 55



Classification of Faculty by Reasons For New First Language
Figure 56



Table 118

V10 Classification of Faculty by The Effectiveness of the Department in Selecting a First Programming Language

CODE	EFFECTIVENESS	COUNT
VEFF	VERY EFFECTIVE	3
EFFE	EFFECTIVE	10
NEU	NEUTRAL	4
INEFF	INEFFECTIVE	1
VINEFF	VERY INEFFECTIVE	1
UNCER	UNCERTAIN	1
	TOTAL	20

Table 119

V13 Classification of Faculty by The Most Important Factors For Choosing The First Programming Language

CODE	FACTORS	YES	NO	TOTAL
HA	HARDWARE AVAILABILITY	12	8	20
SA	SOFTWARE AVAILABILITY	16	4	20
LF	LANGUAGE FEATURES	16	4	20
COST	COST	7	13	20
JM	JOB MARKET	10	10	20
FA	FACULTY AVAILABILITY	2	18	20
BA	OTHERS (E.G. BOOK AVAILABILITY)	1	19	1

Table 120 shows the breakdown of faculty responses to Question 11 and the same is graphically depicted in Figure 50. From the data in Table 26 it is clear that all the respondents teach sequencing, loop decisions, subroutines, variables types, fundamentals in structured, procedural programming concepts, parameters and modularity using their first programming language. 80% of the faculty use the language in their first programming class because of its availability. 65% wanted to use that language as their first programming language because they claimed that the language provided job-related skills, it was usable in real-world, and it was dealing with societal issues. 50% of the faculty selected the language because of its ease of design, structure and they also claimed that the language was designed as a teaching language. 45% of the respondents chose the language because of their belief that it gave students better understanding of GUI (Graphical User Interface) and other features of work stations. Only 30% of them made a claim that the only reason they chose the language because it was easier to introduce and was easy to enforce data abstraction using the language.

Table 120

V9 Classification of Faculty by The Purposes of Using This Language

CODE	PURPOSE	COUNT
P1	TEACH SEQUENCING	20
P2	TEACH LOOP DECISIONS	20
P3	TEACH SUBROUTINES	20
P4	TEACH VARIABLE TYPES	20
P5	TEACH FUNDAMENTALS IN STRUCTURED PROGRAMMING CONCEPTS	20
P6	PARAMETERS, MODULARITY	20
P7	TO PROVIDE JOB-RELATED SKILLS, USABLE IN REAL-WORLD, DEALS WITH SOCIETAL ISSUES	13
P8	LANGUAGE AVAILABILITY	16
P9	GIVES STUDENTS BETTER UNDERSTANDING OF GUI AND OTHER FEATURES OF WORK STATIONS	9
P10	EASE OF DESIGN, STRUCTURE, DESIGNED AS A TEACHING LANGUAGE	10
P11	EASIER TO INTRODUCE AND ENFORCE DATA ABSTRACTION	6

Group III - Course Content of the First Programming Language Class

Two questions to count the number of respondents teaching the first programming language classes were asked as follows:

11. Estimate the percentage of the course content:

- a. Flow-Charts or Pseudocode
- b. Assignment Statements and Variables
- c. Conditional Statements
- d. Loops
- e. Procedures and Functions
- f. Parameters
- g. Input/Output
- h. Data Structures
- I. Other

12. What percentage of course time is spent in each delivery method category?

Table 121 shows the breakdown of faculty responses to Question 11 with the collapsed version graphically shown in Figure 53. All the respondents spent 10% of their time on the topics which include hardware, recursion, algorithms when they teach the first programming language classes. 85% of them spent

10% of their time teaching problem solving. 80% of the faculty spent 10% of their time covering parameters, and input/output when they taught the first programming language classes. 75% of them spend 10% of the course time covering data structures. Actually, everyone in the survey spent almost evenly (10%) of the course time on each of the 10 topics listed in the questionnaire. Only 5% of the faculty spent 30% of the course time on topics such as loops, flow charts or pseudocode, and problem solving. More than 60% of the respondents spent 15% to 20% of their course time on procedures, functions and parameters.

Table 122 shows the breakdown of faculty responses to Question 12. More than 50% of the faculty surveyed spent on the average 65% - 75% of the course time presenting the content of the first programming language class in the lecture form. Almost 80% of them spent between 10% - 40% of the course time doing the hands-on/lab part when teaching the first programming language class. Almost all of the faculty surveyed spent less than 15% of the course time on the discussion while presenting the material.

Table 121

V11 Classification of Faculty by The Percentage of Course Content in Teaching
The First Programming Language Class

CODE	COURSE CONTENT	10%	15%	20%	25%	30%
C1	FLOW CHARTS OR PSEUDOCODE	11	5	1	2	1
C2	ASSIGNMENT STATEMENTS AND VARIABLES	10	10	0	0	0
C3	CONDITIONAL STATEMENTS	14	4	2	0	0
C4	LOOPS	10	5	4	0	1
C5	PROCEDURES AND FUNCTIONS	4	9	6	1	0
C6	PARAMETERS	16	4	0	0	0
C7	INPUT/OUTPUT	16	2	2	0	0
C8	DATA STRUCTURES	15	2	2	1	0
C9	HARDWARE, RECURSION, ALGORITHMS, OTHER TOPICS IN CS.	20	0	0	0	0
C10	PROBLEM SOLVING	17	1	1	0	1

Table 122

V13 Classification of Faculty by The Percentage of Time Spent in Each Delivery Method When Teaching The First Programming Language Class

TIME SPENT	LECTURE	HAND-ON/LAB	DISCUSSION	OTHERS
0%	0	2	6	17
5%	0	2	3	0
10%	0	1	5	0
15%	0	2	0	1
20%	1	2	2	0
25%	1	4	4	2
30%	0	2	0	0
35%	1	1	0	0
40%	2	1	0	0
45%	1	0	0	0
50%	2	2	0	0
55%	0	0	0	0
60%	2	0	0	0
65%	0	0	0	0
70%	2	1	0	0
75%	1	0	0	0
80%	4	0	0	0
85%	1	0	0	0
90%	1	0	0	0
95%	0	0	0	0
100%	1	0	0	0
TOTAL	20	20	20	20

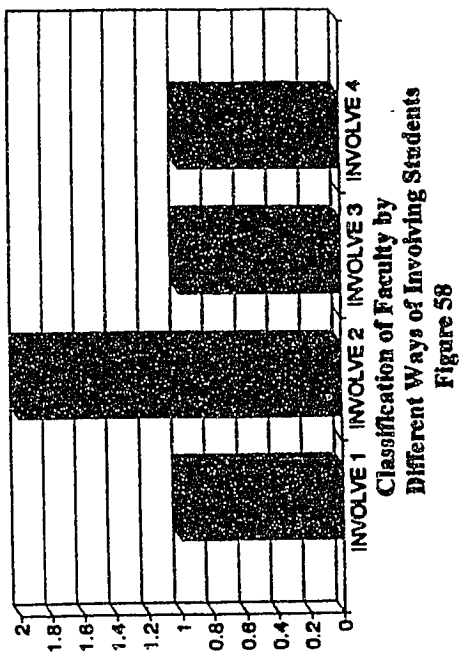
Group IV - Future of First Programming Languages

Two questions to count the number of respondents teaching the first programming language classes were asked as follows:

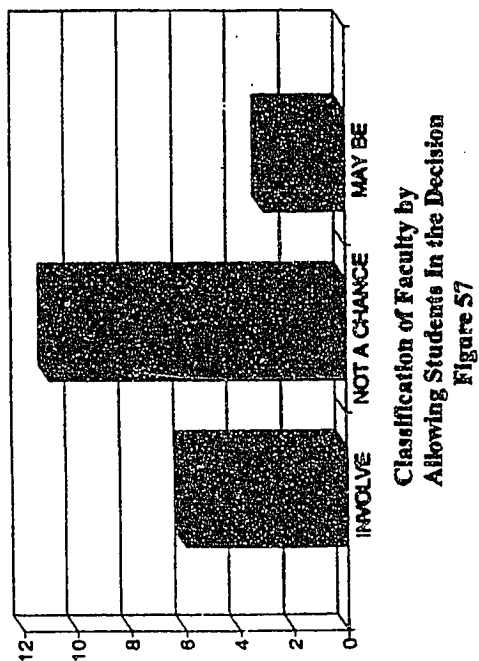
1. Are you planning to change to another first programming language? If yes, what is the new first programming language? What are the reasons behind this selection? When do you expect the change?
2. Will students involve in the decision process of selecting a new first programming language? If yes, how?

Table 123 shows that everyone in the survey was sure about their decision of whether they were changing to a new first programming language or not. Same data is graphically depicted in Figure 54. 70% of the faculty were going with a new first programming language and only 30% of them staying with the same language in their first programming language classes.

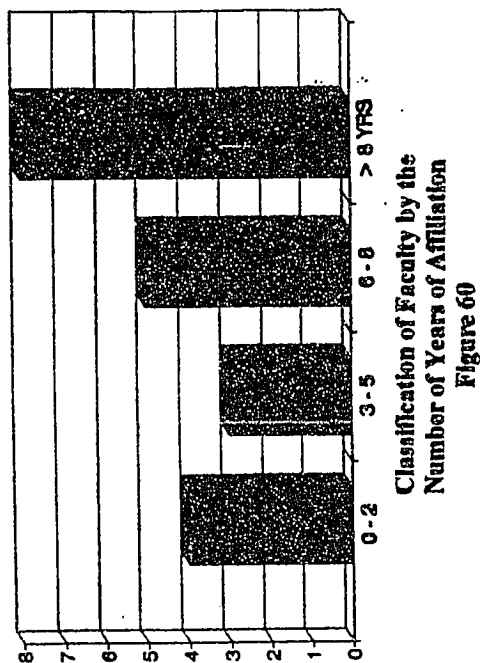
Table 124 shows the breakdown of responses for the second part of Question 14 with the collapsed version graphically shown in Figure 55. 50% of the faculty were going to use either C or C++ as their first programming language. 5% of them were either choosing Scheme or Turing or dBASE as their first programming language. Only 5% of them were going with Pascal as their first programming language.



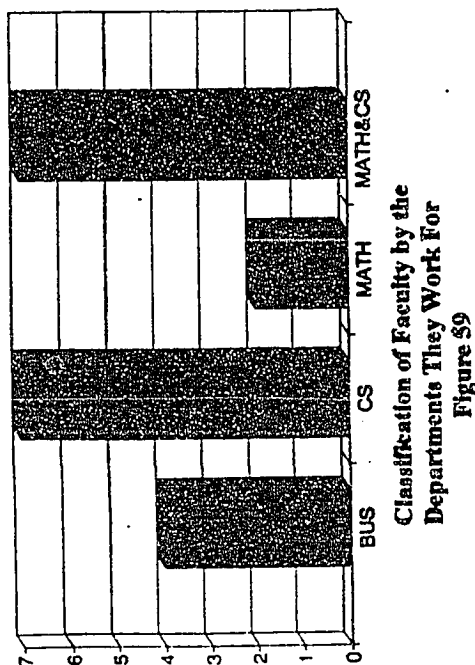
Classification of Faculty by
Different Ways of Involving Students
Figure 58



Classification of Faculty by
Allowing Students In the Decision
Figure 57



Classification of Faculty by the
Number of Years of Affiliation
Figure 60



Classification of Faculty by the
Departments They Work for
Figure 59

Table 123

V14 Classification of Faculty by The Decision of Changing to Another First Programming Language

CHANGING TO A NEW LANGUAGE	COUNT
YES	13
NO	7
MAY BE	0
TOTAL	20

Table 124

V15 Classification of Faculty by The New First Programming Language

NEW FIRST PROGRAMMING LANGUAGE	COUNT
ADA	0
APL	0
ASSEMBLY/MACHINE	0
BASIC	0
C	5
C++	5
COBOL	0
SCHEME	1
FORTH	0
FORTRAN	0
HYPERTALK	0
LISP	0
LOGO	0
MODULA-2	0
PASCAL	1
PL/I	0
PROLOG	0
SMALLTALK	0
TURING	1
OTHER	1
TOTAL	14

Table 125 shows the breakdown of responses for the third part of Question 14 with the collapsed version graphically shown in Figure 56. The most important reason for changing to a new language was job market demand. 50% of the faculty surveyed were changing to C or C++ as their new first programming language due to the demand for C/C++ programmers. 40% of the respondents chose this language because they perceived that this new programming language (C/C++) as the 'future' of programming. 25% of them had chosen the new language because of its features. 20% of the faculty were changing to the new language because this new language formed good programming habits. Only 10% of the faculty wanted the new language because it had the functionality of Pascal.

Table 126 shows that 36% of the faculty were going with a new first programming language within 1 year time period. 57% of them were not really sure about the time table for the new first programming language. 7% were going to wait for few years before they change to a new first programming language. Same data is graphically depicted in Figure 52 in collapsed version.

Table 125

V16 Classification of Faculty by The Reasons For Selecting This New First Programming Language

CODE	REASONS	COUNT
R1	FORM GOOD PROGRAMMING HABITS	4
R2	COMPILER COST	0
R3	COMPILER AVAILABILITY	0
R4	TEACHING STAFF KNOWLEDGE	0
R5	JOB MARKET	10
R6	HARDWARE AVAILABILITY	0
R7	LANGUAGE FEATURES	5
R8	PERCEIVED AS 'FUTURE' OF PROGRAMMING	8
R9	MORE STUDENTS KNOW 'BASIC' NOW	1
R10	LANGUAGE HAS THE FUNCTIONALITY OF PASCAL	2
R11	DIFFERENT INSTRUCTORS USE DIFFERENT LANGUAGES	1
R12	OTHERS	3

Table 126

V17 Classification of Faculty by The Time Deadline For Changing to a New First Programming Language

CODE	DEADLINE FOR CHANGE	COUNT
TIME 1	THIS SEMESTER	2
TIME 2	NEXT SEMESTER	1
TIME 3	NEXT YEAR	2
TIME 4	AFTER NEW ACM GUIDELINES	0
TIME 5	NOT KNOWN YET	3
TIME 6	FEW YEARS	1
NO CHANGE	NO CHANGE AT ALL	11
	TOTAL	20

Table 127 shows that when they are making the decision of choosing a new first programming language only 30% of them were going to involve students in this decision process 15% were not sure about this and remaining 55% were against the idea of involving students in the decision process of selecting the new first programming language. Same data is graphically depicted in Figure 57.

Table 128 the breakdown of faculty responses to Question 15 which shows that 50% of the faculty would ask the students who graduated from their major for their feedback when making the selection of a new first programming language. 17% of them would look at the market demand and student job-placement when they make the selection of the new first programming language. Other 17% would offer pilot courses and see the response before making the final decision of selecting the new first programming language. Surprisingly, only 17% wanted to ask the current students about their views for the selection of the new first programming language. Same results are graphically depicted in Figure 58.

Table 127

V18 Classification of Faculty by The Fact Whether Students Will Be Involved in The Decision of Selecting The New First Programming Language

CODE	WILL STUDENTS BE INVOLVED IN THE SELECTION?	COUNT
INVOLVE	YES	6
NOT A CHANCE	NO	11
MAY BE	MAY BE	3
	TOTAL	20

Table 128

V19 Classification of Faculty by The Ways Students Will Be Involved in The Selection Process of Choosing a New First Programming Language

CODE	WAYS STUDENTS WILL BE INVOLVED IN THE DECISION	COUNT
INVOLVE 1	LISTEN TO STUDENTS WHAT THEY WANT	1
INVOLVE 2	ASK THE STUDENTS WHO GRADUATED	2
INVOLVE 3	JOB-MARKET DEMAND AND STUDENT JOB-PLACEMENT	1
INVOLVE 4	OFFERING PILOT COURSES TO SEE THE DEMAND	1
NO INVOLVEMENT	NO, STUDENTS WILL NOT BE INVOLVED !!!	14
	TOTAL	20

Crosstabulations by Demographic Variables

To determine if any significant differences exist between demographic groups within the survey population of faculty, crosstabulations were obtained, using as independent variables the results of Questions 1, 2 and three other variables were coded in, namely, the type of school (public/private), the school code (4-year college/university), and the status of school (minority/non-minority).

There were six dependent variables. These were: the selection of a first programming language; the reasons behind the selection of this language; the major factors associated with this selection; the selection of a second programming language; the reasons behind the selection of this second programming language; the decision process in the above selection.

As the population was broken into subgroups, the frequencies in some individual cells were so small that it became necessary to collapse categories. The categories in both dependent variables, first programming language and second programming language were collapsed into two, namely, Pascal, and Other programming languages. Some of the independent variables which had a lot of categories in the original questionnaire also had to be collapsed, as needed.

Type of School and First Programming Language

The purpose of this analysis is to determine if the type of a school is a predictor of first programming language used in the Computer Programming I classes. The survey respondents were divided into three categories: Public/Private; 4-Year College/University; Minority/Non-Minority. The languages were grouped into two categories: Pascal and other programming languages.

As shown in Table 129, there seems to a positive correlation between the type of a school (public/private) and the choice of a first programming language used. Overall, 65% of the schools were using Pascal over the other languages in their first programming language class. For public schools this percentage goes up to 71% as compared to 62% for private schools. Chi square value is 0.196.

When the school type of four-year college/university is considered, the positive correlation still holds. The percentage of four-year colleges choosing Pascal in their first programming language class is 56% and for universities this percentage goes up to 73%. Table 130 shows that the Chi square value is 0.642. Table 131 shows that the positive correlation between the type of a school (minority/non-minority) and the selection of a first programming language exists.

When the schools were broken down by the minority/non-minority type, 75% of the minority schools were using Pascal in their first programming language class while 63% of the non-minority schools were using Pascal over the other languages as their first programming language. Chi square value is 0.220.

Table 129

Cross Tabulation of
School Type by First Programming Language (Pascal or Other)

SCHOOL TYPE	OTHER	PASCAL	ALL
PUBLIC	2	5	7
	2.45	4.55	7.00
PRIVATE	5	8	13
	4.55	8.45	13.00
ALL	7	13	20
	7.00	13.00	20.00

CHI-SQUARE = 0.196

Table 130

Cross Tabulation of
School Type by First Programming Language (Pascal or Other)

SCHOOL TYPE	OTHER	PASCAL	ALL
4-YEAR COLLEGE	4	5	9
	3.15	5.85	9.00
UNIVERSITY	3	8	11
	3.85	7.15	11.00
ALL	7	13	20
	7.00	13.00	20.00

CHI-SQUARE = 0.642

Table 131

Cross Tabulation of
School Type by First Programming Language (Pascal or Other)

SCHOOL TYPE	OTHER	PASCAL	ALL
MINORITY	1	3	4
	1.40	2.60	4.00
NON-MINORITY	6	10	16
	5.60	10.40	16.00
ALL	7	13	20
	7.00	13.00	20.00

CHI-SQUARE = 0.220

Type of School and Second Programming Language

The purpose of this analysis was to determine if the type of a school was a predictor of second programming language used in the computer programming II classes. The survey respondents were divided into three categories: Public/Private, 4-Year College/University; Minority/Non-Minority. The languages were grouped into two categories: Pascal and other programming languages.

As shown in Table 132, there seems to have a positive correlation between the type of a school (public/private) and the choice of a second programming language used. Overall, 50% of the schools were using Pascal over the other languages in their second programming language class. For private schools this percentage goes up to 54% as compared to 43% for public schools. Chi square value is 0.220.

When the school type of four-year college/university is considered, the positive correlation still holds. The percentage of four-year colleges choosing Pascal in their second programming language class is 67% and for universities this percentage goes down to 36%. Table 133 shows that the Chi square value is 1.818.

Table 134 shows the positive correlation between the type of a school (minority/non-minority) and the selection of a second programming language.

When the schools were broken down by the minority/non-minority type, 75% of the minority schools were using Pascal in their first programming language class while 44% of the non-minority schools were using Pascal over the other languages as their first programming language. Chi square value is 1.250.

Table 132

Cross Tabulation of
School Code by Second Programming Language (Pascal or Other)

SCHOOL CODE	OTHER	PASCAL	ALL
PUBLIC	4	3	7
	3.50	3.50	7.00
PRIVATE	6	7	13
	6.50	6.50	13.00
ALL	10	10	20
	10.00	10.00	20.00

CHI-SQUARE = 0.220

Table 133

Cross Tabulation of
School Type by Second Programming Language (Pascal or Other)

SCHOOL TYPE	OTHER	PASCAL	ALL
4-YEAR COLLEGE	3	6	9
	4.50	4.50	9.00
UNIVERSITY	7	4	11
	5.50	5.50	11.00
ALL	10	10	20
	10.00	10.00	20.00

CHI-SQUARE = 1.818

Table 134

Cross Tabulation of
School Status by Second Programming Language (Pascal or Other)

SCHOOL STATUS	OTHER	PASCAL	ALL
MINORITY	1	3	4
	2.00	2.00	4.00
NON-MINORITY	9	7	16
	8.00	8.00	16.00
ALL	10	10	20
	10.00	10.00	20.00

CHI-SQUARE = 1.250

Number of Years of Service within the Department and the Selection of First and Second Programming Languages

Would faculty members who have been teaching less than or equal to 7 years would be more prone to be innovative and thus more eager to embrace the new languages than those who are teaching for more than 8 years? To seek an answer to this question and since the contents of some of the cells were too small to come up with a meaningful statistical analysis, the survey population was divided into two groups according to the number of years of service, namely, seven or less years and eight or more years. Similarly, since the contents of some of the cells for the languages were also too small the languages were divided into two groups: Pascal and other programming languages.

Table 135 shows the cross tabulations of the years of service with the selection of a first programming language. 70% of the faculty who have been teaching 7 or less years were using Pascal over the other languages in their first programming language classes while 60% of the faculty with more than 8 years of service were using Pascal as their first programming language. The Chi square value is = 0.220.

Similar results were true with the selection of a second programming language except the relation was not as strong as in case of first programming language. 70% of the faculty who had been teaching less than or equal to 7 years were using Pascal as their second programming language while only 30%

of those with more than 8 years of teaching experience have selected Pascal in their second programming language classes. Table 136 shows that there was a positive correlation between the years of service of a faculty and the choice of a second programming language. The Chi square value is equal to 3.200.

Table 135

**Cross Tabulation of
Years of Service by First Programming Language (Pascal or Other)**

NO. OF YEARS OF SERVICE	OTHER	PASCAL	ALL
7 OR LESS THAN 7 YEARS	3	7	10
	3.50	6.50	10.00
8 OR MORE THAN 8 YEARS	4	6	10
	3.50	6.50	10.00
ALL	7	13	20
	10.00	13.00	20.00

CHI-SQUARE = 0.220

Table 136

**Cross Tabulation of
Years of Service by Second Programming Language (Pascal or Other)**

NO. OF YEARS OF SERVICE	OTHER	PASCAL	ALL
7 OR LESS THAN 7 YEARS	3	7	10
	5.00	5.00	10.00
8 OR MORE THAN 8 YEARS	7	3	10
	5.00	5.00	10.00
ALL	10	10	20
	10.00	10.00	20.00

CHI-SQUARE = 3.200

Type of School and Programming Languages Used in Different Classes in the Curriculum

The first two questions had to do with the correlation between the type of school and the languages in the first and second programming language classes. The next question had to do with the correlation between the type of school and programming languages used in different classes in their curriculum. In this analysis, there does seem to have significant correlation among populations groups as shown in Tables 137, 138, and 139.

Table 137 shows that overall only 40% of the faculty use Pascal in their different computer related classes. For four-year colleges this percentage goes up to 56% and for universities it goes down to 27%. The Chi square value is = 1.650.

The cross tabulation of school type (minority/non-minority) with the programming languages used in different computer classes shows a strong positive correlation. Only 25% minority schools use Pascal in their other computer classes as compared to 44% of the non-minority schools. Table 138 shows the correlation with the Chi square value is = 0.469.

When we cross tabulate school type (public/private) with the programming languages used in other computer science classes, we see that there is a strong positive correlation. Table 139 describes this correlation with a Chi square value of 0.220.

Only 43% faculty from public schools use Pascal in their other computer classes while 54% of the faculty from private schools use Pascal over the other programming languages in their curriculum.

Table 137

Cross Tabulation of
School Type by Programming Languages Used in Different Classes
(Pascal or Other)

SCHOOL TYPE	OTHER	PASCAL	ALL
4-YEAR COLLEGE	4	5	9
	5.40	3.60	9.00
UNIVERSITY	8	3	11
	6.00	4.40	11.00
ALL	12	8	20
	12.00	8.00	20.00

CHI-SQUARE = 1.650

Table 138

Cross Tabulation of
School Status by Programming Languages Used in Different Classes
(Pascal or Other)

SCHOOL STATUS	OTHER	PASCAL	ALL
MINORITY	3	1	4
	2.40	1.60	4.00
NON-MINORITY	9	7	16
	9.60	6.40	16.00
ALL	12	8	20
	12.00	8.00	20.00

CHI-SQUARE = 0.469

Table 139

Cross Tabulation of
School Code by Programming Languages Used in Different Classes
(Pascal or Other)

SCHOOL CODE	OTHER	PASCAL	ALL
PUBLIC	4	3	7
	3.50	3.50	7.00
PRIVATE	6	7	13
	6.50	6.50	13.00
ALL	10	10	20
	10.00	10.00	20.00

CHI-SQUARE = 0.220

Type of School and Reasons for Selecting a First Programming Language

The purpose of this analysis was to determine if type of a school was a predictor of different reasons for selecting a first programming language. There were eight reasons in the original questionnaire. Cross tabulations were done against all of these eight reasons with the type of schools.

Tables 140, 141, and 142 show the breakdown of selecting a programming language in the first programming language class because of the job market demands for that programming language and the type of school. 78% of the faculty from 4-year colleges used job market demands as a reason for choosing a first programming language where as 82% university faculty thought job market demands was a good reason for selecting a first programming language. Though the Chi square values were meaningful (0.051, 1.250, and 0.220), since some of the cell values were too small to derive any further meaningful statistical significance and hence were left for further analysis.

Table 140
Cross Tabulation of
School Type by Job Demands as a Reason for Selecting First Programming
Language

SCHOOL TYPE	YES	NO	ALL
4-YEAR COLLEGE	7	2	9
	7.20	1.80	9.00
UNIVERSITY	9	2	11
	8.80	2.20	11.00
ALL	16	4	20
	16.00	4.00	20.00

CHI-SQUARE = 0.051

Table 141
Cross Tabulation of
School Type by Job Demands as a Reason for Selecting First Programming
Language

SCHOOL STATUS	YES	NO	ALL
MINORITY	4	0	4
	3.20	0.80	4.00
NON-MINORITY	12	4	16
	12.80	3.20	16.00
ALL	16	4	20
	16.00	4.00	20.00

CHI-SQUARE = 1.250

Table 142
Cross Tabulation of
School Type by Job Demands as a Reason for Selecting First Programming
Language

SCHOOL CODE	YES	NO	ALL
PUBLIC	6	1	7
	5.60	1.40	7.00
PRIVATE	10	3	13
	10.40	2.60	13.00
ALL	16	4	20
	16.00	4.00	20.00

CHI-SQUARE = 0.220

PLEASE NOTE

Page(s) missing in number only; text follows.
Filmed as received.

189

UMI

Table 143
Cross Tabulation of
School Type by Design of a Language and Ease of its Use as a Reason for
Selecting First Programming Language

SCHOOL TYPE	YES	NO	ALL
4-YEAR COLLEGE	7	2	9
	7.20	1.80	9.00
UNIVERSITY	9	2	11
	8.80	2.20	11.00
ALL	16	4	20
	16.00	4.00	20.00

CHI-SQUARE = 0.051

Table 144
Cross Tabulation of
School Type by Design of a Language and Ease of its Use as a Reason for
Selecting First Programming Language

SCHOOL TYPE	YES	NO	ALL
MINORITY	3	1	4
	3.20	0.80	4.00
NON-MINORITY	13	3	16
	12.80	3.20	16.00
ALL	16	4	20
	16.00	4.00	20.00

CHI-SQUARE = 0.078

Table 145
Cross Tabulation of
School Type by Design of a Language and Ease of its Use as a Reason for
Selecting First Programming Language

SCHOOL TYPE	YES	NO	ALL
PUBLIC	5	2	7
	5.60	1.40	7.00
PRIVATE	11	2	13
	10.40	2.60	13.00
ALL	16	4	20
	16.00	4.00	20.00

CHI-SQUARE = 0.495

Tables 146, 147, and 148 cross tabulate school type (four-year college/university), (minority/non-minority), and (public/private) with the reason of software limitations as a reason for selecting a first programming language.

When school type (minority/non-minority) was cross tabulated against the reason of software limitations in making a selection of a first programming language, Table 146 shows that 75% of the faculty from the minority schools thought that software limitations was an important reason when selecting a first programming language and this percentage is also 75% for non-minority schools when making a selection of a first programming language. The Chi Square value is = 0.000.

56% of the 4-year college faculty choose the first programming language because of the software limitations while this percentage goes up 91% for the university faculty. Table 147 shows the Chi Square value = 3.300.

When the faculty from public and private schools were cross tabulated against the reason of software limitations, 100% faculty from public institutions over 62% faculty from private institutions selecting the first programming language with the reason of software limitations. Table 148 shows the cross tabulation with the Chi Square value = 3.590.

After examining the Chi square values from the Tables 143, 144, and 145 on the surface, these values show a positive correlation but the contents of some of the cells were too small to do any meaningful statistical analysis and hence were left for further considerations.

Table 146

**Cross Tabulation of
School Type by Software Limitations as a Reason for Selecting First Language**

SCHOOL TYPE	YES	NO	ALL
MINORITY	3	1	4
	3.00	1.00	4.00
NON-MINORITY	12	4	16
	12.00	4.00	16.00
ALL	15	5	20
	15.00	5.00	20.00

CHI-SQUARE = 0.000

Table 147

**Cross Tabulation of
School Type by Software Limitations as a Reason for Selecting First Language**

SCHOOL TYPE	YES	NO	ALL
4-YEAR COLLEGE	5	4	9
	6.75	2.25	9.00
UNIVERSITY	10	1	11
	8.25	2.75	11.00
ALL	15	5	20
	15.00	5.00	20.00

CHI-SQUARE = 3.300

Table 148

**Cross Tabulation of
School Type by Software Limitations as a Reason for Selecting First Language**

SCHOOL TYPE	YES	NO	ALL
PUBLIC	7	0	7
	5.25	1.75	7.00
PRIVATE	8	5	13
	9.75	3.25	13.00
ALL	15	5	20
	15.00	5.00	20.00

CHI-SQUARE = 3.590

When school type (4-year college/university), (public/private), and (minority/non-minority) was cross tabulated against opinion of business advisory council of teachers and business members as a reason for selecting the first programming language, since some of the cell values were very small to calculate any meaningful statistical correlation, no further statistical analysis was done and the data was left for further considerations.

Table 149 shows a sample of these crosstabulations when school type was public/private schools. 86% of the public institutions faculty and 85% of the private institutions faculty consider the opinions of business advisory council of teachers and business members when making a selection of their first programming language. The Chi square value = 0.004.

Table 149

**Cross Tabulation of
School Code by Opinion of Business Advisory Council of Teachers and
Business Members as a Reason for Selecting First Programming Language**

SCHOOL TYPE	YES	NO	ALL
PUBLIC	6	1	7
	5.95	1.05	7.00
PRIVATE	11	2	13
	11.05	1.95	13.00
ALL	17	3	20
	17.00	3.00	20.00

CHI-SQUARE = 0.004

Tables 150, 151 shows a crosstabulations of school type (4-year college/university), and school type (minority/non-minority). 88% of the 4-year college faculty make sure that the first programming language must be a structural language that implements modularity, concurrency, and reusable code capability whereas this percentage goes to 100% when university faculty are considered.

When minority institutions faculty were considered, 100% agree that the first programming language must implement modularity, concurrency, and have the capability of producing reusable code and this percentage is 95% when non-minority institutions faculty are considered.

The Chi square values (1.287 & 0.263) show that there might be a positive correlation. But since some of the cell contents were too small to conduct any further statistical analysis and hence it was left for further study.

Table 150

**Cross Tabulation of
School Type by Structural Language That Implements Modularity,
Concurrency, Reusable Code as a Reason for Selecting First Programming
Language**

SCHOOL TYPE	YES	NO	ALL
4-YEAR COLLEGE	8	1	9
	8.55	0.45	9.00
UNIVERSITY	11	0	11
	10.45	0.55	11.00
ALL	19	1	20
	19.00	1.00	20.00

CHI-SQUARE = 1.287

Table 151

**Cross Tabulation of
School Type by Structural Language That Implements Modularity,
Concurrency, Reusable Code as a Reason for Selecting First Programming
Language**

SCHOOL TYPE	YES	NO	ALL
MINORITY	4	0	4
	3.80	0.20	4.00
NON-MINORITY	15	1	16
	15.20	0.80	16.00
ALL	19	1	20
	19.00	1.00	20.00

CHI-SQUARE = 0.263

Type of School and Factors for Selecting a First Programming Language

The purpose of this analysis is to determine if type of a school is a predictor of different factors for selecting a first programming language. There were fifteen factors in the original questionnaire. Cross tabulations were done against all of these fifteen different factors with the type of schools.

Table 152 shows the breakdown of selecting a programming language in the first programming language class because of the job market decisions for the programming language. Almost 56% of the four-year colleges in the survey were selecting the language in their first programming language class because of the job market decisions. Similarly, 55% of the universities were selecting a language in the first programming language class based on the factor of job market decisions. The results show that there was a strong correlation between the job market demands and the selection of a first programming language. The Chi square is 0.002.

When school code (public/private) were cross tabulated with job market as a factor for selecting a first programming language, Table 153 shows that there was a strong positive correlation between the type of a school (public/private) and job market as a factor for selecting a first programming language.

57% of the public schools choose a first programming language by considering job market as a factor in their selection while 62% of the private schools pay attention to job market as a factor when making their selection of a first programming language. The Chi square value is = 0.037.

Table 154 cross tabulates school type (minority/non-minority) and the job market demands as a factor for selecting first programming language. 75% minority institutions faculty consider job market demands as an important factor in selecting a first programming language while 56% of the faculty from non-minority institutions consider job market demands as an important factor in selecting a first programming language. The Chi square value is = 0.469.

Table 152

**Cross Tabulation of
School Type by Job Demands as a Factor for Selecting First Programming
Language**

SCHOOL TYPE	YES	NO	ALL
4-YEAR COLLEGE	5	4	9
	4.95	4.05	9.00
UNIVERSITY	6	5	11
	6.05	4.95	11.00
ALL	11	9	20
	11.00	9.00	20.00

CHI-SQUARE = 0.002

Table 153

**Cross Tabulation of
School Type by Job Demands as a Factor for Selecting First Programming
Language**

SCHOOL TYPE	YES	NO	ALL
PUBLIC	4	3	7
	4.20	2.80	7.00
PRIVATE	8	5	13
	7.80	5.20	13.00
ALL	12	8	20
	12.00	8.00	20.00

CHI-SQUARE = 0.037

Table 154

**Cross Tabulation of
School Type by Job Demands as a Factor for Selecting First Programming
Language**

SCHOOL STATUS	YES	NO	ALL
MINORITY	3	1	4
	2.40	1.60	4.00
NON-MINORITY	9	7	16
	9.60	6.40	16.00
ALL	12	8	20
	12.00	8.00	20.00

CHI-SQUARE = 0.469

Table 155 shows the breakdown of selecting a programming language in the first programming language class by the type of school and the potential of a language for its data structures as a factor in selecting a programming language. Only 22% of the four-year colleges in the survey were selecting the language in their first programming language class because of the potential of its data structures and this percentage is only 18% if the university faculty were selecting a language in the first programming language class based on the potential of data structures of a programming language. The Chi square value is 0.051.

Table 156 cross tabulates school type (minority/non-minority) and the potential for data structures as a factor for selecting first programming language. 25% minority institutions faculty consider language's potential for its data structures as an important factor in selecting a first programming language while 19% of the faculty from non-minority institutions consider language's potential for its data structures as an important factor in selecting a first programming language. The Chi square value is = 0.078.

When school code (public/private) were cross tabulated with language's potential for its data structures as a factor for selecting a first programming language, Table 157 shows the cross tabulation between the school type (public/private) and language's potential for its data structures as a factor for selecting a first programming language. 14% of the public schools choose a first programming language by considering language's potential for its data

structures as a factor in their selection while 23% of the private schools pay attention to language's potential for its data structures as a factor when making their selection of a first programming language. The Chi square value is equal to 0.220.

In all these 3 cases, on the surface by comparing the Chi square values it seems that there is a positive correlation. But some of the cell values were too small to derive any statistical significance and hence no further analysis was done and the issues were left for further study.

Table 155

**Cross Tabulation of
School Type by Potential For Data Structures as a Factor For Selecting First
Programming Language**

SCHOOL TYPE	YES	NO	ALL
4-YEAR COLLEGE	2	7	9
	1.80	7.20	9.00
UNIVERSITY	2	9	11
	2.20	8.80	11.00
ALL	4	16	20
	4.00	16.00	20.00

CHI-SQUARE = 0.051

Table 156

**Cross Tabulation of
School Type by Potential For Data Structures as a Factor For Selecting First
Programming Language**

SCHOOL TYPE	YES	NO	ALL
MINORITY	1	3	4
	0.80	3.20	4.00
NON-MINORITY	3	13	16
	3.20	12.80	16.00
ALL	4	16	20
	4.00	16.00	20.00

CHI-SQUARE = 0.078

Table 157

**Cross Tabulation of
School Type by Potential For Data Structures as a Factor For Selecting First
Programming Language**

SCHOOL TYPE	YES	NO	ALL
PUBLIC	1	6	7
	1.40	5.60	7.00
PRIVATE	3	10	13
	2.60	10.40	13.00
ALL	4	16	20
	4.00	16.00	20.00

CHI-SQUARE = 0.220

Tables 158, 159, and 160 show the breakdown of selecting a programming language in the first programming language class by a school type and the ability of a programming language to create true abstract data types as a factor for selecting the programming language. Almost 22% of the four-year colleges in the survey were selecting the language in their first programming language class because of the ability of a programming language to create true abstract data types. Similarly, 18% of the universities were selecting a language in the first programming language class based on the factor of ability of a programming language to create true abstract data types. The Chi square is 0.051.

When school code (public/private) were cross tabulated with the ability of a programming language to create true abstract data types as a factor for selecting a first programming language, Table 160 shows that 14% of the public schools choose a first programming language by considering the ability of a programming language to create true abstract data types as a factor in their selection while 23% of the private schools pay attention to the ability of a programming language to create true abstract data types as a factor when making their selection of a first programming language. Table 160 shows the cross tabulation with the Chi square value = 0.220.

Table 159 cross tabulates school type (minority/non-minority) and the ability of a programming language to create true abstract data types as a factor for selecting first programming language. 25% minority institutions faculty

consider ability of a programming language to create true abstract data types as an important factor in selecting a first programming language while 19% of the faculty from non-minority institutions consider the ability of a programming language to create true abstract data types as an important factor in selecting a first programming language. The Chi Square value is = 0.078.

In all these 3 cases, on the surface by comparing the chi square values it seemed that there was a positive correlation. But some of the cell values were too small to derive any statistical significance and hence no further analysis was done and the issues were left for further study.

Table 158

Cross Tabulation of School Type by The Ability to Create True Abstract Data Types as a Factor For Selecting First Programming Language

SCHOOL TYPE	YES	NO	ALL
4-YEAR COLLEGE	2	7	9
	1.80	7.20	9.00
UNIVERSITY	2	9	11
	2.20	8.80	11.00
ALL	4	16	20
	4.00	16.00	20.00

CHI-SQUARE = 0.051

Table 159

Cross Tabulation of School Type by Ability to Create True Abstract Data Type as a Factor For Selecting First Programming Language

SCHOOL TYPE	YES	NO	ALL
MINORITY	1	3	4
	0.80	3.20	4.00
NON-MINORITY	3	13	16
	3.20	12.80	16.00
ALL	4	16	20
	4.00	16.00	20.00

CHI-SQUARE = 0.078

Table 160

Cross Tabulation of School Type by The Ability to Create True Abstract Data Types (ADT) as a Factor For Selecting First Programming Language

SCHOOL TYPE	YES	NO	ALL
PUBLIC	1	6	7
	1.40	5.60	7.00
PRIVATE	3	10	13
	2.60	10.40	13.00
ALL	4	16	20
	4.00	16.00	20.00

CHI-SQUARE = 0.220

When school code (public/private) were cross tabulated with job market as a factor for selecting a first programming language, Table 161 shows that there is a strong positive correlation between the type of a school (public/private) and the ability to form good programming habits as a factor for selecting a first programming language. 43% of the public schools choose a first programming language by considering the ability to form good programming habits as a factor in their selection while 31% of the private schools pay attention to the ability to form good programming habits as a factor when making their selection of a first programming language. The Chi square value is = 0.292.

Table 162 shows the breakdown of selecting a programming language in the first programming language class because of the ability to form good programming habits as a factor for selecting the first programming language. Almost 22% of the four-year colleges in the survey were selecting the language in their first programming language class because of the ability of a language to form good programming habits. Similarly, 46% of the universities were selecting a language in the first programming language class based on the factor of the ability of a language to form good programming habits. The resulted show that there was a strong correlation between the job market demands and the selection of a first programming language. The Chi square is 1.174.

Table 163 cross tabulates school type (minority/non-minority) and the ability of a language to form good programming habits as a factor for selecting first programming language. 50% minority institutions faculty consider the

ability of a language to form good programming habits as an important factor in selecting a first programming language while 31% of the faculty from non-minority institutions consider the ability of a language to form good programming habits as an important factor in selecting a first programming language. The Chi square value is = 0.495.

Table 161

Cross Tabulation of
School Type by the Ability of a Language to Form Good Programming Habits as
a Factor for Selecting First Programming Language

SCHOOL TYPE	YES	NO	ALL
PUBLIC	4	3	7
	4.55	2.45	7.00
PRIVATE	9	4	13
	8.45	4.55	13.00
ALL	13	7	20
	13.00	7.00	20.00

CHI-SQUARE = 0.292

Table 162

Cross Tabulation of
School Type by the Ability of a Language to Form Good Programming Habits as
a Factor for Selecting First Programming Language

SCHOOL TYPE	YES	NO	ALL
4-YEAR COLLEGE	7	2	9
	5.85	3.15	9.00
UNIVERSITY	6	5	11
	7.15	3.85	11.00
ALL	13	7	20
	13.00	7.00	20.00

CHI-SQUARE = 1.174

Table 163

Cross Tabulation of
School Type by the Ability of a Language to Form Good Programming Habits as
a Factor for Selecting First Programming Language

SCHOOL STATUS	YES	NO	ALL
MINORITY	2	2	4
	2.60	1.40	4.00
NON-MINORITY	11	5	16
	10.40	5.60	16.00
ALL	13	7	20
	13.00	7.00	20.00

CHI-SQUARE = 0.495

Table 164 cross tabulates school type (public/private) with the compiler cost as a factor for selecting a first programming language, Table 164 shows that there is a strong positive correlation between the type of a school (public/private) and the compiler cost as a factor for selecting a first programming language. Only 29% of the public schools choose a first programming language by considering the compiler cost as a factor in their selection while 54% of the private schools pay attention to the compiler cost as a factor when making their selection of a first programming language. The Chi square value is equal to 1.174.

When school type (4-year college/university) is cross tabulated with the cost of a compiler as a factor in selecting a programming language in the first programming language class. Almost 56% of the four-year colleges in the survey were selecting the language in their first programming language class because of the cost of a compiler. Similarly, 36% of the universities were selecting a language in the first programming language class based on the factor of the cost of a compiler. The results showed that there was a strong correlation between the cost of a compiler and the selection of a first programming language. The Chi square is 0.737.

Table 164

**Cross Tabulation of
School Type by the Compiler Cost as a Factor for Selecting First Programming
Language**

SCHOOL TYPE	YES	NO	ALL
PUBLIC	2	5	7
	3.15	3.85	7.00
PRIVATE	7	6	13
	5.85	7.15	13.00
ALL	9	11	20
	9.00	11.00	20.00

CHI-SQUARE = 1.174

Table 165

**Cross Tabulation of
School Type by the Compiler Cost as a Factor for Selecting First Programming
Language**

SCHOOL TYPE	YES	NO	ALL
4-YEAR COLLEGE	5	4	9
	4.05	4.95	9.00
UNIVERSITY	4	7	11
	4.95	6.05	11.00
ALL	9	11	20
	9.00	11.00	20.00

CHI-SQUARE = 0.737

When school type (public/private) was cross tabulated with compiler availability as a factor for selecting a first programming language, Table 166 shows that there is a strong positive correlation between the type of a school (public/private) and the compiler availability as a factor for selecting a first programming language. 29% of the public schools choose a first programming language by considering the compiler availability as a factor in their selection while 54% of the private schools pay attention to the compiler availability as a factor when making their selection of a first programming language. The Chi square value is = 1.174.

Table 168 shows the breakdown of selecting a programming language in the first programming language class because of the availability of a compiler as a factor for selecting the first programming language. Almost 56% of the four-year colleges in the survey were selecting the language in their first programming language class because of the availability of a compiler. But only 36% of the universities were selecting a language in the first programming language class based on the factor of the availability of a compiler. The results show that there is a strong correlation between the compiler availability and the selection of a first programming language. The Chi square is 0.808.

Table 167 cross tabulates school type (minority/non-minority) and the availability of a compiler as a factor for selecting first programming language. Only 25% minority institutions faculty consider the availability of a compiler as an important factor in selecting a first programming language while 50% of the

faculty from non-minority institutions consider the availability of a compiler as an important factor in selecting a first programming language. The results show that there is a strong positive correlation between the type of a school (4-year college/university) and the compiler availability as a factor in the selection of a first programming language. The Chi square value is = 0.737.

Table 166

Cross Tabulation of
School Type by Compiler Availability as a Factor for Selecting First Language

SCHOOL TYPE	YES	NO	ALL
PUBLIC	2	5	7
	3.15	3.85	7.00
PRIVATE	7	6	13
	5.85	7.15	13.00
ALL	9	11	20
	9.00	11.00	20.00

CHI-SQUARE = 1.174

Table 167

Cross Tabulation of
School Type by Compiler Availability as a Factor for Selecting First Language

SCHOOL STATUS	YES	NO	ALL
MINORITY	1	3	4
	1.80	2.20	4.00
NON-MINORITY	8	8	16
	7.20	8.80	16.00
ALL	9	11	20
	9.00	11.00	20.00

CHI-SQUARE = 0.808

Table 168

Cross Tabulation of
School Type by Compiler Availability as a Factor for Selecting First Language

SCHOOL TYPE	YES	NO	ALL
4-YEAR COLLEGE	5	4	9
	4.05	4.95	9.00
UNIVERSITY	4	7	11
	4.95	6.05	11.00
ALL	9	11	20
	9.00	11.00	20.00

CHI-SQUARE = 0.737

When school type (public/private) was cross tabulated with teaching staff knowledge as a factor for selecting a first programming language, Table 169 shows that there is a strong positive correlation between the type of a school (public/private) and the teaching staff knowledge as a factor for selecting a first programming language. Almost 72% of the public schools choose a first programming language by considering the teaching staff knowledge as a factor in their selection while 62% of the private schools pay attention to the teaching staff knowledge as a factor when making their selection of a first programming language. The Chi square is = 0.196.

Table 170 cross tabulates school type (minority/non-minority) and the teaching staff knowledge as a factor for selecting first programming language. 50% minority institutions faculty consider the teaching staff knowledge as an important factor in selecting a first programming language while 69% of the faculty from non-minority institutions consider the teaching staff knowledge as an important factor in selecting a first programming language. The results show that there is a strong positive correlation between the type of a school (minority/non-minority) and the teaching staff knowledge as a factor in the selection of a first programming language. The Chi square value is = 0.495.

Table 171 shows the breakdown of selecting a programming language in the first programming language class because of the teaching staff knowledge as a factor for selecting the first programming language. Almost 56% of the four-year colleges in the survey were selecting the language in their first

programming language class because of the teaching staff knowledge. But 73% of the universities were selecting a language in the first programming language class based on the factor of the teaching staff knowledge. The results show that there is a strong correlation between the teaching staff knowledge and the selection of a first programming language. The Chi square is = 0.642

Table 169

**Cross Tabulation of
School Type by Teaching Staff Knowledge as a Factor for Selecting First
Programming Language**

SCHOOL TYPE	YES	NO	ALL
PUBLIC	5	2	7
	4.55	2.45	7.00
PRIVATE	8	5	13
	8.45	4.55	13.00
ALL	13	7	20
	13.00	7.00	20.00

CHI-SQUARE = 0.196

Table 170

**Cross Tabulation of
School Type by Teaching Staff Knowledge as a Factor for Selecting First
Programming Language**

SCHOOL STATUS	YES	NO	ALL
MINORITY	2	2	4
	2.60	1.40	4.00
NON-MINORITY	11	5	16
	10.40	5.60	16.00
ALL	13	7	20
	13.00	7.00	20.00

CHI-SQUARE = 0.495

Table 171

**Cross Tabulation of
School Type by the Teaching Staff Knowledge as a Factor for Selecting First
Programming Age**

SCHOOL TYPE	YES	NO	ALL
4-YEAR COLLEGE	5	4	9
	5.85	3.15	9.00
UNIVERSITY	8	3	11
	7.15	3.85	11.00
ALL	13	7	20
	13.00	7.00	20.00

CHI-SQUARE = 0.642

Table 172 cross tabulates the school type and the availability of texts as a factor in selecting a first programming language. The data shows that there is a strong positive correlation between the type of a school (public/private) and the availability of texts as a factor for selecting a first programming language. Almost 72% of the public schools choose a first programming language by considering the availability of texts as a factor in their selection while 62% of the private schools pay attention to the availability of texts as a factor when making their selection of a first programming language. The Chi square is $\chi^2 = 0.196$.

Table 173 shows the breakdown of selecting a programming language in the first programming language class because of the availability of texts as a factor for selecting the first programming language. Almost 78% of the four-year colleges in the survey were selecting the language in their first programming language class because of the availability of texts. But 54% of the universities were selecting a language in the first programming language class based on the factor of the availability of texts. The results show that there is a strong correlation between the availability of texts and the selection of a first programming language. The Chi square is $\chi^2 = 1.174$.

Table 174 cross tabulates school type (minority/non-minority) and the availability of texts as a factor for selecting first programming language. 50% minority institutions faculty consider the availability of texts as an important factor in selecting a first programming language while 69% of the faculty from non-minority institutions consider the availability of texts as an important factor in

selecting a first programming language. The results show that there is a strong positive correlation between the type of a school (minority/non-minority) and the availability of texts as a factor in the selection of a first programming language. The Chi square value is 0.495.

Table 172

**Cross Tabulation of
School Type by Availability of Texts as a Factor for Selecting First Language**

SCHOOL TYPE	YES	NO	ALL
PUBLIC	5	2	7
	4.55	2.45	7.00
PRIVATE	8	5	13
	8.45	4.55	13.00
ALL	13	7	20
	13.00	7.00	20.00

CHI-SQUARE = 0.196

Table 173

**Cross Tabulation of
School Type by the Availability of Texts as a Factor for Selecting First Language**

SCHOOL TYPE	YES	NO	ALL
4-YEAR COLLEGE	7	2	9
	5.85	3.15	9.00
UNIVERSITY	6	5	11
	7.15	3.85	11.00
ALL	13	7	20
	13.00	7.00	20.00

CHI-SQUARE = 1.174

Table 174

**Cross Tabulation of
School Type by the Availability of Texts as a Factor for Selecting First Language**

SCHOOL STATUS	YES	NO	ALL
MINORITY	2	2	4
	2.60	1.40	4.00
NON-MINORITY	11	5	16
	10.40	5.60	16.00
ALL	13	7	20
	13.00	7.00	20.00

CHI-SQUARE = 0.495

Table 175 shows the breakdown of selecting a programming language in the first programming language class because of the hardware availability as a factor for selecting the first programming language. Almost 44% of the four-year colleges in the survey were selecting the language in their first programming language class because of the availability of hardware. But 36% of the universities were selecting a language in the first programming language class based on the factor of the availability of hardware. The results show that there is a strong correlation between the availability of hardware and the selection of a first programming language. The Chi square is 0.135.

Table 176 cross tabulates the school type and the availability of hardware as a factor in selecting a first programming language. The data shows that there is a strong positive correlation between the type of a school (public/private) and the availability of hardware as a factor for selecting a first programming language. Almost 43% of the public schools choose a first programming language by considering the availability of hardware as a factor in their selection while 39% of the private schools pay attention to the availability of hardware as a factor when making their selection of a first programming language. The Chi square is = 0.037.

Table 177 cross tabulates school type (minority/non-minority) and the availability of hardware as a factor for selecting first programming language. Only 25% minority institutions faculty consider the availability of hardware as an important factor in selecting a first programming language while 44% of the

faculty from non-minority institutions consider the availability of hardware as an important factor in selecting a first programming language. The results show that there is a strong positive correlation between the type of a school (minority/non-minority) and the availability of hardware as a factor in the selection of a first programming language. The Chi square value is = 0.469.

Table 175

Cross Tabulation of
School Type by Hardware Availability as a Factor to Select a New First
Programming Language

SCHOOL TYPE	YES	NO	ALL
4-YEAR COLLEGE	4	5	9
	3.60	5.40	9.00
UNIVERSITY	4	7	11
	4.40	6.60	11.00
ALL	8	12	20
	8.00	12.00	20.00

CHI-SQUARE = 0.135

Table 176

Cross Tabulation of
School Type by Hardware Availability as a Factor to Select a New First
Programming Language

SCHOOL TYPE	YES	NO	ALL
PUBLIC	3	4	7
	2.80	4.20	7.00
PRIVATE	5	8	13
	5.20	7.80	13.00
ALL	8	12	20
	8.00	12.00	20.00

CHI-SQUARE = 0.037

Table 177

Cross Tabulation of
School Type by Hardware Availability as a Factor to Select a New First
Programming Language

SCHOOL TYPE	YES	NO	ALL
MINORITY	1	3	4
	1.60	2.40	4.00
NON-MINORITY	7	9	16
	6.40	9.60	16.00
ALL	8	12	20
	8.00	12.00	20.00

CHI-SQUARE = 0.469

Table 178 cross tabulates the school type and the cost of a language as a factor in selecting a first programming language. The data shows that there is a positive correlation between the type of a school (public/private) and the cost of a language as a factor for selecting a first programming language. Almost 86% of the public schools choose a first programming language by considering the cost of a language as a factor in their selection while 54% of the private schools pay attention to the cost of a language as a factor when making their selection of a first programming language. The Chi square is $= 2.031$.

Table 179 shows the breakdown of selecting a programming language in the first programming language class because of the cost of a language as a factor for selecting the first programming language. Almost 67% of the four-year colleges in the survey were selecting the language in their first programming language class because of the cost of a language while 64% of the universities were selecting a language in the first programming language class based on the factor of the cost of a language. The results show that there is a strong correlation between the availability of hardware and the selection of a first programming language. The Chi square is 0.020.

Table 180 cross tabulates school type (minority/non-minority) and the cost of a language as a factor for selecting first programming language. Only 25% minority institutions faculty consider the cost of a language as an important factor in selecting a first programming language while 75% of the faculty from non-minority institutions consider the cost of a language as an important factor in

selecting a first programming language. The results show that there is a positive correlation between the type of a school (minority/non-minority) and the cost of a programming language as an important factor in the selection of a first programming language. The Chi square value is = 3.516.

Table 178

Cross Tabulation of School Type by Cost of a Language as a Factor to Select a New First Language

SCHOOL TYPE	YES	NO	ALL
PUBLIC	6	1	7
	4.55	2.45	7.00
PRIVATE	7	6	13
	8.45	4.55	13.00
ALL	13	7	20
	13.00	7.00	20.00

CHI-SQUARE = 2.031

Table 179

Cross Tabulation of School Type by Cost of a Language as a Factor to Select a New First Language

SCHOOL TYPE	YES	NO	ALL
4-YEAR COLLEGE	6	3	9
	5.85	3.15	9.00
UNIVERSITY	7	4	11
	7.15	3.85	11.00
ALL	13	7	20
	13.00	7.00	20.00

CHI-SQUARE = 0.020

Table 180

Cross Tabulation of School Type by Cost of a Language as a Factor to Select a New First Language

SCHOOL TYPE	YES	NO	ALL
MINORITY	1	3	4
	2.60	1.40	4.00
NON-MINORITY	12	4	16
	10.40	5.60	16.00
ALL	13	7	20
	13.00	7.00	20.00

CHI-SQUARE = 3.516

Table 181 cross tabulates the school type and the features of a language as a factor in selecting a first programming language. Almost 29% of the public schools choose a first programming language by considering the features of a language as a factor in their selection while only 15% of the private schools pay attention to the features of a language as a factor when making their selection of a first programming language. The Chi square is = 0.495.

Table 182 cross tabulates school type (minority/non-minority) and the features of a language as a factor for selecting first programming language. 50% minority institutions faculty consider the features of a language as an important factor in selecting a first programming language while only 13% of the faculty from non-minority institutions consider the features of a language as an important factor in selecting a first programming language. The Chi square value is = 0.808.

Table 183 shows the breakdown of selecting a programming language in the first programming language class because of the features of a language as a factor for selecting the first programming language. Only 11% of the four-year colleges in the survey were selecting the language in their first programming language class because of the features of a language while 27% of the universities were selecting a language in the first programming language class based on the factor of the features of a language. The Chi square is 2.812.

After examining the Chi square values, on the surface, it seems that there is a positive correlation between the type of school (public/private, 4-year college/university, minority/non-minority) and the features of a language as a factor when making the selection of a first programming language, but since some of the cell contents were very small, no further statistical analysis was carried out at this time and the problem was left for further study.

Table 181

Cross Tabulation of
School Type by Language Features as a Factor to Select a New First Language

SCHOOL TYPE	YES	NO	ALL
PUBLIC	2	5	7
	1.40	5.60	7.00
PRIVATE	2	11	13
	2.60	10.40	13.00
ALL	4	16	20
	4.00	16.00	20.00

CHI-SQUARE = 0.495

Table 182

Cross Tabulation of
School Type by Language Features as a Factor to Select a New First Language

SCHOOL TYPE	YES	NO	ALL
MINORITY	2	2	4
	0.80	3.20	4.00
NON-MINORITY	2	14	16
	3.20	12.80	16.00
ALL	4	16	20
	4.00	16.00	20.00

CHI-SQUARE = 2.812

Table 183

Cross Tabulation of
School Type by Language Features as a Factor to Select a New First Language

SCHOOL TYPE	YES	NO	ALL
4-YEAR COLLEGE	1	8	9
	1.80	7.20	9.00
UNIVERSITY	3	8	11
	2.20	8.80	11.00
ALL	4	16	20
	4.00	16.00	20.00

CHI-SQUARE = 0.808

Type of School By the Time Line for Evaluating a Choice of a New First Programming Language

Surprisingly, the method of selecting a new first programming language is not evaluated every semester. The purpose of this analysis is to determine if type of a school is a predictor of different timeliness for re-evaluating the choice of a first programming language. There were seven different timeliness in the original questionnaire. Cross tabulations were done against all of these seven different timeliness with the type of schools.

Table 184 cross tabulates school type (public/private) and the time line of every year to re-evaluate the choice of a first programming language. Every year 86% of the public school faculty re-evaluate their choice of a first programming language. For private schools this percentage goes down to 76%. The Chi square value is = 0.220.

When school type (4-year college/university) is cross tabulated in Table 185, 78% of the 4-year college faculty do their re-evaluation of the selection of a first programming language every year while 82% of the university faculty re-evaluate their selection of a first programming language every year. The Chi square value is = 0.051.

The cross tabulations of school type (minority/non-minority) with the time line of every year for re-evaluating the selection of a first programming language show that 100% of the minority school faculty re-evaluate their choice of a first

programming language every year whereas for non-minority school faculty this ratio is 75%. Table 186 shows the Chi square value is = 1.250. Since some of the cell contents were too small to do any meaningful statistical analysis. Hence, these were left for further study without making any statistical inferences.

Table 184

Cross Tabulation of
School Type by Time Line of Every Year to Select a New First Programming
Language

SCHOOL CODE	YES	NO	ALL
PUBLIC	6	1	7
	5.60	1.40	7.00
PRIVATE	10	3	13
	10.40	2.60	13.00
ALL	16	4	20
	16.00	4.00	20.00

CHI-SQUARE = 0.220

Table 185

Cross Tabulation of
School Type by Time Line of Every Year to Select a New First Programming
Language

SCHOOL TYPE	YES	NO	ALL
4-YEAR COLLEGE	7	2	9
	7.20	1.80	9.00
UNIVERSITY	9	2	11
	8.80	2.20	11.00
ALL	16	4	20
	16.00	4.00	20.00

CHI-SQUARE = 0.051

Table 186

Cross Tabulation of
School Type by Time Line of Every Year to Select a New First Programming
Language

SCHOOL STATUS	YES	NO	ALL
MINORITY	4	0	4
	3.20	0.80	4.00
NON-MINORITY	12	4	16
	12.80	3.20	16.00
ALL	16	4	20
	16.00	4.00	20.00

CHI-SQUARE = 1.250

There are more who re-evaluate their of a first programming language every 2-4 year than those who just want to use the same language in the first programming language classes. Of the 20 faculty surveyed, 11 say that they re-evaluate their choice of the first programming language every 2-4 years and 9 of them do not. When school type (minority/non-minority) is considered, these numbers are 15 for re-evaluating every 2-4 year and 5 are for no change.

Table 188 cross tabulates school type (public/private) and the time line of every 2-4 year to re-evaluate the choice of a first programming language and does not show any correlation between the school type (public/private) and the time line of 2-4 years to re-evaluate the selection of a first programming language. Every 2-4 years none of the of the public school faculty re-evaluate their choice of a first programming language. For private schools this percentage goes up to 69%. The Chi square value is = 6.111.

When school type (4-year college/university) is cross tabulated in Table 187, it shows that there is a strong positive correlation between the type of school (4-year college/university) and the time line of every 2-4 years for re-evaluation of the selection of a first programming language. 56% of the 4-year college faculty do their re-evaluation of the selection of a first programming language every 2-4 years while 54% of the university faculty re-evaluate their selection of a first programming language every 2-4 years. The Chi square value is = 0.002.

Table 189 shows that there seems to be a positive correlation between the school type (minority/non-minority) and the re-evaluation of selecting a first programming language as the need arises without being waiting for a certain time period and the doing the re-evaluation of the selection of a first programming language. But some of the cell contents were too small to conclude any statistical significance. So, this problem needed further analysis.

Table 187

**Cross Tabulation of
School Type by 2-4 Year Time Period to Select a New First Programming
Language (Pascal or Other)**

SCHOOL TYPE	YES	NO	ALL
4-YEAR COLLEGE	5	4	9
	4.95	4.05	9.00
UNIVERSITY	6	5	11
	2.20	8.80	11.00
ALL	11	9	20
	11.00	9.00	20.00

CHI-SQUARE = 0.002

Table 188

**Cross Tabulation of
School Type by 2-4 Year Time Period to Select a New First Programming
Language (Pascal or Other)**

SCHOOL CODE	YES	NO	ALL
PUBLIC	0	4	4
	2.20	1.80	4.00
PRIVATE	11	5	16
	8.80	7.20	16.00
ALL	11	9	20
	11.00	9.00	20.00

CHI-SQUARE = 6.111

Table 189

**Cross Tabulation of
School Type by 2-4 Year Time Period to Select a New First Programming
Language (Pascal or Other)**

SCHOOL STATUS	YES	NO	ALL
MINORITY	3	1	4
	3.00	1.00	4.00
NON-MINORITY	12	4	16
	12.00	4.00	16.00
ALL	15	5	20
	15.00	5.00	20.00

CHI-SQUARE = 0.000

When school type (4-year college/university) is cross tabulated in Table 190, 67% of the 4-year college faculty do their re-evaluation of the selection of a first programming language as need arises while 82% of the university faculty re-evaluate their selection of a first programming language as need arises. The Chi square value is = 0.606.

The cross tabulations of school type (minority/non-minority) with the time line of as need arises for re-evaluating the selection of a first programming language show that 75% of the minority school faculty re-evaluate their choice of a first programming language as need arises whereas for non-minority school faculty this ratio is also 75%. Table 191 shows the Chi square value is = 0.000.

Table 192 cross tabulates school type (public/private) and the time line of as need arises to re-evaluate the choice of a first programming language. As need arises 71% of the public school faculty re-evaluate their choice of a first programming language. For private schools this percentage goes up to 77%. The Chi square value is = 0.073.

The only problem in these crosstabulations was that some of the cell contents were too small to do any meaningful statistical analysis. Hence, these were left for further study without making any statistical inferences.

Table 190

**Cross Tabulation of
School Type by as Need Arises to Select a New First Programming Language**

SCHOOL TYPE	YES	NO	ALL
4-YEAR COLLEGE	6	3	9
	6.75	2.25	9.00
UNIVERSITY	9	2	11
	8.25	2.75	11.00
ALL	15	5	20
	15.00	5.00	20.00

CHI-SQUARE = 0.606

Table 191

**Cross Tabulation of
School Type by as Need Arises to Select a New First Programming Language**

SCHOOL STATUS	YES	NO	ALL
MINORITY	3	1	4
	3.00	1.00	4.00
NON-MINORITY	12	4	16
	12.00	4.00	16.00
ALL	15	5	20
	15.00	5.00	20.00

CHI-SQUARE = 0.000

Table 192

**Cross Tabulation of
School Type by as Need Arises to Select a New First Programming Language**

SCHOOL CODE	YES	NO	ALL
PUBLIC	5	2	7
	5.25	1.75	7.00
PRIVATE	10	3	13
	9.75	3.25	13.00
ALL	15	5	20
	15.00	5.00	20.00

CHI-SQUARE = 0.073

School Type and Different Purposes for Using a First Programming Language

The purpose of this analysis is to determine if type of a school is a predictor of different purposes for using a first programming language. There were eleven different purposes in the original questionnaire. Cross tabulations were done against all of these eleven different purposes with the type of schools.

Table 193 cross tabulates school type (minority/non-minority) with the purpose of language availability for selecting a first programming language. 45% of the faculty take into consideration a fact of availability of a language when making a selection of a first programming language and 55% of them do the selection without thinking about the fact of an availability of a language when making their selection. Only 50% of the minority institutions faculty consider language availability as a purpose when making their selection of a first programming language whereas 44% faculty from the non-minority institutions consider language availability while making the selection of their first programming language. The table shows that there is a strong positive correlation between the type of school (minority/non-minority) and the language availability when making the selection of a first programming language. The Chi square value is = 0.051.

When school type (4-year college/university) was cross tabulated with language availability as a purpose in making the selection of a first programming language, Table 194 shows a positive correlation. 67% of 4-year college faculty versus 27% of the university faculty use language availability as a purpose when selecting the first programming language. The Chi square value is = 3.104.

The cross tabulations of school type (public/private) with the purpose of language availability when selecting a first programming language show that there is a strong positive correlation. Only 29% of the public institutions faculty use the availability of a programming language as a purpose when making their selection. This percentage goes up to 54% when private institutions faculty are considered. Table 195 shows the Chi square value = 1.174.

Table 193

**Cross Tabulation of
School Type by Language Availability as a Purpose to Select a New First
Programming Language**

SCHOOL TYPE	YES	NO	ALL
MINORITY	2	2	4
	1.80	2.20	4.00
NON-MINORITY	7	9	16
	7.20	8.80	16.00
ALL	9	11	20
	9.00	11.00	20.00

CHI-SQUARE = 0.051

Table 194

**Cross Tabulation of
School Type by Language Availability as a Purpose to Select a New First
Programming Language**

SCHOOL TYPE	YES	NO	ALL
4-YEAR COLLEGE	6	3	9
	4.05	4.95	9.00
UNIVERSITY	3	8	11
	4.95	6.05	11.00
ALL	9	11	20
	9.00	11.00	20.00

CHI-SQUARE = 3.104

Table 195

**Cross Tabulation of
School Type by Language Availability as a Purpose to Select a New First
Programming Language**

SCHOOL TYPE	YES	NO	ALL
PUBLIC	2	5	7
	3.15	3.85	7.00
PRIVATE	7	6	13
	5.85	7.15	13.00
ALL	9	11	20
	9.00	11.00	20.00

CHI-SQUARE = 1.174

When school type (4-year college/university) was cross tabulated with parameters and modularity as a purpose in making the selection of a first programming language, Table 196 shows a positive correlation. 78% of 4-year college faculty versus 55% of the university faculty use parameters and modularity as a purpose when selecting the first programming language. The Chi square value is = 1.174.

Table 197 cross tabulates school type (minority/non-minority) with the purpose of teaching parameters and modularity for selecting a first programming language. 65% of the faculty take into consideration a fact of teaching parameters and modularity when making a selection of a first programming language and 35% of them do the selection without thinking about the fact of teaching parameters and modularity when making their selection. Only 25% of the minority institutions faculty consider teaching parameters and modularity as a purpose when making their selection of a first programming language whereas 75% faculty from the non-minority institutions consider teaching of parameters and modularity when making the selection of their first programming language. The table shows that there is a positive correlation between the type of school (minority/non-minority) and the purpose of teaching modularity and parameters when making the selection of a first programming language. The Chi square value is = 3.516.

The cross tabulations of school type (public/private) with the purpose of teaching parameters and modularity when selecting a first programming language show that there is a strong positive correlation. 57% of the public institutions faculty use the teaching of parameters and modularity as a purpose when making their selection. This percentage goes up to 69% when private institutions faculty are considered. Table 198 shows the Chi square value = 0.292.

Table 196

**Cross Tabulation of
School Type by Parameters and Modularity as a Purpose to Select a New First
Language**

SCHOOL TYPE	YES	NO	ALL
4-YEAR COLLEGE	7	2	9
	5.85	3.15	9.00
UNIVERSITY	6	5	11
	7.15	3.85	11.00
ALL	13	7	20
	13.00	7.00	20.00

CHI-SQUARE = 1.174

Table 197

**Cross Tabulation of
School Type by Parameters and Modularity as a Purpose to Select a New First
Language**

SCHOOL TYPE	YES	NO	ALL
MINORITY	1	3	4
	2.60	1.40	4.00
NON-MINORITY	12	4	16
	10.40	5.60	16.00
ALL	13	7	20
	13.00	7.00	20.00

CHI-SQUARE = 3.516

Table 198

**Cross Tabulation of
School Type by Parameters and Modularity as a Purpose to Select a New First
Language**

SCHOOL TYPE	YES	NO	ALL
PUBLIC	4	3	7
	4.55	2.45	7.00
PRIVATE	9	4	13
	8.45	4.55	13.00
ALL	13	7	20
	13.00	7.00	20.00

CHI-SQUARE = 0.292

The cross tabulations of school type (public/private) with the purpose of giving students better understanding of GUI (Graphical User Interface) and other features of work stations when selecting a first programming language show that there is a strong positive correlation. Only 43% of the public institutions faculty use the better understanding of GUI (Graphical User Interface) and other features of work stations as a purpose when making their selection. This percentage goes up to 54% when private institutions faculty are considered. Table 199 shows that there is a strong positive correlation and the Chi square value is = 0.220.

Table 200 cross tabulates school type (minority/non-minority) with the purpose of better understanding of GUI (Graphical User Interface) and other features of work stations for selecting a first programming language. 50% of the faculty take into consideration a fact of better understanding of GUI (Graphical User Interface) and other features of work stations when making a selection of a first programming language and 50% of them do the selection without thinking about the fact of better understanding of GUI (Graphical User Interface) and other features of work stations when making their selection. 50% of the minority institutions faculty consider better understanding of GUI (Graphical User Interface) and other features of work stations as a purpose when making their selection of a first programming language whereas 50% faculty from the non-minority institutions consider better understanding of GUI (Graphical User Interface) and other features of work stations while making the selection of their

first programming language. The table shows that there is a strong positive correlation between the type of school (minority/non-minority) and the purpose of better understanding of GUI (Graphical User Interface) and other features of work stations when making the selection of a first programming language. The Chi square value is = 0.000.

When school type (4-year college/university) was cross tabulated with better understanding of GUI (Graphical User Interface) and other features of work stations as a purpose in making the selection of a first programming language, Table 201 shows a strong positive correlation. 56% of 4-year college faculty verses 46% of the university faculty use better understanding of GUI (Graphical User Interface) and other features of work stations as a purpose when selecting the first programming language. The Chi square value is = 0.202.

Table 199
Cross Tabulation of
School Type by Giving Students Better Understanding of Gui and Other
Features of Work Stations as a Purpose to Select a New First Programming
Language

SCHOOL TYPE	YES	NO	ALL
PUBLIC	3	4	7
	3.50	3.50	7.00
PRIVATE	7	6	13
	6.50	6.50	13.00
ALL	10	10	20
	10.00	10.00	20.00

CHI-SQUARE = 0.220

Table 200
Cross Tabulation of
School Type by Giving Students Better Understanding of Gui and Other
Features of Work Stations as a Purpose for Using a New First Programming
Language

SCHOOL TYPE	YES	NO	ALL
MINORITY	2	2	4
	2.00	2.00	4.00
NON-MINORITY	8	8	16
	8.00	8.00	16.00
ALL	10	10	20
	10.00	10.00	20.00

CHI-SQUARE = 0.000

Table 201
Cross Tabulation of
School Type by Giving Students Better Understanding of Gui and Other
Features of Work Stations as a Purpose for Using a New First Programming
Language

SCHOOL TYPE	YES	NO	ALL
4-YEAR COLLEGE	5	4	9
	4.50	4.50	9.00
UNIVERSITY	5	6	11
	5.50	5.50	11.00
ALL	10	10	20
	10.00	10.00	20.00

CHI-SQUARE = 0.202

Table 202 cross tabulates school type (minority/non-minority) with the purpose of ease of design and structure for selecting a first programming language. 30% of the faculty take into consideration a fact of ease of design and structure when making a selection of a first programming language and 70% of them do the selection without thinking about the fact of ease of design and structure when making their selection. 50% of the minority institutions faculty consider ease of design and structure as a purpose when making their selection of a first programming language whereas only 25% faculty from the non-minority institutions consider ease of design and structure of a language while making the selection of their first programming language. The table shows that there is a strong positive correlation between the type of school (minority/non-minority) and the ease of design and structure of a language when making the selection of a first programming language. The Chi square value is = 0.952.

When school type (4-year college/university) was cross tabulated with the ease of design and structure of a language as a purpose in making the selection of a first programming language, Table 203 shows a strong positive correlation. 33% of 4-year college faculty versus 27% of the university faculty use the ease of design and structure of a language as a purpose when selecting the first programming language. The Chi square value is = 0.087.

The cross tabulations of school type (public/private) with the ease of design and structure as a purpose when selecting a first programming language show that there is a strong positive correlation. Only 29% of the public institutions faculty use the ease of design and structure of a language as a purpose when making their selection. This percentage goes up to 31% when private institutions faculty are considered. Table 204 shows that there is a strong positive correlation and the Chi square value is = 0.010.

Table 202
Cross Tabulation of
School Type by Ease of Design, Structure as a Purpose to Select a New First Language

SCHOOL TYPE	YES	NO	ALL
MINORITY	2	2	4
	1.20	2.80	4.00
NON-MINORITY	4	12	16
	4.80	11.20	16.00
ALL	6	14	20
	6.00	14.00	20.00

CHI-SQUARE = 0.952

Table 203
Cross Tabulation of
School Type by Ease of Design, Structure as a Purpose to Select a New First Language

SCHOOL TYPE	YES	NO	ALL
4-YEAR COLLEGE	3	6	9
	2.70	6.30	9.00
UNIVERSITY	3	8	11
	3.30	7.70	11.00
ALL	6	14	20
	6.00	14.00	20.00

CHI-SQUARE = 0.087

Table 204
Cross Tabulation of
School Type by Ease of Design, Structure as a Purpose to Select a New First Language

SCHOOL TYPE	YES	NO	ALL
PUBLIC	2	5	7
	2.10	4.90	7.00
PRIVATE	4	9	13
	3.90	9.10	13.00
ALL	6	14	20
	6.00	14.00	20.00

CHI-SQUARE = 0.010

The cross tabulations of school type (public/private) with the language which provides job-related skills and is usable in real-world and deals with societal issues as a purpose when selecting a first programming language are shown in Table 205. 86% of the public institutions faculty use a language which provides job-related skills, is usable in real-world and deals with societal issues as a purpose when making their selection. This percentage goes down to 77% when private institutions faculty are considered. Table 204 shows that there is a strong positive correlation and the Chi square value is = 0.010. But since some of the cell contents were too small to make any statistical analysis and hence left for further discussion.

Table 206 cross tabulates school type (minority/non-minority) with the purpose of selecting a programming language which provides job-related skills, is usable in real-world and deals with societal issues. 50% of the faculty take into consideration a fact of teaching a language which provides job-related skills, is usable in real-world and deals with societal issues when making a selection of a first programming language and 50% of them do the selection without thinking about the fact of teaching a language which provides job-related skills, is usable in real-world and deals with societal issues. Only 25% of the minority institutions faculty consider teaching a language which provides job-related skills, is usable in real-world and deals with societal issues as a purpose when making their selection of a first programming language whereas 56% faculty from the non-minority institutions consider teaching a language which provides job-related

skills, is usable in real-world and deals with societal issues while making the selection of their first programming language. The table shows that there is a strong positive correlation between the type of school (minority/non-minority) and the selection of a first programming language which provides job-related skills, is usable in real-world and deals with societal issues. The Chi square value is = 1.250

When school type (4-year college/university) was cross tabulated with the purpose of selecting a language which provides job-related skills, is usable in real-world, and deals with societal issues, Table 207 shows a strong positive correlation. 56% of 4-year college faculty versus 46% of the university faculty use a language which provides job-related skills, is usable in real-world, and deals with societal issues. The Chi square value is = 0.087.

Table 205
Cross Tabulation of
School Type by to Provide Job-related Skills, Usable in Real-world and Deals
with Societal Issues as a Purpose for Using First Programming Language

SCHOOL TYPE	YES	NO	ALL
PUBLIC	6	1	7
	5.60	1.40	7.00
PRIVATE	10	3	13
	10.40	2.60	13.00
ALL	16	4	20
	16.00	4.00	20.00

CHI-SQUARE = 0.220

Table 206
Cross Tabulation of
School Type by to Provide Job-related Skills, Usable in Real-world and Deals
with Societal Issues as a Purpose for Using First Programming Language

SCHOOL TYPE	YES	NO	ALL
MINORITY	1	3	4
	2.00	2.00	4.00
NON-MINORITY	9	7	16
	8.00	8.00	16.00
ALL	10	10	20
	10.00	10.00	20.00

CHI-SQUARE = 1.250

Table 207
Cross Tabulation of
School Type by to Provide Job-related Skills, Usable in Real-world and Deals
with Societal Issues as a Purpose for Using First Programming Language

SCHOOL TYPE	YES	NO	ALL
4-YEAR COLLEGE	5	4	9
	4.50	4.50	9.00
UNIVERSITY	5	6	11
	5.50	5.50	11.00
ALL	10	10	20
	10.00	10.00	20.00

CHI-SQUARE = 0.202

School Type by the New First Programming Language

At this point, we wanted to know if they were going to use a new first programming language. Out of the surveyed population, 65% of the schools were going to use some other language than Pascal as their first programming language and 35% of the participating schools were going to keep on using Pascal in their first programming language classes. Here, we wanted to see if there was any correlation between the type of a school (4-year college/University) and the choice of the new first programming language.

The cross tabulations of school type (4-year college/university) with the selection of another first programming language shows that there is a strong positive correlation. Table 208 shows that 67% of the 4-year college faculty were going to choose some other programming language than Pascal as their first programming language while 64% of the university faculty were going to use some other programming language than Pascal in their first programming language class. The Chi square value is = 0.020.

Out of these 13 schools, 7 schools were very specific about their new first programming language. These schools were going to use either C/C++ or Pascal in their first programming language classes. So, cross tabulations were done with the type of school and the selection of either C/C++ or Pascal as their first programming language.

Table 209 cross tabulates school type (public/private) with the new first programming language (C/C++ or Pascal). It shows that there is a strong positive correlation. 67% of the public schools were going to use C/C++ in their first programming classes while only 50% of the universities going with C/C++ as their new first programming language. The Chi square value is = 0.194.

When school type (4-year college/university) was cross tabulated with the new first programming language (C/C++ or Pascal), it also shows that there is a positive correlation. Only 33% of the 4-year college faculty were selecting C/C++ as their new first programming language while 75% of the university faculty were choosing C/C++ in their new first programming classes. Table 210 shows the Chi square value = 1.215.

Table 208

**Cross Tabulation of
School Type by Another Language to Select as a New First Programming
Language**

SCHOOL TYPE	YES	NO	ALL
4-YEAR COLLEGE	6	3	9
	5.85	3.15	9.00
UNIVERSITY	7	4	11
	7.15	3.85	11.00
ALL	13	7	20
	13.00	7.00	20.00

CHI-SQUARE = 0.020

Table 209

**Cross Tabulation of
School Type by New First Programming Language (Pascal or C/C++)**

SCHOOL CODE	C/C++	PASCAL	ALL
PUBLIC	2	1	3
	1.71	1.29	3.00
PRIVATE	2	2	4
	2.29	1.71	4.00
ALL	4	3	7
	4.00	3.00	7.00

CHI-SQUARE = 0.194

Table 210

**Cross Tabulation of
School Type by New First Programming Language (Pascal or C/C++)**

SCHOOL TYPE	C/C++	PASCAL	ALL
4-YEAR COLLEGE	1	2	3
	1.71	1.29	3.00
UNIVERSITY	3	1	4
	2.29	1.71	4.00
ALL	4	3	7
	4.00	3.00	7.00

CHI-SQUARE = 1.215

Type of School and Student Involvement in the Decision of Selecting a First Programming Language

Surprisingly, there were more who would not like to involve students in the selection of a first programming language than those who simply state that they would seek student input when selecting a new first programming language. Out of the 20 faculty surveyed, 11 say they would not like student involvement, 6 of them would allow students in the selection process while 3 of them said that they may seek student input in the selection. How the type of a school correlates with the student involvement is shown in Tables 211, 212, and 213, which shows statistically significant data.

When school type (4-year college/university) was cross tabulated with the fact that whether students will be involved in the decision of selecting first programming language, Table 211 shows that 44% of the 4-year college faculty versus 18% of the university faculty wanted to let the students involve in the decision of selecting the first programming language. The Chi square value is = 1.626.

Table 212 cross tabulates school type (minority/non-minority) with the fact that whether students will be involved in the decision of selecting first programming language. 25% of the minority faculty as compared to 31% of the non-minority faculty wanted students involved in the decision process of selecting a first programming language. The Chi square value is = 0.060.

Finally, the cross tabulation of school type (public/private) with the fact that whether students will be involved in the decision of selecting first programming language shows that 29% of the public institutions faculty versus 31% of the private institutions faculty would like involve students in the decision of selecting a first programming language. The Chi square = 0.010.

Table 211

**Cross Tabulation of
School Type by Whether Students Will Be Involved in the Decision of Selecting
First Language**

SCHOOL TYPE	NO	YES	ALL
4-YEAR COLLEGE	5	4	9
	6.30	2.70	9.00
UNIVERSITY	9	2	11
	7.70	3.30	11.00
ALL	14	6	20
	14.00	6.00	20.00

CHI-SQUARE = 1.626

Table 212

**Cross Tabulation of
School Type by Whether Students Will Be Involved in the Decision of Selecting
First Language**

SCHOOL STATUS	NO	YES	ALL
MINORITY	3	1	4
	2.80	1.20	4.00
NON-MINORITY	11	5	16
	11.20	4.80	16.00
ALL	14	6	20
	14.00	6.00	20.00

CHI-SQUARE = 0.060

Table 213

**Cross Tabulation of
School Code by Whether Students Will Be Involved in the Decision of Selecting
First Language**

SCHOOL STATUS	NO	YES	ALL
PUBLIC	5	2	7
	4.90	2.10	7.00
PRIVATE	9	4	13
	9.10	3.90	13.00
ALL	14	6	20
	14.00	6.00	20.00

CHI-SQUARE = 0.010

Type of School and Different Ways Students Will be Involved in the Decision of the Selection of a First Programming Language:

There were 5 different ways of student involvement listed in the faculty responses. While doing the cross tabulations, some of the cells had too few values. So, I had to combine these cells to do some meaningful analysis. Tables 214 and 215 show the cross tabulations of the type of school and different ways of student involvement in the selection of a first programming language. There seems to be a positive correlation between the type of a school and the ways of student involvement in the selection of a first programming language.

Table 214 shows the cross tabulations of school type (minority/non-minority) and the ways students will be involved in the decision process of the selection of a first programming language. 75% of the faculty from minority institutions wanted to listen to students and ask them what they want and also ask the students who graduated while 69% of the non-minority institutions faculty wanted to listen to students and ask them what they want and also ask the students who graduated. The Chi square value is = 0.060.

When school type (public/private) was cross tabulated against the different ways students will be involved in the decision of selecting a first programming language, Table 215 shows that there is a strong positive correlation with a Chi square value of 0.010. More than 71% of the public institutions faculty would like to listen to students about their choice and also ask

the students who have graduated about their opinions in the selection of a first programming language and 69% private institutions faculty would like to listen to students about their choice and also ask the students who have graduated about their opinions in the selection of a first programming language.

Table 214

**Cross Tabulation of
School Status by Different Ways Students Will Be Involved in the Decision of
Selecting First Programming Language**

SCHOOL STATUS	ONE WAY	OTHER WAY	ALL
MINORITY	3	1	4
	2.80	1.20	4.00
NON-MINORITY	11	5	16
	11.20	4.80	16.00
ALL	14	6	20
	14.00	6.00	20.00

CHI-SQUARE = 0.060

Table 215

**Cross Tabulation of
School Type by Different Ways Students Will Be Involved in the Decision of
Selecting First Programming Language**

SCHOOL CODE	ONE WAY	OTHER WAY	ALL
PUBLIC	5	2	7
	4.90	2.10	7.00
PRIVATE	9	4	13
	9.10	3.90	13.00
ALL	14	6	20
	14.00	6.00	20.00

CHI-SQUARE = 0.010

Type of School and the Opinion That the Method of Selecting Programming Language is Effective

The purpose of this analysis was to find out if there was any correlation between the type of school and the opinion that the method of selecting a programming language was effective. Tables 216, 217, and 218 show that there is a strong positive correlation.

When school type (public/private) was cross tabulated with the opinion that the method of selecting first programming language was effective, 43% of the public institutions faculty thought that the method was effective while 54% of the private institutions faculty said that the method of selection was effective. Table 216 which contains the Chi square value of 0.006 shows that there is a strong positive correlation.

Table 217 cross tabulates the school type (4-year college/university) and the opinion that the method of selecting first programming language is effective. 67% of the 4-year college faculty were of the opinion that the method was effective while only 36% of the university faculty thought that the method of selection was effective. The Chi square value of 1.818 shows that there is a positive correlation.

The cross tabulation of school type (minority/non-minority) and the opinion that the method of selecting a first programming language is effective shows that only 25% of the minority institutions faculty were of the opinion that the method was effective as compared to 56% of the non-minority faculty said that there method was effective. Table 218 shows the Chi square value = 1.250.

Table 216

**Cross Tabulation of
School Status by the Opinion That the Method of Selecting First Programming
Language Is Effective**

SCHOOL STATUS	NO	YES	ALL
PUBLIC	4	3	7
	3.50	3.50	7.00
PRIVATE	6	7	13
	6.50	6.50	13.00
ALL	10	10	20
	10.00	10.00	20.00

CHI-SQUARE = 0.006

Table 217

**Cross Tabulation of
School Status by the Opinion That the Method of Selecting First Programming
Language Is Effective**

SCHOOL TYPE	NO	YES	ALL
4-YEAR COLLEGE	3	6	9
	4.50	4.50	9.00
UNIVERSITY	7	4	11
	5.50	4.50	11.00
ALL	10	10	20
	10.00	10.00	20.00

CHI-SQUARE = 1.818

Table 218

**Cross Tabulation of
School Status by the Opinion That the Method of Selecting First Programming
Language Is Effective**

SCHOOL TYPE	NO	YES	ALL
MINORITY	3	1	4
	2.00	2.00	4.00
NON-MINORITY	7	9	16
	8.00	8.00	16.00
ALL	10	10	20
	10.00	10.00	20.00

CHI SQUARE = 1.250

Analysis of the Interviews

Introduction

After conducting the faculty survey and carefully examining the faculty questionnaire, it was clear that there were not too many broad questions which dealt with the selection of a first programming language. The faculty interviews gave that chance of asking some broad questions on the selection of a first programming language(s) and the process of selecting the first programming language(s). I also wanted to get a feel for the ACM (Association of Computing Machinery) guidelines and faculty perception of these guidelines. Another part of the interview consisted of faculty responses about the future of programming languages and the selection of a first programming language(s). Finally, the interviews concentrated on the faculty responses about the fact that whether students should be involved in the decision process and if the answer was yes, I wanted to know the ways students will be involved in the above decision process.

A lot of changes took place between the time that BASIC was used as a programming language to introduce students to computers and programming. Then came along FORTRAN as a language for number crunching. Almost all of the engineering schools were using FORTRAN. On the other side COBOL was making its own impact in the business world. But generally, students who were

in the COBOL programming classes already had their exposure to programming using some kind of BASIC. All the earlier versions of BASIC did not teach good programming habits. Structured programming was not born yet! FORTRAN had the same problems of unstructured programming language structures. During these days the main objective of programming was to write programs which worked and gave right answers! COBOL programmers carried the non-structured techniques of BASIC when they were writing COBOL codes. The birth of structured programming came in the 70s when Pascal was designed as a teaching language. Pascal became the favorite of all the computer science departments in the United States. Pascal became the first language which was used mainly because of its structures and its capabilities to teach structured programming techniques. Pascal was at the top for more than 10 years as the most taught first programming language. Heavy demand and the use of fourth-generation languages (4GLs) with the emphasis on object oriented programming in the industry opened up the old wounds of which language should be used as a first programming language. An introduction of Pascal in the high school curriculums and industry demands are forcing colleges and universities to re-think their choice of a first programming language. Are we ready for a change which will let us use a language with the capabilities of object oriented programming in our first programming language classes?

Introductory Computer Science Courses and Programming Language(s) Used

From the point of view of the faculty in the survey population, one of the most important things that happened was that it became easier for everyone to know that the time has come to rethink the selection of a language in their first programming language classes. Three things made this possible. One was the very heavy demand and use of object oriented programming languages in the industry. The demand for people with C/C++ background has been increasing everyday for past couple of years. This heavy demand sometimes forced some schools to make another choice for their first programming course.

The second important thing that happened was that the revised ACM guidelines were just published in September 1991 which were demanding a big change in the course content of CS1 and CS2. Rather than suggesting a fixed number of topics to be covered in either CS1 or CS2, the guidelines gave a list of all the possible modules which one could think of in a programming language course, and left it to the individuals to pick and choose the modules to be covered in their first programming language classes. And as it happened in the past when Pascal dominated the selection during the last revision of ACM guidelines in 1978 during which times schools were changing from BASIC-like languages.

Finally, the third thing was the proliferation of inexpensive and very good turbo personal computer compilers for languages like C and C++. Borland's C/C++ compiler, Microsoft'S C/C++ compilers were among the leaders. The advances in the local area networks also made possible to schools which could not afford to buy huge and expensive main frame computers and compilers. From the interviews, it appears that many among the 4-year college and university faculty elected to use C or C++ in their first programming language classes if they were going to change to a new first programming language. For example, a content analysis of the interviews shows that of the 20 people interviewed, 13, representing 65%, indicated that they were thinking of changing to another first programming language. Out of the 13 faculty, 10, representing 77%, indicated that their new first programming language was going to be either C or C++. Of course, everyone was not sold on the idea of using C/C++ as a first programming language class, as the following excerpt from the transcript reveals:

Q -Why are you using Pascal in your department? As you can see, your neighbors have dropped Pascal and moved to C as their choice of a first language. Why not you?

A - I thought a lot about should we move to C or C++. I would not use it without an add-on. Something to force them into some better habits than C would give you. I think they would get themselves into way too much trouble; I think the instructor would go nuts trying to debug C for students because basically anything would work. You can declare a variable as one thing and use it in a different way. I am sure, I would not want to teach it in the beginning courses.

Q - But, most of the schools are trying to switch to either C or C++ as their choice of a first language.

A - I think, some of these schools are choosing C in their first programming language classes for the reasons of job market demands and hence student demands. But, we are supposed to cultivate these minds who were new to programming, and teach them the concepts of programming, structured programming techniques.

Not all the interviewees were as open in attributing to their selection of a first programming language, but most of them admitted that they will have to switch to a new first programming language soon and it will be either C or C++ unless some other language gets popular. They admitted that there will be basically two reasons for a change: one of the reasons will be because the language they are trying to select is the best language available at that time to teach programming concepts and the other reason will be that they will be forced to a change because of the competition from the other schools in the area and/or job market demands for the people with the knowledge of that language.

Introductory Computer Science Course and Its Content

Another important thing about the first programming language class is that not only that people were discussing about what language they will be changing to but what should be taught in the course. When the new ACM guidelines were released in 1991, the discussion on the contents of first and second programming classes received a special attention. Unlike the previous two sets of guidelines (1968, and 1978), these guidelines did not recommend a specific list of topics to be covered in these classes. Instead 1991 ACM

guidelines created a list of modules, but did not recommend any particular number of modules to be covered in a first programming language class. This meant every one had to pick and choose the modules. Another problem with these guidelines was that the amount of material to be covered if one wanted to include all the modules recommended by the ACM guidelines. There were basically two groups of faculty when asked about the ACM guidelines: One group, who supported the guidelines 100% generally came from larger universities who had hard-core computer science programs and whose programs were accredited while the other group, from smaller-sized four-year colleges whose programs were less hard-core, was not that supportive of these ACM guidelines. Here is how a sample excerpts from the transcripts from each group reveals:

Q - What do you think about the ACM guidelines?

A - You have to be very energetic to get through. I think that's a lot to cover in the first course. I think there are some good ideas, but, you have to look at the difference in a program like here at our school and the program like at ... school. Our program is not the same like their'S. Personally, I could not cover all that stuff they have in the guidelines. Not without making this a four-credit course and I don't know whether that'S what we should do. At ... (school) when you are trying to get a liberal arts education, if you bring people into that major and do that in the first course, I don't think they will be there for the second course! It is overwhelming to look at the amount of material that they expect to be covered. I just don't see how we can do it. Again, I am not saying that all those things should not be covered. But you are back to the whole argument of do you do depth-first or breadth-first.

Q - So, what do you think, will they change the guidelines?

A - You know, I am actually surprised that it has not happened before this, where the major splits out a little more so we just are not majoring in Computer Science but, and I guess, some schools do this but the program is more specialized where you do either software development or you do more hardware development or even in software development, you do more business applications or you do more scientific applications and I think you got split some of that stuff. So, I don't know whether we are not doing some injustice to our students by trying to teach them such breadth!

Q - So, are they going to change sometime soon?

A - No, they will just do the guidelines! We probably look at them for couple of years and see what works and in the next round, they will do the changes again! I bet, within the next 10 years, the guidelines will be dramatically different.

The following excerpt is from the transcript of a faculty who is in the other group who supports and follow the guidelines 100%:

Q - So. are you following the ACM guidelines?

A - ACM guidelines, yes! yes!

Q - Do you agree 100% with the ACM guidelines? Do you do exactly whatever they say?

A - Yes, we do whatever they say. We are ACM accredited! We may not agree with the guidelines but we do whatever they say.

Q - As a personal opinion, what do you think about the ACM guidelines?

A - I have always been on the curriculum committee so I always have to deal with the ACM guidelines and every time we go through our curriculum revision, we go through the ACM guidelines. My personal opinion is that it is better than having no guidelines at all!

Q - ... What do you think will happen to the ACM guidelines with respect to the first programming course in the future?

A - Oh, I don't know. I really don't know much to comment on this. The ACM guidelines, I don't think that they can get any more stricter but they can get looser. If they get too stricter they might lose lots of their supporters. They have to get looser. I think they will move towards more freedom and let you do what you want.

For some, the ACM guidelines were to be followed depending upon certain things. One interviewee said that he would follow the ACM guidelines, but sometimes, they cannot follow them as the following excerpt from the transcript reveals:

Q - What do you think about the ACM guidelines? Do you agree with the ACM guidelines with respect to the first programming language course?

A - Actually, it depends upon the college and quality of students. First of all, you have to see how much the students know and how much capacity they have to learn. You have to see that whether they are ready to learn or not. Then you go up to pointers. If the students are below average or average, then I don't think we should cover up to pointers.

Another interviewee, who was also a computer center director said that in her view, different schools have different purposes in mind when teaching the first programming language class. The following excerpt from the transcript reveals her views about the content of the first programming language class:

Q - What are your thoughts about the introductory programming language course (CS1 course) at your school with regards to the language you use and the content of the course?

A - In that course, really in my view, we need to give basic information to students. I approach the class as we should not just be teaching a language but we should be teaching good concepts and a couple of concepts I work for are proper use of variables: if it is declared as an integer, it should be used as an integer, if it is declared as a character then it should be used as a character those kinds of things which are really basic things. They need to know the difference between sequencing, decision making, looping and subroutines, procedures and functions. When I teach the class, I really stress when a function should

be used, when a procedure should be used. In a function, no side effects, no reading or writing, it should calculate a value, return a value, and not do anything else. If you have to do reads or writes or whatever, you should use a procedure. All local variables, passing everything. Those are concepts that I think they should learn. Then I look at what vehicle we are going to do it with and what we can use it here is Pascal. I look at Pascal not as a language that they going to use when they get out in the real world when working but a language that will force them to do the things I said above.

Based on the above interview and several others, there seemed to be an agreement between the faculty about the content of the first programming language class. They were using the similar guidelines of 1978 which dealt with the content of the course. Only difference was that some of them had changed or in the process of changing to a new programming language.

Future of Introductory Computer Science Courses

Most respondents were thinking about the first programming language course on the guidelines of a change. In response to the question, "What do you think will happen in a couple of years in that course?", Most of the interviewees indicated that they were sure about a new first programming language and at the same time they also had strong feelings when they were describing the content of the course.

It was very interesting to listen to those who had very strong positive feelings toward the current languages used in the first programming languages classes. One such person was a part time computer science faculty teaching the first programming language class offered under the mathematics department who recounted his conversation as follows:

Q - What do you think will happen in a couple of years in that course?

A - Here, at our school or across the country?

Q - Both – at this school and through out the country.

A - I think, people will move away from Pascal. I think they will move to something more recent and I think part of the reason they will change is because of the change in the structure of computer science. Computer Science itself changes so much. The hardware changes. I think that people almost feel compelled that they need to change what they are teaching to stay up because everything else is changing. I don't know what language it will be. I have been to talks where people have suggested number of things. There are those MODULA-2 proponents out there. I was not really happy with MODULA-2 either. I don't know. There are a variety of things. You could look at perhaps to do programming through SQL or you look at some of the other products like I am going to give an off the wall example. If you look at Word Perfect for Windows, there is a macro writing facility. The macro book that fits all the macro commands is an entire programming language: it has loops, it has decision-making; you can call other macros. There is a book just as big as our Pascal text book. There are a variety of things that can be used. I don't know. May be people will go in a lot of different directions. Because we have different options, now.

Q - But, because of the ACM guidelines, is that possible?

A - Yeah, but ACM does not specify what we should teach. They just specify what should be taught.

Q - So, you don't see any future for Object Oriented Programming for the first course?

A - Well, you know, I see a lot of future in an object oriented programming, but in a next couple of years for the first course, I don't know. May be down the road perhaps after 6 - 8 years. But I just don't think the change will happen that fast. I will give you kind of an analogy. People have been saying COBOL has been dead for years. But we still are teaching COBOL, we still are hiring people who are teaching/using COBOL. I think it is going to take a long time before object oriented programming is going to be used in that first course. We have lots of companies that they are not using it, lots of people who are trained in it are not using it. I don't feel adequately prepared to teach a first course in object oriented programming. I need to rethink the way programming is taught if that is what I am going to do. I won't be ready to do it.

In all cases, interviewees were asked, at the outset, to describe their feelings toward the future of the first programming language class, in the hope that responses to the question would draw out succinct statements that would encapsulate their thinking. Some cooperated by offering a phrase or two. For those who did, the term "object oriented programming" was the most recurrent, with seven respondents employing the phrase. The second most common phrase used was "C or C++" given by five people.

Most of the larger universities with enough resources and large student demands were thinking and acting very differently. Rather than offering only one programming language in their first programming language classes, they were implementing a new approach to this change in demand. They were offering different languages in the same class but different sections, and they were putting students with respective backgrounds and majors in those sections.

As one of the interviewee was commenting that it would be a major trend in most of the schools to try to satisfy the demands from various departments and students. In his department they were teaching Pascal and C/C++ at the same time and in the same first programming language classes by offering different sections and appropriately putting students in those sections.

Of the 20 people interviewed, only two people presented strong negative feelings towards the change in the selection of a new programming language. They were teaching in small, private four-year colleges. Their basic claim was that "why should you change just for the sake of change? and "If things are not broken, why fix them?"

After analyzing how the interviewees described their views towards the future of a first programming language classes, we can make the following summarizing observations:

1. Some of the interviewees, mainly from larger universities, were already ready for a change in using a first programming language. They had been using Pascal for more than 10 years as their first programming language. Their basic argument for a change was the job market demands. Also, another reason they were claiming for a change was the new ACM guidelines. Whenever new ACM guidelines were established, it was a signal for these schools to revise their curriculum and to implement the necessary changes.

2. The group of faculty, mainly from a smaller, private four-year colleges, were not ready for this change, not at this time! But they knew that the change was coming in their use of a first programming language. Their argument was based on the following two reasons: one being the competition from other area schools who were going to change the language in their first programming classes by looking at the industry demands. Secondly, the change in the structure of computer science would force them to change. The changes in the hardware, and software would force them to change just to keep up with the technology.

3. There was a third group, very small, actually 2 of them, who were happy with the things they were. They did not see a change was needed at this time and in the near future. They had just changed their curriculums using the 1978 ACM guidelines. The reasons for these schools for not changing at this time were mainly economical. They were trying to adopt to the fast-changing technology as fast as they could. The other reason for not to change, I thought, for these schools was the knowledge of the faculty. For example, language like C++, the faculty would have to learn the language first before they could implement the changes in their curriculum.

Factors Influencing the Selection of a New First Language

There was hardly anyone among the interviewees whose opinion about the reasons for the selection of a new first programming language did not change at some time although some experienced more dramatic changes than others. There were some who qualified the changes in their reasoning by saying, for example, that the only reason they were selecting a language because there was nothing else available or they did not have a choice, or they were told to select that language. But, they were using different reasons, for example, the features of a language was one of the most important reasons for them to select a language. The availability of hardware and software at affordable prices had made the above reasoning more easy. Specially, the prices of language compilers on PCs were making it easy for schools with very limited budgets for their software purchases to buy these new language compilers.

In this category also, I saw some grouping amongst the interviewees. The larger group who came mainly from research-oriented larger universities were changing to a new language which provided them object-oriented features for one and only one reason which was the industry demands! The job market

demands for people with C/C++ backgrounds had forced these schools to change their selection of a first programming language to C/C++ from Pascal.

This argument is revealed neatly in the following excerpt from one of the transcripts:

Q - So, do you pay close attention to the industry demands and job market requirements in making the selection of a first programming language?

A - Yes, we pay very close attention to the job market demands! We want to make sure that our students when graduated should have a chance of 100% in the job market. Being in the vicinity of RTP, we are fortunate enough to look at the demands of these companies very closely. And they also count on our help of providing students who are going to be ready for work once they are hired. Right now, industry is in a heavy demand for people with C and C++ backgrounds, and that is why we switched to C++ one year ago.

There was an interesting response from one of the faculty, who was from a small, private four-year college on the same question. They were using Pascal in their first programming language classes and were located very closed to RTP (Research Triangle Park) area. This is how the conversation was recorded:

Q - So, was there any thought about the job market demands when making the selection of a new first programming language?

A - No, and I think part of the reason that we have not talked a lot about that is because all of our graduates get jobs, and they get good jobs and most of the time what will happen is if you look at our graduates, we don't have most of our graduates working in any one language. It's not like 90% of our people work with C because they don't and I think if that would happen we might look at something else. But our students, our graduates work in such a variety of areas that most of the time what happens is our graduates will get hired and they will go for two-three days of training in whatever they are going to work in and they make adjustment very quickly because they have a good foundation.

When asked about the difficulty in teaching and learning C or C++ as a first programming language, almost everyone in the group of 20 had an unanimous response shown by the following sample excerpts from the transcripts:

Q - What about students? Are they comfortable using C++ as their first programming language?

A - No, it is difficult. But students are more sophisticated and they are ready to learn because it is more difficult. If we make it too easy for them, they may not take it more seriously. In fact, our Pascal course was the harder course. But C++ is a challenge course and they seem to like a challenge, That is really not a problem. They work hard. I know, they complain, they always complain, but finally they are gaining more knowledge.

Q - Lots of schools are going for C right now. The reason they want to use C is not because they want to teach the language, but because of the heavy demand from the industry, What do you think?

A - It is good and in a production environment, it is probably what a lot of our students are going to use it but I don't think that it is what we need to use in the first course. Now, I agree that I will look happy should they use C with data structures as an example or some higher level course. But I am not really comfortable with C in the beginning. I don't think it will install good habits for the programmers. I am not that concerned after that first course they can get that many jobs using what they learned. You know, what are we doing? Are we teaching a skill how to write a program in that particular language or are we teaching good habits, are we making good programmers? And I hope what we are doing is making good programmers!

The preceding discussion dealt with the selection of C or C++ as a new first programming language and the reasons behind the selection. In summary, we can make the following observations:

1. Of the 20 interviewees, no one was saying nice things about C from the students point of view. Everyone agreed that it was not a good language to be taught in that first programming language class. Everyone agreed that C++ was not an easy language to be used in that class. Everyone claimed that Pascal was the best designed language to be used in teaching in a first programming language class.

2. But, we were not going with our instincts. Some other outside forces were guiding our selection process. We all knew that the selection we were making was going to hurt students but we still were going ahead with our choices!

3. We, educators, have a big problem. Are we teaching the students for the sake of jobs or are we teaching students for the sake of giving knowledge? I am sure, we are supposed to do both of these things in the best possible manner and at the best possible times.

Selection Process of a First Programming Language

The initial decision of changing to a new first programming was certainly the beginning of a process of change for most of the interviewees, and so it was interesting to see their process of selecting a first programming language.

In this regard, the common theme was a group of faculty would get together and decide what was their next first programming language. There was not a single common process which was used in the selection process. The selection process depended upon the type of school, size of school, the area in which school was located, and the type of the program in which the first programming language class was offered.

After talking with all of the 20 interviewees, one thing which was common to all of these schools' selection process was the students' involvement in the decision process. Depending upon how they handle the first programming class it could decide their future in a sense that they might stay in the major or change to some other major where there is no programming involved.

So, this was a very important decision in the students' education, I was wondering about the interviewees thought about student involvement in the above decision process. The following excerpts from different transcripts reveal their opinions:

Q - Now, when you select a language for the first course here, was there a selection process? How did you decide to use Pascal?

A - I inherited Pascal! You know, I taught at ... before I came here, and I also inherited it there. Since I have been here, we talked about should Pascal be the one? And every time , we are coming up with the same conclusion that it works well for us. What I would much rather see happen is more of an emphasis is put on a laboratory along with the course, so that students get lots of practice with it. I am more concerned about the amount of practice they have than should we change to something else or not. Because, I think, Pascal is working okay.

Q - Most of the schools, I think, when they select a language they never ask students but only ask faculty. Should we involve students in deciding when we change a language? Should we get the input from students?

A - Well, I think, what will be important is to get some comments from people who have gone through our program. I don't know, people coming in really do not have any basis to make a decision. I always think it is funny, when we ask a student if that was a good text book for the course they have just gone through. Because they don't have anything to judge it on. They never used another text book. They never seen anything else, and I think it is the same thing if you ask a student who has never used any other language that if this is a good language. Well, they don't have anything to compare it to. I like to go back and survey our graduates ...

Q - Those who have gone through your program, like seniors ...

A - Yes, exactly right! Well, I would really like to ask people who are in the job market and to see what they think. Talk to some people who are programming in C, we know a number of people who are writing application software using C, like we have students working with IBM writing application software using C, we have a number of COBOL programmers, and see what they would say. Now, they are in the job market and they have been through our program. They could tell us that.

One of the interviewee from a large university with a large number of undergraduates in the computer science department had this to say on the same subject:

Q - How do you select a language for the first course? Do you have a group or any one person? Who decides it?

A - Oh, that was a terrible, terrible, long and agonizing process! I have got probably a fourth of file cabinet full of documentation, electronic mail back and forth from all the faculty in the Computer Science department: ones that wanted and ones that didn't. Lots of different newspaper articles, lots of e-mail messages! We had e-mail war for almost an year and half on this subject and we have been agonizing over it for, may be, five years now before we decided to switch.

Q - So, was there a split between the group?

A - The main split was that people did not want to have to learn anything new.

Q - You mean the faculty?

A - Yes, the faculty! They are too busy with their agenda! You know, every one has their own agenda! That means they have to spend their energy. All the faculty do not know C++. There are 3 or 4 people who know it and the rest of them don't really want to learn it so they teach upper-level classes. Most of them say that if it is not broken why fix it. It is a lot of attitudes, don't bother me type.

Q - But, we have to keep up with the changing technology, so they will have to change. One thing you did not mention are the students. Do you take students' views into account when you select a language? They may not be current students but who have graduated and are in the work force.

A - We do have some surveys that did include students. And students, when they start are not capable of giving some significant input. One of our nationally well-known faculty, he wasn't concerned because he was mostly doing only graduate-level courses until his son came here as a freshman. When his was trying to get a job during the summer, everybody wanted him to know C or C++. Then, this faculty member came to me and asked: "what is this? why are we teaching Pascal? Why are not we teaching C or C++?" and he had never been on the conversation before. Now, certainly he is one of the main pushers for C++ because it was very important for him.

After analyzing how the interviewees described their selection process of a first programming language, we can make the following summarizing observations:

1. Some of the interviewees did not think that students could give meaningful input in the selection process. At some point they might be right. For example, students who did not have enough programming experience probably could not add anything important to the selection process. Actually, one of the interviewee had an interesting notion about this. She always thought

it was funny when she used to ask a student if that was a good text book for the course they had just gone through. Because they didn't have anything to judge it on. They had never used another book. They had never seen anything else and she thought it was the same thing if she asked a student who had never used any other language that if that was a good language. Well, they didn't have anything to compare it to.

2. The selection process has been changing with the times. There were supporters of the thought of students involvement in the decision process. If the main goal of the selection process is to help students in their education process and make them more marketable when they graduate, we have to ask them what they need.

So, we should ask our graduates who have been in the job force for a while about their thoughts in the selection process of a new first programming language.

3. Rather than making the selection just by the faculty and the graduates of the school, we should have industry representatives involve in the decision process which will in turn help the school and the graduating students of the school.

CHAPTER V

CONCLUSIONS AND RECOMMENDATIONS

The purpose of this study was to focus on the selection of a first programming language in introductory computer science classes. The main objective was to identify the choice of first programming language, reasons behind the selection, factors involved in the decision process, and differences in reasons for choosing a particular programming language among groups of students and faculty. Students and faculty from twenty four-year colleges and universities in North Carolina involved with the first computer science course were chosen as subjects. Two instruments were used for data collection. One was a survey administered to 322 students and 20 faculty members, and the other was an open-ended interview with those 20 faculty members.

Summary of Findings

In summary, the survey of students from the twenty four-year colleges/universities showed that the language with which they had the most experience was Pascal, and it was the heavily used language, whereas BASIC took the number two place, followed by COBOL and C/C++.

The survey also showed that these students' first experience with programming was BASIC followed by Pascal, followed by COBOL and FORTRAN.

The top three reasons for learning these languages were job market demands, someone's advice, and popularity of the language at that time.

If the students were given another chance of learning a first programming language all over again, the survey showed that their number one choice would be Pascal – and not BASIC – followed by C/C++, BASIC, and COBOL. The top three reasons for this selection were: the language was used in other computer science courses, they wanted to learn the language, and it was an easy language to learn. Examination of some of the variables as possible predictors of these languages suggests the following:

1. The choice of a first programming language did not depend upon the type of school (four-year college/University, Minority/Non-Minority, and Public/Private).
2. Perceived popularity of a language was strongly related to the type of school.
3. Language choice by some authority person was strongly related to the type of school.
4. Perceived easiness of learning the language was strongly related to the type of school.
5. Job market demand was strongly related to the type of school.

6. If the students were given another chance to select a new first programming language, selection of a new first programming language a student was related to the type of a school (four-year college/University), but when type of school (Public/Private) or (Minority/Non-Minority) were considered, one could not relate a selection of new first programming language.

In summary, the survey of faculty at twenty four-year colleges/universities showed that Pascal was the most widely taught first programming language, and C/C++ was the number one choice by most faculty who planned to change to a new first programming language.

The four most important factors which played a major role in the faculty selection of a first programming language were: good programming habits, job market demands, structure and design of a language with good data structures, and hardware/software availability. At the same time, the least important factor in making the selection was availability of a knowledgeable faculty.

The survey pointed out that a majority of faculty surveyed re-evaluate their choice of a language either every 2-4 years or as the job market demands. Further, the survey showed that most of these faculty would like to involve their students in the decision process of selecting a new first programming language in some way and upto certain extent.

The survey showed that private schools consider feedback of their industry advisory council while making the selection of a new first programming language, which in turn helps the job market demands by appropriately making the right selection. Examination of certain variables as possible predictors of the first programming language suggests the following:

1. Compiler cost, compiler availability, teaching staff knowledge, hardware availability, and cost of a language were strongly related to the selection of a first programming language.

2. Job market demand was strongly related to the type of a school when the selection of a first programming language was made.

3. Ability of a language to form good programming habits was strongly related to the type of a school.

4. Availability of a language was strongly related to the type of a school.

5. Modularity, parameters, ease of design, and structure of a language were strongly related to the type of a school.

6. A language which provides job related skills, usable in the real world, and deals with societal issues was strongly related to the type of a school.

7. Selection of a new first programming language was strongly related to the type of a school.

8. Students' involvement in the decision process of selecting a new first programming language and the type of school were strongly related.

9. Students and faculty agree on almost all things surveyed, including the language itself and the reasons behind the selection of a first programming language.

Follow-up interviews of the twenty faculty yielded very interesting and useful data to augment the conclusions derived from the survey. A lot of information was anecdotal, but informal content analysis showed certain noteworthy trends.

It appears from the interviews that in the last couple of years, 65% percentage of the faculty had been thinking about changing to a new first programming language. The publication of new ACM guidelines (1991), introduction of user-friendly but inexpensive compilers, heavy demand from industry, and extra-ordinary advances in the technological field – all led them to believe that they have found the replacement for Pascal, and it is either C or C++. 71% of them are leaning towards C/C++ and claiming that object oriented programming is going to stay here!

It was also clear from the interviews that larger universities will try to follow these new ACM guidelines, whether they like it or not and whether the students can handle the material or not, while four-year colleges will try to keep close to these universities in following the new ACM guidelines. These colleges will try to pay more attention to students and see if they can handle all that material or not.

The basic difference seems to be whether one should teach structured programming and not worry about industry demands or worry about industry demands and forget about teaching structured programming when teaching the first programming language classes.

Analysis of the interviews also showed that there were fundamental differences among the faculty's thoughts about the future of their first programming language classes. Of course, they did agree on one thing, which was that there will be new first programming language within a year. But that is where the agreement ended. Some claimed that object oriented programming will be the standard of industry when making the selection of a new first programming language, while there were a few who were thinking about some other Pascal-like structured language. Then there were the supporters of C/C++. Some of them were planning to use C with some add-ons which will force structured programming habits to be used in the first programming language class. And finally, there were some who claimed that there will be more than one first programming language used in the same class and that they would offer different sections under the same name and appropriately put students in the right sections depending upon their need, background, and major requirements. In considering these choices, the last one might be possible in larger universities where there will be large student demand and enough resources to satisfy it.

Finally, the analysis also showed that in the larger universities, the selection process is done by the group of faculty in the department and is mainly driven by job market while in a smaller schools the process typically involves an advisory council of professionals from the industry and a very small number of faculty in the department.

General Recommendations

The results of the study can be used in a couple of ways. One way is to use the findings as guidelines for designing and/or revising curriculums in Computer Science Education. Other colleges and universities might benefit, perhaps also in developing nations in the third world who are trying to design computer science curriculums.

The other way to use the findings is to point out areas for further investigation. For example, the study can be replicated, using populations other than from North Carolina four-year colleges/universities. Also, other possible predictors of first programming language selection may be examined.

Recommendations for Four-Year Colleges and Universities

On the basis of findings in both student and faculty surveys and faculty interviews, the following recommendations are offered for the four-year colleges and universities studied and for other institutions who may see usefulness for them:

1. Continue the practice of offering first programming language classes on the assumption that the students taking these classes do not know how to program. This will enable those who come with some programming backgrounds without right kind of notions about programming, for example, no use of internal documentation, use of GOTOs and whose main goal is as long as the programs work and give right answers they do not need to worry about anything else to learn good programming habits. It is almost like wiping the slate clean.

2. Do not make job market demands as the main thing in the selection process of a first programming language. This means to re-prioritize goals in teaching the first programming class. Our main goal in the first programming class should be to teach structured programming concepts which is supported by the ACM guidelines and to teach these concepts we need a programming language as a vehicle. So, we must make sure that the language we are trying to use to satisfy that goal must do that easily.

3. Try to follow the ACM guidelines as closely as possible. This will help smaller schools keep up with the larger universities. We have to keep in mind, however, that these are just guidelines. They do not specify what we should we teach; they only specify what should be taught. So, it is possible to adjust these according to the needs of students. Our first priority in teaching a first programming language is to teach good programming habits and to create good programmers. Our immediate and main concern in teaching a first programming language class should be students and nothing else including the ACM guidelines, job market demands!

4. Keep on revising computer science related curriculums every 2-4 years. But keep in mind that revision does mean we have to change everything from top to bottom in the curriculum. If something is working nicely, do not try to change it. It is like the saying "if it is not broken, don't fix it!" So, if a programming language used in a first programming language classes is producing the expected results, for instance, it is teaching good programming habits and producing good programmers, do not hurry in changing the language just for the sake of change!

5. Utilize the expertise in the nearby industry to design an advisory council whose job will be to keep an eye on the industry demands and students' needs and try to keep a balance between them. This will help both industry and schools. The industrial community will think that we, as educators, are trying to seek their advice to satisfy their needs.

And, it will help the school by producing marketable graduates who will get immediate jobs after their graduation. Also, it will be a good community involvement which in turn helps our students.

6. Make students' input part of the evaluation process when revising the computer science curriculum. Graduates who have joined the work force should be surveyed about their experiences. They will be our direct contact to the job force who will keep on guiding us to the right track so that our future graduates will follow their path of success in getting the good jobs. Also, being the alumni, they will continue to feel a part of the school.

7. Always involve faculty and related staff on the campus when making the selection of a new first programming language(s). This will help the computer science department satisfy needs of students. Finally, it will unite the whole campus and no one will feel left out and everyone will feel important and think that their opinion counts.

Suggestions for Further Research

Some areas for further investigation were suggested by the student and faculty surveys, and others were suggested by the faculty interviews:

1. Replicate the survey, using the same faculty sample, but other student populations, perhaps including all the seniors in the Computer Science related majors in North Carolina four-year colleges and universities.

2. Replicate the survey, using other student populations, perhaps including all students who have graduated with Computer Science and related majors and are now using some language for programming in their jobs.

3. Replicate the survey, using other populations, perhaps including all four-year colleges and universities throughout the United States.

4. Compare the results of colleges and universities from industrial states with those of their counter parts in non-industrial states.

5. Compare the attitudes of Community College faculty about the first programming language used with those of their counter parts in four-year colleges and universities.

6. Use as dependent variables the choice of either first or second programming languages to define general success of students in Computer Science and related majors. For example, a study might describe or measure the "success" of the graduates in their jobs after graduation.

7. Use as dependent variable the choice of a first programming language to define general success in the enrollments of Computer Science and related majors. For example, a study might describe or measure the enrollment trend towards the Computer Science after their success/failure in first programming language classes. In the past 15 years, incredible progress has been made in Computer Technology and Software Engineering.

Almost every four-year college and university is trying to offer some programming classes and other Computer Science related courses. Since the first programming language is often the basis for these classes, we have to make sure to be very careful in choosing the first programming language so that students, faculty, and schools are successful in their respective missions of learning, teaching, and offering knowledge. This study of factors influencing adoption of a first programming language in introductory computer science courses in North Carolina may not only help other schools in the United States but also those schools in other countries who are following in the footsteps of these schools in designing their Computer Science related curriculum.

APPENDIX A

LIST OF ALL 4-YEAR COLLEGES AND UNIVERSITIES IN NORTH CAROLINA

Appalachian State University
Atlantic Christian College
Barber-Scotia College
Belmont Abbey College
Bennett College
Campbell University
Catawba College
Davidson College
Duke University
East Carolina University
Elizabeth City State University
Elon College
Fayetteville State University
Gardner-Webb College
Greensboro College
Guilford College
High Point College
Johnson C. Smith University
Lenoir-Rhyne College
Livingstone College
Mars Hill College
Meredith College
Methodist College
Mount Olive College
North Carolina Agricultural And Technical State University
North Carolina Central University
North Carolina State University
North Carolina Wesleyan College
Pembroke State University
Pfeiffer College
Queens College
Saint Andrew's Presbyterian College
Saint Augustine's College
Salem College
Shaw University
University of North Carolina at Asheville
University of North Carolina at Chapel Hill
University of North Carolina at Charlotte
University of North Carolina at Greensboro
University of North Carolina at Wilmington
Wake-Forest University
Warren Wilson College
Western Carolina University
Wingate College
Winston-Salem State University

APPENDIX B

STUDENT SURVEY QUESTIONNAIRE

1. You are a

- Freshman
- Junior
- Sophomore
- Senior
- Non-Degree

2. You are in the age group of

- 18 - 22
- 23 - 27
- 28 - 32
- 33 - 37
- 38 - 42
- 43 - 47
- 48 - 52
- 53 - 57
- 58 - 62
- 63 - above

3. You are a

- Male
- Female

4. The number of computer courses taken:

- 1
- 2
- 3
- 4
- More than 4

5. What is your major (if declared)?

If your major is not declared, what is your prospective major?

6. Rate your experience with the following programming languages you know (leave blanks for others you don't know) according to the following scale: 1 = Expert, 2 = Advanced, 3 = Fluent, 4 = Familiar, 5 = Novice, 6 = Never used, 7 = No knowledge whatsoever

- Ada
- APL
- Assembly/Machine
- BASIC
- C
- C++
- COBOL
- Ed-Scheme
- Forth
- FORTRAN
- HYPERTALK
- LISP
- Logo
- Modula-2
- Pascal
- PL/I
- PROLOG
- SmallTalk
- Turing
- Other (fill-in) -----

7. Which programming language(s) do you use most to do your programming?(Check)

- Ada
- APL
- Assembly/Machine
- BASIC
- C
- C++
- COBOL
- Ed-Scheme
- Forth
- FORTRAN
- HYPERTALK
- LISP
- Logo
- Modula-2
- Pascal
- PL/I
- PROLOG
- SmallTalk
- Turing
- Other (fill-in) -----

8. Which was the first programming language you learned?

How long ago?

_____ years _____ months

Why? (Check all that apply)

- The language was used in the first course in the department.
- Wanted to learn that language.
- Someone told you to take that language course.
- That was the only language course available for you at that time.
- It was a popular language at that time.
- There were lots of jobs available at that time for those people who knew that language.
- That was the only language offered by the school.
- It was a required language for the major.
- It was the easy language.
- Other (fill-in) _____

9. Which was the second programming language you learned?

How long ago?

_____ years _____ months

Why? (Check all that apply)

- The language was used in the second course in the department.
- Wanted to learn that language.
- Someone told you to take that language course.
- That was the only language course available for you at that time.
- It was a popular language at that time.
- There were lots of jobs available at that time for those people who knew that language.
- That was the only language offered by the school.
- It was a required language for the major.
- It was the easy language.
- Other (fill-in) _____

10. If you could do it all over again, which programming language would you want to learn first?

Why?

APPENDIX C

FACULTY SURVEY QUESTIONNAIRE

1. Please provide the name of the department in which you work.

2. How many years have you been affiliated with this department?

3. Which programming language is taught/used in Introductory Computer Programming Language Course (CS1)? (Check)

- Ada
- APL
- Assembly/Machine
- BASIC
- C
- C++
- COBOL
- Ed-Scheme
- Forth
- FORTRAN
- HYPERTALK
- LISP
- Logo
- Modula-2
- Pascal
- PL/I
- PROLOG
- SmallTalk
- Turing
- Other (fill-in) _____

4. Which programming language is taught/used in Advanced Computer Programming Language Course (CS2)? (Check)

- Ada
- APL
- Assembly/Machine
- BASIC
- C
- C++
- COBOL
- Ed-Scheme
- Forth
- FORTRAN
- HYPERTALK
- LISP
- Logo
- Modula-2
- Pascal
- PL/I
- PROLOG
- SmallTalk
- Turing
- Other (fill-in) _____

5. How does the programming language chosen in introductory computer science courses relate to programming languages used in other computer science courses?

6. What is the decision process for choosing the first programming language to be used in introductory computer science courses? Who is involved in making this decision?

7. What factor(s) played a major role in the above decision?

8. How often is the decision regarding choice of a programming language reevaluated?

9. What, in your opinion, is the purpose of using this programming language in introductory computer programming course (CS1) ?

10. In your opinion, how would you rate the effectiveness of your department's ability to accurately select a programming language for introductory computer science classes? (Check)

-- Very effective

-- Effective

-- Neutral

-- Ineffective

-- Very Ineffective

-- Uncertain

11. Estimate the percentage of the course content spent:

- a. Flow-Charts or Pseudocode _____ %
- b. Assignment statements and Variables _____ %
- c. Conditional Statements _____ %
- d. Loops _____ %
- e. Procedures and Functions _____ %
- f. Parameters _____ %
- g. Input/Output _____ %
- h. Data Structures _____ %
- i. Other (fill-in) _____ %

12. What percentage of course time is spent in each delivery method category:

- a. Lecture _____ %
- b. Hands-on/Lab _____ %
- c. Discussion _____ %
- d. Other (fill-in) _____ %

13. In your opinion, which factors are most important for choosing a programming language for CS1? (check all that apply)

- Hardware Availability _____
- Software Availability _____
- Language Features _____
- Cost _____
- Job Market _____
- Other _____

14. Are you planning to change to another computer language (for CS1)?

Yes —— No ——

If Yes: (a) What Language?

(b) Please state reason(s) for change.

When do you expect to change another computer language (for CS1)?

15. Will students involved in the decision process?

Yes —— No ——

If Yes, how?

16. Other Comments?

APPENDIX D

CONSENT FORM FOR FACULTY

Dear fellow Computer Programming I teacher:

My name is Lal T. Shimpi, a professor at Meredith College and a doctoral student at the University of Massachusetts School of Education in Amherst, Massachusetts. I am asking you to be one of the participants in a research project, which is exploratory study of programming languages used in introductory computer programming classes in schools from North Carolina. The purpose of this study is to find out the factors involved in selecting a programming language and students' views about the languages used in introductory computer programming classes (CS1).

The research procedure will consist of a 15 to 20 minute interview in person or by telephone. There is a questionnaire for the faculty who is teaching this introductory computer programming class which will take 10 minutes at the most to complete which will be mailed through e-mail or through first class mail with postage-paid return envelope included for the prompt response. The other questionnaire is for the students which will take 10 minutes to complete which will be hand-delivered and collected at the same day by myself.

I will be calling you to obtain your consent to participate and setup a convenient time to interview you. I want to stress that your participation is voluntary and you are free to withdraw your consent without prejudice to the study.

This study is anonymous, and published results will make no reference to your name. However, I will very likely give descriptions that may be unique to you such as place of work, discipline area and history of involvement with programming languages. During the interview, which is qualitative, feel free to tell me what to keep off the record and I will honor your request. For the purpose of ensuring a good report and analysis, I will request that our interview be audio taped.

If you have any questions with the research procedure, please contact me through any one of the following addresses:

Home:
2401E Still Forest Place
Raleigh, NC 27607
(919) 787-8609

Office:
Meredith College
Dept of Maths & Computer Science
3800 Hillsborough Street
Raleigh, NC 27607
(919) 829-8614

e-mail address:
laichand@ecsvax.unc.edu

To indicate your consent to participate in this project, kindly sign below and mail this form, using the enclosed self-addressed stamped envelope.

Name of the Faculty: _____

Telephone: _____

Name of the class teaching: _____

Number of students in the class: _____

Today's date: _____

Signature: _____

APPENDIX E

CONSENT FORM FOR STUDENTS

Dear Computer Programming I student:

My name is Lal T. Shimpi, a professor at Meredith College and a doctoral student at the University of Massachusetts School of Education in Amherst, Massachusetts. I am asking you to be one of the participants in a research project, which is exploratory study of programming languages used in introductory computer programming classes in schools from North Carolina. The purpose of this study is to find out the factors involved in selecting a programming language and students' views about the languages used in introductory computer programming classes (CS1).

The research procedure consists of a completing a questionnaire which will take 5-10 minutes to complete and will be collected at the end of the class.

I want to stress that your participation is voluntary and will not affect your course grade whether you participate in the study or not. This study is anonymous, and published results will make no reference to your name.

If you have any questions with the research procedure, please contact me through any one of the following addresses:

Home:
2401E Still Forest Place
Raleigh, NC 27607
(919) 787-8609

Office:
Meredith College
Dept of Math & Computer Science
3800 Hillsborough Street
Raleigh, NC 27607
(919) 829-8614

e-mail address:
lalchand@ecsvax.uncecs.edu

APPENDIX F

INTERVIEW GUIDELINES AND QUESTIONS

All faculty who are teaching introductory computer programming classes were contacted first by a letter explaining in details about the study. Then a follow-up telephone conversation and/or actual meeting was decided between the researcher and the faculty at the faculty's convenience. This was usually the same day when the students' questionnaires were delivered in-person to the faculty teaching the class(es). Total interview time was thirty minutes. In order to be rigorous about obtaining information, I made sure that all the participating faculty were answering the same set of questions. An agenda of the interview was prepared which included the set of open-ended questions listed below. After the permission of the participant, all the interviews were recorded on an audio tape.

Questions for faculty interview:

1. What are your thoughts about the introductory computer science course with regards to its content and the language used?
2. What do you think about the future of introductory computer science course?
3. What is the future of programming languages used in introductory computer science courses?

4. What are your thoughts about ACM guidelines with regards to introductory computer science (CS1) curriculum?
5. What do you think about students' input in the decision process of selecting a programming language for CS1 course?
6. What are your thoughts about the selection process of a programming language used in CS1 course at your school?
7. Do you suggest any changes in the selection process?
8. Any final comments?

APPENDIX G

SAMPLE INTERVIEW TRANSCRIPTS

Sample Transcript #1

Q - Hi, [M . .]. This is Lal Shimpi. How are you doing?

A - I am fine. You are on right time, too!

Q - Yes. So, can I have your permission to record this interview?

A - Yes, no problem.

Q - O. K. Now I am recording. Now, as I said in my letter to you, the purpose of this interview is to get some answers to some open-ended questions which deal with the choice of a first programming language. For example, the selection process within your department, the time-line for changing a new first programming language, whether students (current and/or past graduates) are involved in this decision process, ACM guidelines with regards to the first programming language, and so on. So, let me start with the first question. What are your thoughts about the introductory programming language course (CS1 course) at your school with regards to the language you use and the content of the course?

A - O. K. That course, really, in my view, we need to give basic information to students. I approach the class as we should not just be teaching a language, but we should be teaching good concepts, and a couple of concepts I work for are proper use of variables: if it is declared as an integer; it should be used as an integer, if it is declared as a character then it should be used as a character those kinds of things, which are really basic things. They need to know the difference between sequencing, decision making, looping and subroutines, procedures and functions, When I teach the class, I really stress when a function should be used, when a procedure should be used. In a function, no side effects, no reading or writing, it should calculate a value, return a value, and not do anything else. If you have to do reads or writes or whatever, you should use a procedure. All local variables, passing everything. Those are concepts that I think they should learn. Then I look at what vehicle we are going to do it with and what we can use here is Pascal. I look at Pascal not as a language that they going to use when they get out in the real world when working but a language that will force them to do the things I just said I need to them to pick up.

Q - Why Pascal at this place?

A - I think primarily because it was written as a teaching language and so it has some good features: it makes them to declare variables ahead of time which is something better than a language like FORTRAN in which you can just slide in your variables. It is a structured language and It is relatively easy to start. As you know that they can sit down and write a program after the first day. I thought a lot about should we move to C or C ++. I would not use C without an add-on. Something to force them into some better habits than C would give you. I think they could get themselves into way too much trouble; I think the instructor will go nuts trying to debug C for students because basically anything would work, you know, you can declare variables as one thing and use it in a different way. You can declare variables sequentially and use them as arrays; I mean that is really a powerful stuff for upper-level courses, but I am sure would not want to teach it in the beginning level courses. I have looked at a little bit to some add-ons using C++ or using some product that is like basically a pre-compiler to C to force them into some good habits and I am not opposed to those on the other hand I don't see why you should change just for sake of change. If using Pascal is not broken why change.

Q - Lots of schools are going for C right now. Back home in India, schools are trying to use C. I think they are going to C for the wrong reasons. The reason they want to use C is not because they want to teach the language, but because everyone is saying that 'C is good' , 'C is good'.

A - It is good and in a production environment, it is probably what a lot of our students are going to use it but I don't think that it is what we need to do in the first course. Now, I agree that I will look at something like should they use C with data structures as an example or some higher level course. But I am not really comfortable with C in the beginning. I don't think it will install good habits for the programmers. I am not that concerned after that first course they can get that many jobs using what they learned. You know, what are we doing? Are we teaching a skill how to write a program in that particular language or are we teaching good habits, are we making good programmers? And I hope what we are doing is making good programmers,

Q - So, at least we have to teach good programming tactics. Because that is the first course we are teaching them how to write a program, so there better be a good structured way of how to write a program.

A - That's right. Good foundation for whatever they are going to build on.

Q - What do think what will happen in a couple of years in that course?

A - Here, at our school or across the country?Q - Both -- at this school and throughout the country.

A - I think, people will move away from Pascal. I think they will move to something more recent and I think part of the reason they will change is because of the change in the structure of computer science. Computer science itself changes so much. The hardware changes. I think that people almost feel compelled that they need to change what they are teaching to stay up because everything else is changing. I don't know what language it will be. I have been to talks where people have suggested number of things. There are those MODULA-2 proponents out there. I was not really happy with MODULA-2 either. I don't know. There are a variety of things. You could look at perhaps to do programming through SQL or you look at some of the other products like I am going to give an off the wall example. If you look at Word Perfect for Windows, there is a macro writing facility. The macro book that fits all the macro commands is an entire programming language: it has loops, it has decision-making; you can call other macros. There is a book just as big as our Pascal text book.

Q - Like dBASE is ..

A - Exactly ! There are a variety of things that can be used. I don't know. May be people will go in a lot of different directions. Because we have different options, now.

Q - But, because of the ACM guidelines, is that possible?

A - Yeah, but ACM does not specify what we should teach. They just specify what should be taught.

Q - So, you don't see any future for Object Oriented Programming for the first course?

A - Well, you know, I see a lot of future in an object oriented programming, but in a next couple of years for the first course, I don't know. May be down the road perhaps after 6 - 8 years. But I just don't think the change will happen that fast. I will give you kind of an analogy. People have been saying COBOL has been dead for years. But we still are teaching COBOL, we still are hiring people who are teaching/using COBOL. I think it is going to take a long time before object oriented programming is going to be used in that first course. We have lots of companies that they are not using it, lots of people who are trained in it are not using it. I don't feel adequately prepared to teach a first course in object oriented programming. I need to rethink the way programming is taught if that is what I am going to do. I won't be ready to do it.

Q - Scarcity of the availability of software for object oriented programming ...

A - That's right. That and the textbooks that are out there. If you look at the textbooks, most of the text books about programming assume that you know how to program.

Q - It is like the change of BASIC to Pascal. Ten or so years ago, we used to start with BASIC in our first programming ...

A - Yeah, but you know my first course was FORTRAN.

Q - Oh, your's was FORTRAN..

A - Yeah, I did FORTRAN, Advanced FORTRAN, then COBOL, and advanced COBOL. When I took Data Structures, we had to learn, it was a four credit class, actually three credits were for data structures and one credit was for learning Pascal.

Q - So, what year was that?

A - Ah!

Q - Was it using punch cards?

A - Ha! Ha! Ha! Yes, it was using punch cards!

Q - Well, I kind of had a strange start. My first Computer Science class was with my first degree. The first computer science class I took was FORTRAN back in 1973. But I did not get a degree then. I left and went back in, I want to say, was it 1980 or 1981? and really started working part-time on a B.S. degree because at that time I had another job. At that time I took advanced FORTRAN and then COBOL. So, you can see, first time when I went to school, Pascal was not around.

Q - Oh, Pascal was not there?

A - No ! Pascal was developed in early 70's.

Q - But, if you had a choice, like right now, you have so many choices, which language you would start? Suppose, I want to learn my first programming language, what language should I learn first?

A - Ah ! It depends on your background. If you are mature enough, I think starting with an object oriented language like C++ in the first class might be reasonable, but if you are eighteen year old coming through high school as a

traditional student, I really like something more structured.

Q - Like Pascal?

A - Yeah! Pascal and I won't be opposed to do something like using the language out of dBASE IV. I mean, anything that is more structured. Ah, I just think that object-oriented is such a different concept that I don't know that I that point whether the students are mature enough. I think they need some other things under their belt. They don't have enough math-reasoning, they don't have enough problem-solving, they don't have enough abstraction skills.

Q - Of course, it is their first course. They don't have anything at all! They are learning programming. They are learning about how to write programs ...

A - That's right ! And I think there are too many other things that you really need to know to be good at object oriented programming.

Q - So, you don't see right away object-oriented programming in the first programming class?

A - Sure, I don't !!!

Q - Within five-seven years?

A- For the first course, I am sure I don't ! Of course, for the upper level courses, I think, it is very important and I would hate at this point to have students graduate without having a course in it. But that first course, no ! no! no!

Q - Now, when you select a language for the first course here, was there a selection process? How did you decide to use Pascal?

A - I inherited Pascal! You know, I taught at ... before I came here, and I also inherited it there. Since I have been here, we talked about should Pascal be the one? And every time, we are coming up with the same conclusion that it works well for us. What I would much rather see happen is more of an emphasis is put on a laboratory along with the course, so that students get lots of practice with it. I am more concerned about the amount of practice they have than should we change to something else or not. Because, I think, Pascal is working okay.

Q - So, there was no other classification except that it was a structured language...

A - Yes and there is the thing of availability of a compilers.

Q - So, the availability of hardware and software ...

A - Yes! Hardware and software availability were also the reasons to go with Pascal, and Pascal works so well on a PC and it is practically so cheap to use it. Also, it gives students capability to do it in their own room. If you are looking at something like C++, you are talking about relatively a much larger package. Obviously, it will also run on a PC and the cost difference is not that great. But you are talking about a considerable different hardware requirements!

Q - So, there was no thought about job market demands and ...

A - No, and I think part of the reason that we have not talked a lot about that is because all of our graduates get jobs, and they get good jobs and most of the time what will happen is if you look at our graduates, we don't have most of our graduates working in any one language. It's not like 90% of our people work with C because they don't and I think if that would happen we might look at something else. But our students, our graduates work in such a variety of areas that most of the time what happens is our graduates will get hired and they will go for two-three days of training in whatever they are going to work in and they make adjustment very quickly because they have a good foundation.

Q - So, main thing for the selection was hardware and software price, and because it was a good language to work in,

A - Yeah, right !

Q - Most of the schools, I think, when they select a language they never ask students but only ask faculty. Should we involve students in deciding when we change a language? Should we get the input from students?

A - Well, I think, what will be important is to get some comments from people who have gone through our program. I don't know, people coming in really have not any basis to make a decision. I always think it is funny, when we ask a student if that was a good text book for the course they have just gone through. Because they don't have anything to judge it on. They never used another text book. They never seen anything else, and I think it is the same thing if you ask a student who has never used any other language that if this is a good language. Well, they don't have anything to compare it to. I like to go back and survey our graduates ...

Q - Those who have gone through your program, like seniors ...

A - Yes, exactly right! Well, I would really like to ask people who are in the job market and to see what they think. Talk to some people who are programming in

C, we know a number of people who are writing application software using C, like we have students working with IBM writing application software using C, we have a number of COBOL programmers, and see what they would say. Now, they are in the job market and they have been through our program. They could tell us that.

Q - Last thing is about the ACM guidelines for the course. What do you think about the ACM guidelines?

A - Oh !

Q - These are the new guidelines for CS1 and CS2 also.

A - They are, let me see how do I want to put it, you have to be very energetic to get through. I think that's a lot to cover in the first course. I think there are some good ideas, But, you have to look at the difference in a program like here at our school and the program like at ... school, our program is not the same like theirs. Personally, I could not cover all that stuff they have in the guidelines. Not without making this a four-credit course and I don't know whether that's what we should do. At ... (school) when you are trying to get a liberal arts education, if you bring people into that major and do that in the first course, I don't think they will be there for the second course!

Q - That's right !

A - And you know the thing is, Lal, people are not coming to ...(school). People do not come here to major in Computer Science. People come here because they want to go to school here and then they figure out that they want to do is major in Computer Science and that's when they come into our program. When people go to some other schools, they go to ... (school) because of the Computer Science program and they know that they are in for that program and not a liberal arts education. And I am not saying that those students are weaker students. I am saying that those students are well-rounded.

Q - They don't come for your program only !

A - That's right. They come for school as a whole, and so, it would really concern me to try to cover that much material on that period of time. Specially considering that our students start generally when they are sophomores in programming. Sometimes they are juniors. It is overwhelming to look at the amount of material that they expect to be covered. I just don't see how we can do it. Now, I am not only talking about Computer Science I (CS1), but ACM curriculum guidelines as a whole. I don't know whether you can cover all of that. It is really tough. I don't know.

Again, I am not saying that all those things should not be covered. But you are back to the whole argument of do you do depth-first or breadth-first. I don't know whether one is better than the other and for some of us it is just typical. We will do one for a while and go back to the other.

Q - And, I don't know whether other colleges, who covers all the material? There is no way one can cover all the material.

A - Yeah.

Q - So, what do you think will they change the guidelines?

A - You know, I am actually surprised that it has not happened before this, where the major splits out a little more so we just are not majoring in Computer Science but, and I guess some schools do this but the program is more specialized where you do either software development or you do more hardware development or even in software development, you do more business applications or you do more scientific applications and I think you got to split some of that stuff out because when these people get jobs, they need to have breadth, granted, but they don't need to know all of that stuff because most of the times they are going to end up working on a small segment anyway. I am always amazed when I talked to folks from IBM, they might know about networking, yet they don't know how to use a PC, or they might know something about a PC but they have no idea about an IBM main-frame. So, I don't know about whether we are not doing some injustice to our students by trying to teach them such breadth.

Q - So, are they going to change sometime soon?

A - No, they will just do the guidelines. We probably look at them for couple of years and see what works and in the next round, they will do the changes again! When were the last guidelines done?

Q - 1991!

A - Before that?

Q - 1978!

A - So, twelve years before they changed them.

Q - Actually, the recent guidelines were done in 1988 but they did not come out

till 1991.

A - So, let us say about every 10 years they change. I bet, within the next 10, the guidelines will be dramatically different.

Q - Right now, they have introduced modules. You are supposed to cover module 1, module 2, module 7, and so on.. You can skip some of them.

A - Yeah! Pretty confusing!

Q - The big problem is for the person who is writing a book? He/she cannot write a book which will contain all these modules, because the book will be a huge volume. No body is going to buy the book. So, most of them are using the old guidelines.

A - I think, you are right. You know the other thing is that, I get really concerned about programs like ours that is small where there is one full-time person and couple of part-time people, how can you maintain, how can you keep up with all of that across the board? You know, I always think of the old jack of all trades and master of none, it is really difficult to teach in a program where you only have one or even two people to be able to teach all of that confidently.

Q - I am sure, students might get sick of seeing that one face all the times, because they will come from one class and see the same face another computer class..

A - Yeah ! Well, you know, we almost do a disservice to our students if we pretend to be an expert in all of those areas. Because there is no way that anybody true really is !

Q - Well, that was very good. Any final thoughts?

A - No, but I would like to see the results.

Q - Yes, I will let you know about the findings. Thanks very much.

Sample Transcript #2

Q - Hi, I am Lal Shimpi from Meredith College. I would like to start the interview once you give me permission to record it. Do I have your permission to record our conversation?

A - Yeah, I am ready at this end. Start your questions.

Q - First question is about the programming used in the first computer science classes. What language do you use for your first computer science course?

A - First computer science course you mean the first CIS course right here in our division?

Q - Yeah, the first programming language class.

A - COBOL is used as our first programming language.

Q - What should be used as a first programming language?

A - I think we should use, since this is a business division, COBOL is I think appropriate. If it is for scientific applications, I think, it is better to use either Pascal or C.

Q - So, the other departments should use either Pascal or C. Why?

A - For scientific applications, I think, a structured programming language will be appropriate for math and other scientific applications.

Q - If you had a choice between Pascal and C, which one would you choose?

A - C without any doubts !

Q - Is it a good choice for the first programming language class?

A - Yes, I think so. These days most of the universities are using C as the first programming language.

Q - Next, I would like to ask you about the content of a first programming language class. What should we cover in a first programming language course?

A - OK. We should cover basic fundamentals like how to declare data types,

variables. They should learn if-else statement, loops, up to arrays, I think, we should cover.

Q - No records ...

A - Yeah, no records, no pointers, no structures.

Q - What do you think about the ACM guidelines? Do you agree with the ACM guidelines with respect to the first programming language course?

A - Actually, it depends upon the college and quality of students. First of all, you have to see how much the students know and how much capacity they have to learn. You have to see that whether they are ready to learn or not. Then you can go up to pointers. If the students are below average or average, then I don't think we should cover up to pointers.

Q - So, we don't have to follow the ACM guidelines.

A - No, we should follow the ACM guidelines. But sometimes, you know, we cannot follow them.

Q - OK ! What do think will happen within the next five years to programming languages? What language will dominate the first programming language classes?

A - After five years? You mean in the future?

Q - Yes!

A - May be C, but C++ may take over C.

Q - Any other new language do you think will emerge?

A - Right now, I do not think so.

Q - What about object oriented programming?

A - That is also going to be very popular.

Q - Now, when you select a language, do you ask students what language should be used in the first programming language class?

A - First of all, students do not know what is a programming language.

Q - But, what about the seniors? Should you ask them about the language selection?

A - Yes, I think, we should ask them. At least we will get some feed back from seniors. Also, we should ask our past graduates who are in the work force and get their feed back about the selection of a programming language. I think it will be a good idea.

Q - How do you select a language? What is a selection process in the department? Do you decide it? Is there a committee who makes the decision?

A - OK ! Yes, I can decide myself. I will have to sit with other members of the CIS faculty and the chairperson and discuss the advantages and disadvantages of the language. Also discuss about the hardware and the software.

Q - Does it make a difference about a language used in the first programming language class and programming languages used in the other CIS classes? Do you pay attention when making a selection of a first programming language about the languages used in the follow-up CIS courses?

A - Of Course, we should pay attention. We have to pay attention when selecting a first programming language.

Q - What do think, what will happen to the first programming language class the way it has been taught in your department right now and in the future? Will it be the same?

A - No, I think, we are lagging behind in that class as compared to some other schools. The way we teach, and the number of chapters we cover, it should definitely be changed in the near future. To compete with others, we have to cover more material and change the way we teach our first programming language class.

Q - For example, the Microcomputer Applications course has changed over the time, do you think, the same thing will happen with the first programming language class? Will it change because we have to teach students coming up with different backgrounds or what?

A - Yeah, after few years the course will change definitely. Then, we won't have to teach some of the basic concepts. Right away, we can start with data structures.

Q - Any other comments?

A - I enjoyed the interview. I thought it was very informative. I will be interested in the results.

Sample Transcript #3

Q - The first question is about what language do you use in your first programming language class?

A - One year ago, we switched from Pascal to C++. Though Pascal is the best structured teaching language, but is not used widely in the industry which is widely using either C or C++. All they are (I mean the industry) looking is people who can program in C or C++. The reason we did not go with C was that when we were choosing C compiler, most of them assumed that you knew a lot about programming while the C++ compilers were a little more user-friendly for a new programmer. They give you more warnings or error messages than the C compilers.

Q - Is it a Turbo C++ or UNIX C++?

A - It is UNIX C++ with lots of work stations.

Q - Last summer, one of my students were taking a course on UNIX here in your department, but they were using C and not C++. It was taught by some adjunct faculty.

A - That's a different course! We have a course 258 which is C but it assumes that they had some data structures and that's for those people who know generally how to program or want to learn fast how to write programs. We have that course may be for ten years. But, that's a different than teaching a new programmer how to program.

Q - Are there different 'first' programming languages? For example, C++ for CS majors, and some other language for other majors.

A - Yes, there are different first programming languages. We still teach Pascal in our 110 classes once a year now because there are couple of engineering curriculums that did not want to switch to C or C++. They did not want to learn anything new, and that's why they use Pascal. The second group is 112. In 112 we use FORTRAN because those engineers think that FORTRAN is still the number cruncher, always has been and always will be and they do not want to switch! And the third group is the group on which we have a control over and for this group, we think that the best language right now is C++ for lots of reasons. There was a big article in the newspaper which says a lot better than what I can about object oriented programming. It basically talks about how you can do a program so much faster with objects because you can just take modules from

what you have done before and put them together and you are building with blocks instead of with individual lines of code, Once you are comfortable with the concept of modules, you can just put them together and that is what this whole thing is about.

Q - Pascal has been used for more than 10 years ...

A - We have been using Pascal for 13 years.

Q - What do think will happen with the next, say, 5 years from now?

A - Better object oriented programming language! I don't think we will get away from object oriented programming. Like the newspaper article said, it is just easy, we don't have to re-invent the wheel every time. If you want a particular module and you have done it before, you don't have to do it again. You just pick it out and put it in.

Q - What about the students? Are they comfortable using C++ as the first programming language?

A - No, it is difficult. But students are more sophisticated and they are ready to learn because it is more difficult. If we make it too easy for them, they may not take it more seriously. In fact, our Pascal course was the harder course. But C++ is a challenge course and they seem to like a challenge, That is really not a problem. They work hard. I know, they complain, they always complain, but finally they are gaining more knowledge.

Q - So, will C++ be used for few years?

A - I am sure it will stay for few years. For example, we had Pascal for 13 years. Before Pascal, we used PL/1 only for five or six years and the reason was that it was so big that you can never hope to learn the whole thing! It was not feasible for your own PC.

Q - It is the same thing with ADA. It was too big to be used on a PC. Otherwise, everyone was thinking that ADA will be the language of 90s! But that never happened.

A - Yes ! That's true! Before PL/1, we used to use FORTRAN. When I started here, We were teaching FORTRAN. Over twenty-five years of our history, we have switched four times.

Q - So, Pascal stayed there the longest!

A - Yes, Pascal was there the longest. Because it was really the best! That's why it was designed for.

Q - Now, do you have any say over other courses in other departments in the use and/or selection of languages, first programming languages?

A - Yes! I have just received a message from Mathematics Education department saying that those students who want to minor in Computer Science, what should they take first? The student has to take C++. Because if they don't take C++, they cannot go further! All our other courses in the department have a prerequisite of C++. So, if other people are using our courses and they want to go on then they have to take C++. We have a minor in Computer Science and they have to do C++. If you need only one course in our department then you can take either one of them but if you want to go on then you must take C++. So, we do have a little bit of control.

Q - Are these students, at least some of them, coming with Pascal background? Do they take a Pascal course first and then take C++?

A - No, if they have Pascal it is transferred in. Most of the high schools do Pascal. Most of the AG tests use Pascal, but they are going to change it to C++. I don't know if they are going keep both.

Q - What do think about the teachers in high schools? They got to know C++ or C before they can teach these kids.

A - We get a lot of them here taking that C course in the evening. And, if you are a programmer, you can take that one course and pretty much switch the language. So, I don't see any problem. Any thing you need to learn, you just have to learn it. If you are a programmer, you can pretty much switch a language.

Q - So, what do you see in the next five years, if the high school kids are going to come out with C++ background then what we have to do at the college/university level?

A - Well, when they come out of high school, see, our first course does not assume that they know how to program and I did a survey of one year of one class about how many people had x amount of experience with Pascal and there was no correlation between the people who knew how to program when they came here and the grades they made on these courses. Because what happens is that the half of the grade in the first course is internal documentation and if they are self taught, they would not have done any of that. So, they lost 50% of the grade right there. Basically they say that "my program runs, I get the

right answers, i don't have to document !" They loose a lots of points. They know too much. They know how to write programs, they know it will work without the documentation but they do not have these good habits. So, it is almost like you have to wipe the slate clean if they know how to program because they have picked up a lots of bad habits and teach them right habits.

Q - That is one of the biggest problem, I think. People who come with a background like that. Particularly, that is what used to happen with people who came with a BASIC background. They were so much used to using GOTOs, so when they were taking Pascal, they would use the same logic in their mind to write a code.

A - You know, a lot of times they will be sitting in that class and thinking that "Oh, I know this stuff ..." and they don't really pay attention and one day they wake up and they are lost because they have not paid attention. All that time they were just translating from what they know from Pascal and it is too late.

Q - How do you select a language for the first course? Do you have a group or any one person? Who decides it?

A - Oh, that was a terrible, terrible, long and agonizing process. I have got probably a fourth of file cabinet full of documentation, electronic mail back and forth from all the faculty in the Computer Science department: ones that wanted and ones that didn't. Lots of different newspaper articles, lots of e-mail messages! We had e-mail war for almost an year and half on this subject and we have been agonizing over it for, may be, five years now before we decide to switch.

Q - So, was there a split between the group?

A - The main split was that people did not want to have to learn anything new.

Q - You mean the faculty?

A - Yes, the faculty! They are too busy with their agenda! You know, every one has their own agenda! That means they have to spend their energy. All the faculty do not know C++. There are 3 or 4 people who know it and the rest of them don't really want to learn it so they teach upper-level classes. Most of them say that if it not broken why fix it. It is a lot of attitudes, don't bother me type.

Q - But, we have keep up with the changing technology, so they will have to change. One thing you did not mention are the students. Do you take students' views into account when you select a language? They may not current students but who have graduated and are in the work force.

A - We do have some surveys that did include students. And students, when they start are not capable of giving some significant input. One of our nationally well-known faculty, he wasn't concerned because he was mostly doing only graduate-level courses until his son came here as a freshman. When his was trying to get a job during the summer, everybody wanted him to know C or C++. Then, this faculty member came to me and asked: "what is this? why are we teaching Pascal? Why are not we teaching C or C++?" and he had never been on the conversation before. Now, certainly he is one of the main pushers for C++ because it was very important for him.

Q - Second thing is that do you pay very close attention to what language do you use in the next courses as compared to the first language you use?

A - Oh, yes! Definitely! See, we don't switch languages like the technical institutes do. They will have COBOL for couple of quarters, then they switch to PL/1, then to Pascal, APL, and so on. We do not do that. Our primary teaching language for 3 sequential courses is C++ and we really do not want any other language except a course like assembly language where we teach assembly language and an operating systems course...

Q - In an operating systems course, they could use C++...

A - Yes, they could. But they have to have assembly. Then, we have a numerical methods course, where they have to do some number crunching and people who are teaching the course think that FORTRAN is the number cruncher, so they have to use FORTRAN where the programs are not that big, cumbersome. After they had a couple of semesters of courses like Pascal to switch to FORTRAN to do some number crunching is not that difficult. Though they have never seen it and it is really not taught in the class, they usually pick it up easily.

Q - There are lots of books on numerical analysis and I am sure they will either suggest use of Pascal or FORTRAN.

A - Always FORTRAN! Not Pascal ! None of the instructors who have taught our numerical analysis course have used Pascal. Always FORTRAN! It is really up to the instructor, but they have always used FORTRAN.

Q - Or they should give the students a choice of the language. They can use whatever language they know to program. You mentioned C++ courses. Is there a C++ 1 and C++ 2 courses?

A - Yes, plus data structures!

Q - What is the dividing line? How far do they go in the first course?

A - I will give you a copy of our course syllabi that will show you the material covered in each of those classes.

Q - It is like Pascal – Pascal 1 and Pascal 2 !

A - Exactly!

Q - So, are you following the ACM guidelines?

A - ACM guidelines, yes! yes! It is exactly the same syllabus we have for Pascal 1 and Pascal 2. Actually, you cannot tell the difference.

Q - Do you agree 100% with the ACM guidelines? Do you do exactly what ever they say?

A - Yes, we do whatever they say. We are ACM accredited, we may not agree with the guidelines but we do whatever they say.

Q - As a personal opinion, what do you think about the ACM guidelines?

A - I have always been on the curriculum committee so I always have to deal with the ACM guidelines and every time we go through our curriculum revision, we go through the ACM guidelines. My personal opinion is that it is better than having no guidelines at all!

Q - OK! The new 1991 guidelines, they do not really tell you exactly how much material to cover. Lot of people have lot of problems with these guidelines. With the books too, since the guidelines only tell you that these are the fifteen modules and you pick and choose. You might choose to teach modules 1 through 9 and then modules 14 while others might cover modules 1 through 7 and the module 15. Depending upon the concentration, everyone can choose their own modules and still claim to follow the ACM guidelines and teach the same course. The people who are writing books for these classes using these new guidelines face a problem of which modules to be included in the book. Of Course, one solution might be to include all the modules except in this case the book will be a hugh volume! What do think will happen to the ACM guidelines with respect to this first programming course in the future?

A- Oh, I don't know. I really don't know much to comment on that. The ACM guidelines, I don't think that they can get any more stricter but they can get looser. If they get too stricter they might loose lots of their supporters.

They have to get looser. Like the requirements for accreditation, they are so strict now that they are restricting education. They need to back up a little! I think they will move towards more freedom: "do what you want".

Q - They got to keep everybody happy too! You know, there are always these two groups: one that is pure hard core computer science and the others. The first programming language course is an important course because depending upon how do they do in the course and how/what we teach in that course will make them continue taking more classes in the major or they will change their major. I thought, we have to be very careful in how we teach the first course and also what we teach in that first course!

A - I can give you a detailed syllabus so that you can see. Also, two of our faculty members are writing their own C++ book. We have been using it for two years. We taught a couple of pilots using the same book. We have testing out every chapter they right. There are no textbooks covering C++. There are couple in the writing process. But none are available today!

Q - Well, thanks very much for your time. See you, Bye, Bye!

A - Bye!

BIBLIOGRAPHY

1. Abbott, R.J., "An Informal Survey of Computer Science Courses", SIGCSE Bulletin (ACM), Vol. 7, No. 2, June 1975.
2. ACM Curriculum Committee on Computer Science. "Curriculum Recommendations For The Undergraduate Program in Computer Science. A working report of the ACM Committee on Curriculum in Computer Sciences," SIGCSC Bulletin (ACM), Vol 1, No. 1, p. 1-16.
3. ACM Curricula Recommendations for Computer Science Volume I, p. 1-78.
4. ACM Curriculum Committee, "Input from ACM Curriculum Committee on Computer Science", SIGCSE Bulletin (ACM) Vol. 1, No. 2, p.30-34.
5. Ahlgren, D., A. Sapega and H. Warner. "A Sequence of Computing Courses for Liberal Arts Colleges," SIGCSE Bulletin (ACM), Vol 10, No. 1, (Feb. 1978) p. 180-182.
6. Alspaugh, C.A. "Identification of Some Components of Computer Programming Aptitude," Journal for Research in Mathematics Education, Vol. 3, (1972), p. 89-98.
7. Arblaster, Andrew. "Human Factors in the Design and Use of Computing Languages," International Journal of Man-Machine Studies, Vol. 17, (1982), p. 211 - 224.
8. Atchison, W. F. "Computer Education Past, Present, and Future," SIGCSE Bulletin (ACM), Vol. 13, No. 4, (December 1981), p. 2-6.
9. Augenstein, Moshe., Tanenbaum, Aaron., and Weiss, Gerald. "Selecting a Primary Programming Language for a Computer Science Curriculum: PL/I, Pascal and Ada," SIGCSE Bulletin(ACM), Vol. 15, No. 1, (Feb. 1983), p. 148-153.
10. Austing, R.H., Barnes, B.H., and Engel, G.L. "A Survey of the literature in Computer Science Education since Curriculum '68", Communications of the ACM, Vol 20, No. 1, (January, 1977), p. 13-21.
11. Austing, Richard H. and Engel, Gerald L., "A Computer Science Course Program for Small Colleges", Communications of the ACM, Vol. 16, No. 3, 1973.

12. Baron, Naomi S. "The Future of Computer Languages: Implications for Education," SIGCSE Bulletin (ACM), Vol. 34, No. 3, (1986), p. 44-49.
13. Baron, Naomi S. "Computer Languages: An Explorer's Guide," New York: Doubleday (1986).
14. Bauer, M. A. "Experiences with Pascal in an Introductory Course," SIGCSE Bulletin (ACM), Vol. 11, No. 1, (1979), p. 158-161.
15. Bauer, Roger, Mehrens, William A., and Vinsonhaler, John F. "Predicting Performance in a Computer Programming Course," Educational and Psychological Measurement, Vol. 28 (1968), p. 1159-64.
16. Beidler, John, Austing, Richard H. and Cassel, Lillian N. "Computing Programs in Small Colleges," Communications of the ACM, Vol 28, No. 6, (Jun 1985), p. 605-616.
17. Bell, Doug., and Scott, Peter. "A First Course in Programming," SIGCSE Bulletin (ACM), Vol. 19, No. 2, (June 1987), p. 48-50 and 57.
18. Blaisdell, J.H., and Burroughs, A. "How To Tell if a Programming Language is OK: What's wrong with BASIC for Teaching Business Students How to Program?," SIGCSE Bulletin (ACM), Vol. 17, No. 3, (1985), p. 5-8.
19. Blank, George. "A Tourist's Guide to the Cybernetic Tower of Babel," Creative Computing, Vol. 7, No. 11, (Nov. 1981), p. 94, 96, 98, 100,102-103.
20. Boom, H.J., and Jong, E.D., "A Critical Comparison of Several Programming Language Implementations," Software-Practice and Experience, 10, (1980), p. 435-473.
21. Brookshear, J. Glenn. "The University Computer Science Curriculum: Education Versus Training," SIGCSE Bulletin (ACM) Vol. 17, No. 1, (March, 1985), p. 23-30.
22. Butcher, D.F., and Muth, W.A. "Predicting Performance in an Introductory Computer Science Course," Communications of the ACM, Vol. 28, No. 3, (1985), p. 263-268.
23. Cameron, J. S., and Karian, Z. A. "Computer Sciences Curricula for Small Colleges," SIGCSE Bulletin (ACM), Vol 5, No. 2, p. 215-219.

24. Cheatham, T. E. Jr. "The Recent Evolution of Programming Languages, "Information Processing 71: Proceedings of the IFIP Congress 1971 (Edited by Friedman, C. V.), Vol. 1, p. 298-313, New York: North-Holland; 1972.
25. Cherniak, R. "Introductory Programming Reconsidered – A User Oriented Approach," SIGCSE Bulletin (ACM), Vol. 8, No. 1, (Feb. 1976), p. 65-68.
26. Cohen, Peter. "Computing in College Courses: The Dartmouth Experience," ERIC, AN: ED217877, CHN: IR010263, (1982).
27. Cole, Bernard., "Choosing What's Best For Your Purposes: The Family Tree Of Computer Languages," Popular Computing , Vol. 2, No. 11, (Sept. 1983) p. 82-86.
28. Coombs, M.J., Gibson, R., and Aity, J.L. "Learning a First Computer Language: Strategies for Making Sense," International Journal of Man-Machine Studies, (1982), Vol. 16, p. 449-486.
29. Cugini, John V. "Selection and use of General-Purpose Programming Languages– Overview," Vol. 1, National Bureau of Standards, Washington, D.C. Inst. for Computer Sciences and Technology, (1984).
30. Cunningham R. S. "Computing And Computer Science in The Liberal Arts College," SIGCSE Bulletin (ACM), Vol 8, No. 3, p. 24-25.
31. Czedik-Eysenberg, G. "Programming Languages for Teaching Computer Science: A Comparative Analysis of Pascal, Ada and Modula-2" Dissertation (1988).
32. Doyle, J.R., Stretch, D.D., "The Classification of Programming Languages by Usage," International Journal of Man-Machine Studies, Vol. 26, No. 3, (March 1987), p. 343-360.
33. Dupras, M., Lemay, F., and Mili, A. "Some Thoughts on Teaching First Year Programming," SIGCSE Bulletin (ACM) Vol. 16, No. 1 (1984), p. 148-153.
34. Eastman, Caroline M., and Blumsack, Steven L. "Repackaging The Introductory Course To Separate The Control Language," SIGCSE Bulletin (ACM), Vol. 24, No.4, p. 155-157.
35. Ellison, Robert J. "A Programming Sequence for the Liberal Arts College," SIGCSE Bulletin (ACM), Vol. 18, No. 2, p. 161-164.

36. Elson, Mark. Concepts of Programming Languages, Chicago, IL: Science Research Associates, Inc., 1973.
37. Evans, A., A Comparison of Programming Languages: Ada, Pascal, C Prentice-Hall, 1984, p. 66-94.
38. Ever, Jacob. "BASIC beats PASCAL," Creative Computing, Vol. 7, No. 11 (Nov. 1981), p. 166, 168, 170, 172.
39. Fasana, Paul J. and Shank, Russell., "Tutorial on Generalized Programming Languages and Systems," American Society for Information Science, Annual Convention (1967), New York.
40. Ferchichi, Ahmed, and Jaoua, Ali. "Teaching First Year Programming: A Proposal," SIGCSE Bulletin (ACM), Vol. 19, No. 3 (Sept. 1987), p. 48-50.
41. Fosberg, Mary Lee Harris. "Adapting Curriculum 78 To a Small University Environment", SIGCSE Bulletin (ACM), Vol. 34, No. 4, p. 179-183.
42. Frank, Thomas S., Smith, James F. "Ada as a CS1-CS2 Language," SIGCSE Bulletin (ACM), Vol. 22, No. 2 (June 1990), p. 47-51.
43. Friedman, Linda Weiser. "From Babbage To Babel and Beyond: A Brief History of Programming Languages," Computer Languages, Vol. 17, No. 1 (January 1990), p. 1-17.
44. Friedstein, Harriet G. "What Computer Courses Do student Really Need?", College Teaching, Vol.34.
45. Furugori, T., Jalics, P., "The First Course In Computer Science -- A Small Survey," SIGCSE Bulletin (ACM), Vol. 9, No. 1 (Feb. 1977).
46. Gabrini, Philippe, J., Adams, J. Mack., and Kurtz, Barry L. "Converting from Pascal to Modula-2 in the Undergraduate Curriculum," SIGCSE Bulletin (ACM), Vol. 15, No. 3 (1986) p. 50-52.
47. Gibbs, Norman E. and Tucker, Allen B. "A Model Curriculum for a Liberal Arts Degree in Computer Science," Communications of the ACM, Vol 29, No. 3, (Mar, 1986), p. 202- 210.
48. Goulet, Daniel V., Morris, Robert P., and Staal, Bruce. "Applying Model Curricula to a Particular Environment," Interface, Spring 1982.

49. Gries, D. "What Should we Teach in our Introductory Programming Course?" SIGCSE Bulletin (ACM), Vol. 6, No. 1 (Feb. 1974), p. 81-89.
50. Guinan, Tricia and Stephens, Larry. "Factors Affecting the Achievement of High School Students in Beginning Computer Science Courses," Journal of Computers in Mathematics and Science Teaching (Fall 1988), p. 61-63.
51. Hanson, A., and Maly, K. "A First Course in Computer Science," SIGCSE Bulletin (ACM), Vol. 7, No. 1 (Feb. 1975), p. 95-101.
52. Higman, B. A Comparative Study of Programming Languages, American Elsevier, New York, (1967).
53. Hill, David B. "Programming Languages for Service Courses and Courses for C. S. Majors," SIGCSE Bulletin (ACM), Vol. 18, No. 4, p. 43-45.
54. Horowitz, Ellis. Programming Languages: A Grand Tour, Computer Science Press, Inc. (1983).
55. Hostetler, Terry R., "Predicting Student Success in an Introductory Programming Course," SIGCSE Bulletin (ACM), Vol. 23, No. 2, p. 40-43, and 49.
56. Iverson, K.E. A Programming Language, John Wiley and Sons (1962).
57. Knuth, Donald, The Art of Computer Programming, Vols. 1, 2, 3, Addison-Wesley, Boston (1983).
58. Koffman, E. B. "The Case for Modula-2 in CS1 and CS2," SIGCSE Bulletin (ACM), Vol. 20, No. 1 (1988), p. 49-53.
59. Koffman, E.B., Miller, P.L., and Wardle, C.E., "Recommended Curriculum for CS1," Communications of the ACM, Vol. 27, No. 10 (Oct. 1984), p. 998-1001.
60. Krus, David, J., Lu, Mei Yen. "Bias in Computer Languages Comparisons: A FORTRAN Phobic Cabal?," Educational and Psychological Measurement, Vol. 47, No. 3 (Fall 1987), p.643-47.
61. LaLonde, W., Pugh, J. "Smalltalk as the First Programming Language: The Carleton Experience," Journal of Object-Oriented Programming, Vol. 3, No. 4 (1990).

62. Landin, P.J. "The next 700 Programming Languages," Communications of the ACM, 9, (1966) p. 157-166.
63. Lawrence, Dr. J., and Roth, R. Waldo. "Computer Science for Liberal Arts Colleges," SIGCSE Bulletin (ACM), Vol. 5, No. 1 (Feb. 1973), p. 70-73.
64. Lee, M.C. "A Course in Programming Languages for Computer Science Majors," SIGCSE Bulletin (ACM), Vol. 18, No. 3 (Sept. 1986), p. 17-18, 25.
65. Lee, Martin P., Jeffreys, Sue, and Peacock, Derek. "dBASE as a First Programming Language," Collegiate Microcomputer, Vol. 7, No. 2 (May 1989), p. 111-115.
66. Leeper, R.R., and Silver, J.L. "Predicting Success in a First Programming Course," SIGCSE Bulletin (ACM), Vol. 22, No. 3, p. 147-148.
67. Lemos, Ronald S. "Teaching Programming Languages: A Survey of Approaches," SIGCSE Bulletin (ACM), Vol. 30, No. 3, p. 174-181.
68. Little, Joyce Currie, Austing, Richard H., Seeds, Harice, Maniotes, John, and Engel, Gerald L. "Curriculum Recommendations and Guidelines for the Community and Junior College Career Program in Computer Programming," SIGCSE Bulletin (ACM), Vol 5, No. 3, p. 17-36.
69. Lopez, A. A., R. Ramyon and R. Tardiff. "A Survey of Computer Science Offerings in Small Liberal Arts Colleges," Communications of the ACM, Vol 20, No. 12, (Dec.1977), p. 902-906.
70. Lopez, Antonio M. "An Implementation of ACM Curriculum 77," SIGCSE Bulletin (ACM), Vol 10, No. 2, (June 1978), p. 47-52.
71. Lowther, J.L. and Motteler, Z.C. "Teaching Good Programming Techniques," SIGCSE Bulletin (ACM), Vol. 17, No. 4, p. 10-11.
72. Luker, Paul A. "Never Mind the Language, What About the Paradigm?," SIGCSE Bulletin (ACM), Vol. 28, No. 1, p. 252-256.
73. Maddux, Cleborne D. and Cummings Rhoda E., "BASIC, Logo, and Pilot: A Comparison of Three Computer Languages," LOGO in the Schools, Hayworth Press (1984), Vol. 2, No. 2-3, p. 139-63.
74. Mahoney, Michael K. "Some Thoughts on Revising a Computer Science Program," SIGCSE Bulletin (ACM), Vol. 19, No. 3, p. 20-24.

75. Marcel, Dupras, LeMay Fernand, and Mili, Ali. "Some Thoughts on Teaching First Year Programming," SIGCSE Bulletin (ACM), Vol. 31, No. 3, p.148-153.
76. Mavaddat, F. "An Experiment in Teaching Programming Languages," SIGCSE Bulletin (ACM), Vol. 25, No. 2, p. 45-55.
77. Mazlack, L.J. "Identifying Potential to acquire Programming Skill," Communications of the ACM, (January 1980), Vol. 23, No. 1, p. 14-17.
78. McGee, Linda, Polychronopoulos, Gerasimoula, and Wilson, Carol. "The Influence of BASIC on Performance in Introductory Computer Science Courses using Pascal," SIGCSE Bulletin (ACM), Vol. 19, No. 3 (Sept. 1987), p. 29-34.
79. McGettrick, Andrew D. The Definition of Programming Languages, Cambridge University Press (1980).
80. McIntyre, Pat. "Teaching Structured Programming in the Secondary Schools," Krieger Publishing Company, Melbourne, FL 32902 (1991).
81. Meinke, J.G. and Beidler, J.A. "Alternatives to the Traditional First Course in Computing," Proceedings Twelfth SIGCSE Technical Symposium, p. 57-60 (1981).
82. Merritt, Susan M. "On the Importance of Teaching PASCAL in the IS Curriculum," SIGCSE Bulletin (ACM), Vol. 15, No. 3 (1980), p. 88-91.
83. Mir, Carol Loeb. "A Comparison of String Handling in Four Programming Languages," Technical Progress Report, Masters Theses, University Of North Carolina, Chapel Hill (1972), p. 108.
84. Morris, A. H. "Can Ada Replace FORTRAN for Numerical Computation?," SIGPLAN Notices, Vol. 16, No. 12.
85. Motil, John. "Begin-BIG: An Approach to the Introductory Computing Course," SIGCSE Bulletin (ACM), Vol. 23, No. 1 (March 1991), p. 226-230.
86. Mundie, David A. "Pascal vs Basic: A Polemical Comparison of the two as General-Purpose Microprocessor Languages," People's Computers, Vol. 6, No. 4 (Jan/Feb 1978), p. 41-47.

87. Nartker, T. A., "An Undergraduate Computer Science Curriculum", SIGCSE Bulletin (ACM), Vol. 12, No. 4, p. 41-55.
88. Naugler, David, R. "Computer Science 0", Technical Committee on Computer Education Newsletter, Vol. 6, No. 1 (Summer 1989), p. 2-3.
89. Newton, Glen E. and Starkey, J. Denbigh. "Teaching Both PL/I and Fortran To Beginners," SIGCSE Bulletin (ACM), Vol. 8, No. 2, p. 106-107.
90. Nutt, G. "A Comparison of PASCAL and FORTRAN as Introductory Programming Languages," SiGPLAN Notices, Vol. 13, No. 2 (1978).
91. Oman, P.W. "Identifying Student Characteristics Influencing Success In Introductory Computer Science Courses," AEDS Journal (Winter/Spring 1986), Vol. 19, p. 226-233.
92. Organick, E.I., Forsythe, A.I., and Plummer, R.P. Programming Language Structures, Academic Press, Inc. (1978).
93. Peck, J.E.L. "Comparison of Languages," in Programming Teaching Techniques, (W.M.Turski, ed.) North Holland Publishing Company (1973).
94. Peterson, W. Introduction to Programming Languages, Prentice-Hall, Inc. (1984).
95. Pintrich, P.R., Berger, C.F., and Stemmer, P.M. "Students' Programming Behavior in a Pascal Course," Journal of Research in Science Teaching (Oct. 1987), Vol. 24, No. 5, p. 451-466.
96. Powell, James D. "Use of Model Curricula in Reviewing Established Programs," SIGCSE Bulletin (ACM), Vol 38, No. 3, (Feb. 1978), p. 12-15.
97. Prather, Ronald E. and Schlesinger, Judith D. "A Lecture/Laboratory Approach to the First Course in Programming," SIGCSE Bulletin (ACM), Vol. 15, No. 3, p. 20-25.
98. Pratt, Terrence W. Programming Languages, Prentice-Hall, Inc. (1984).
99. Ralston, A. "FORTRAN and the First Course in Computer Science," SIGCSE Bulletin (ACM), Vol. 3, No. 4 (Dec. 1974), p. 24-29.
100. Ralston, A. "The Future of Higher-Level Languages (in teaching)," International Computer Symposium (1973), A. Gunther, B. Levrat, and H. Lipps (Eds.), American Elsevier, New York (1974), p. 1-10.

101. Reisman, Sorel. "A Survey Of Pedagogical Programming Languages," SIGCSE Bulletin (ACM), Vol. 27, No. 2, p. 13-21.
102. Ruby, Douglas A. "A Survey on Computer Science Curricula," SIGCSE Bulletin (ACM), Vol 8, No. 1, (Feb. 1976), p. 313-323.
103. Sammet, J.E. Programming Languages: History and Fundamentals, Englewood Cliffs, N.J. Prentice-Hall (1970).
104. Sammet, J. E. "Programming Languages: History and Future," Communications of the ACM, Vol. 15 (1972), p. 601-610.
105. Schneider, G. Michael. "The Introductory Programming Course in Computer Science – Ten Principles," SIGCSE Bulletin (ACM), Vol. 21, No. 1(1989), p. 22 - 35.
106. Shaw, A., Almes, G.T., Newcomer, J.M., Reid, B.K., and Wulf, W. "A Comparison of Programming Languages for Software Engineering," Software-Practice and Experience, Vol. 11 (1981), p. 1-52.
107. Shirkhande, Neelima and Singh, L.P. "The War Of Languages," SIGCSE Bulletin (ACM), Vol. 18, No. 2 (June, 1986), p. 63 and 81.
108. Skublics, Suzzane, White, Paul. "Teaching Smalltalk as a First Programming Language," SIGCSE Bulletin (ACM), Vol. 23, No. 1 (March 1991), p. 231-234.
109. Smith, Carol and Rickman, Jon. "Selecting Languages for Pedagogical Tools in the Computer Science Curriculum", SIGCSE Bulletin (ACM), Vol. 8, No. 3 (Sept. 1976), p. 39-47.
110. Sointseff, N. "Programming Languages For Introductory Computing Courses – A Position Paper," SIGCSE Bulletin (ACM), Vol. 31, No. 4, p. 108-140.
111. Sointseff, N., and Yezerski, A. "A Survey Of Extensible Languages," Annual Review in Automatic Programming, Pergamon Press, London, 7, Part 5 (1974), p. 267-307.
112. Sparks, Leslie E. "Comments on Programming Languages," ACCESS, Vol. 7, No. 1, Issue 37 (Jan/Feb 1988), p. 39-41.
113. STEELMAN Requirements for DoD High Order Computer Programming Languages, Department of Defense, Washington, D.C., (1978).

114. Stoob, John C. "Thoughts on University Computer Curricula", SIGCSE Bulletin (ACM), Vol. 16, No. 3, (September, 1984), p. 13-16.
115. Swaine, Michael. "Programming Paradigms," Dr. Dobbs Journal, Vol. 15, Issue 1 (Jan 1990), p. 129-131.
116. Tatar, Gerald James. "A Comparison of Instructional Interactive Programming Techniques between the C and BASIC Programming Languages," Ph.D. Dissertation, (1986), p. 4365, Vol. 47/12-A, Dissertation Abstracts International.
117. Taylor, D. "Languages: Past, Present, and Future," Computer Languages, Vol. 4 (1987), p. 57-62.
118. Tennent, R.D., Principles of Programming Languages, Prentice-Hall International, Inc. (1981).
119. Tharp, Alan L. "Selecting the "Right" Programming Language," SIGCSE Bulletin (ACM), Vol. 14, No. 1 (Feb. 1982), p. 151-155.
120. Tharp, A.L., "A Comparison of COBOL, FORTRAN, PL-1 and SPITBOL," Computer Languages, 2, p. 171-178 (1977).
121. Tucker, Allen B., Programming Languages, McGraw-Hill Publishing Company., 1986.
122. Wardle, Caroline "A Computer Science Program At a College with Limited Resources," SIGCSE Bulletin (ACM), Vol 32, No. 3, p. 68-70.
123. Wegner, P., "Programming Languages - The First 25 Years," IEEE-TC, (C-25,12) (Dec. 1976), p: 1207-1225.
124. Wexelblat, R.L., "The Consequences of One's First-Programming Language," Software-Practice and Experience, Vol. 11, p. 733-740 (1981).
125. Wileman, S.A., Konvalina, J., and Stephens L.J. "Factors Influencing Success in Beginning Computer Science Courses," Journal of Educational Research (March/April 1981), 74(4), p. 223-26.
126. Willis, Neil, "Computing Science Courses - Training or Education", SIGCSE Bulletin (ACM), Vol. 24, No. 5, p. 298-303.

127. Winslow, Leon E., Lang, Joseph E. "Ada in CS1," SIGCSE Bulletin (ACM), Vol. 9, No. 2 (1989) p. 209-212.
128. Woodhouse, D. "Introductory Courses in Computing: Aims and Languages," Computer Education, Vol. 7, No. 2 (1983), p. 79-89.
129. Worland, Peter B. "Using the ACM Computer Science Curriculum Recommendations in a Liberal Arts College," SIGCSE Bulletin (ACM), Vol. 10, No. 4, (December, 1978), p. 16-18.
130. Zinn, K.L. "Comparative Study of Languages for Programming Interactive Use of Computers in Instruction," Boston: Office of Naval Research (1969). RM-1469.
131. "Special Issue on the Challenges of Teaching Computer Programming," Communications of the ACM, Vol. 29, No. 9 (Sept. 1986).