The University of Southern Mississippi

GENDER UNDERREPRESENTATION IN BEGINNING COMPUTER PROGRAMMING COURSES

by

Margaret Lynne Perkins West

Abstract of a Dissertation
Submitted to the Graduate School
of The University of Southern Mississippi
in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy

December 2001

UMI Number: 3038700

ProQuest Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346

The University of Southern Mississippi

GENDER UNDERREPRESENTATION IN BEGINNING COMPUTER PROGRAMMING COURSES

by

Margaret Lynne Perkins West

A Dissertation
Submitted to the Graduate School
of The University of Southern Mississippi
in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy

Approved:

Director

Dean of the Graduate School

December 2001

ABSTRACT

GENDER UNDERREPRESENTATION IN BEGINNING

COMPUTER PROGRAMMING COURSES

by

Margaret Lynne Perkins West

December 2001

This qualitative study examines the underrepresentation of females in beginning computer science courses at the community college level. The question of females majoring in computer science has resurfaced as the percentage of women earning degrees in computer science is still decreasing. It is believed that this trend will continue, and the Association for Computing Machinery (ACM) has strongly urged direct action to attract and retain more women to computing and computer science.

The main problem seems to be retention. The vast number of females who drop out of computer science have been attributed to many causes, but since 1990, research has focused on gender bias and stereotyping, societal factors, attitudes toward computing, gender grouping, computer experience, "hard" computing versus "soft" computing, learning styles, and motivation. The emphasis in research has been on external reasons that females drop out, and very little has surfaced concerning learning theory. Both mathematics and computer science are sequential in nature, in that new ideas and concepts are built on preceding knowledge. Since foundational knowledge is of paramount importance, the first courses in computer science are strategic in building that

foundation. This research study investigated the characteristics of female computer science majors, their perceptions of how they have constructed their deep internal knowledge schemas, and their learning styles.

This study proposes a working model to increase the retention of females in computer science. In this study lack of prior computing experience did not influence success. The subjects worked mostly in isolation, interacting with the instructors rather than with classmates. Gender bias is still an issue, affecting learning and self-esteem and creating an exclusionary environment for female students. All subjects reported at least one affirming relationship to help support and motivate them to continue in the program. Computer programming concepts were related not to mathematical concepts but to every day experiences. Difficulties in understanding a concept were directly related to the inability to connect that concept with a personal experience. There was no difference in learning styles between female majors, female non-majors, and male majors.

ACKNOWLEDGMENTS

This writer would like to thank her committee chair, Dr. Susan Ross, and the other committee members, Dr. Joseph Thrash, Dr. Hugh Garraway, Dr. Donald Cotten, and Dr. Jose Contreras, for their advice and support throughout the duration of this project. I would especially like to thank Dr. Susan Ross for her encouragement and guidance and Dr. J.T. Johnson for his investment of time and resources in mentoring me in educational research and statistics.

Thanks is also due to Dr. Gary E. Price of Price Systems, Inc., for permission to reproduce the reliability table for the instrument used in this study.

Great appreciation must be expressed to Ms. Linda Lightsey and Mrs. Eula West for the gracious donation of their time in editing this manuscript.

Special thanks go to my husband Bill for his tireless encouragement and constant support, to his mother, Mrs. Eula West for her unceasing prayers on my behalf, and to my parents, Mr. and Mrs. George T. Perkins, for their support and encouragement. Special thanks are also due to my prayer partners, Mary Beth Lane and Pattie Barber, for their constant encouragement and priceless friendship, and for the many other family and friends who prayed me through this graduate degree to completion.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

Figure

LIST OF TABLES

Table

CHAPTER I

INTRODUCTION

Statement of the Problem

In the United States, as in other western countries, the proportion of computer science (CS) graduates who are women has been declining in the last fifteen years (Dryburgh, 2000). At the same time the demand for computer programmers and systems analysts has been increasing. In the United States the demand has exceeded the supply in recent years, resulting in a shortage of more than 190,000 programmers and systems analysts (Dryburgh, 2000). The use of computers and computer technology is extensive in both industry and the service sector; both require computer programmers and systems analysts. Government projections indicate that future job prospects for those with CS degrees are good. So why are women not taking advantage of this situation?

While the gender gap is closing between men and women who surf the Internet, that is not the case for women entering the technology field. Even though women account for 49.9 per cent of those who use the Internet and are expected to surpass men in 2001, the rate of computer degrees earned by women does not follow that trend (Miller, 2000). Women in Technology International reported that fewer than twenty per cent of the bachelor's degrees awarded in computer science and computer engineering are earned by women, despite the fact that half of the students attending college today are female (Miller, 2000).

Since the number of women earning bachelor's degrees affects the number of women earning higher degrees and ultimately impacts the job market, these facts are of great concern. There is a critical labor shortage in the computer science field, and although women constitute more than half the population, they are a significantly under-represented percentage of the population earning computer science degrees (Camp, 1997). The rapid growth of the computing industry has created a shortage of computer scientists, and the shortage of women entering the field has exacerbated the problem (Camp, Miller, & Davies, in press). If women were attracted to the computing disciplines at an equal pace as men, then even the tremendous shortages recently reported could be filled (Camp et al., in press).

Background of the Study

The percentage of women receiving bachelor's degrees in computer science nearly doubled from 19% in 1975 to 35% in 1982 and remained steady at about that figure up to the late 1980s (Bunderson & Christensen, 1995). However, universities are reporting dramatic decreases in the percentage of women currently majoring in computer science without a corresponding decrease in other mathematics/science majors (Bunderson & Christensen, 1995). From 1983 to 1996, the percentage of BS/BA degrees across the United States awarded to women in computer science went from a high of 37.1% to a low of 27.5%, a 25.9% decrease (Camp et al., in press).

Unfortunately, the 2000 report of the Association for Computing Machinery (ACM) predicts this trend will continue. This implies that the computing community cannot sit back and assume that as the numbers of students rise, the percentage of female students will automatically rise and the shortage will take care of itself. The ACM report strongly urges direct action to attract and retain more women to computing in all points in the computer science pipeline, that is, K-12, undergraduate, graduate, faculty and industry (Camp et al., in press).

Purpose of the Study

The percentage of female students enrolled in the beginning computer science programming courses at a southern community college parallels that trend. The percentage of female computer programming students traditionally decreases significantly as students move through the three-course sequence of Introductory Programming, Programming I, and Programming II. Since 1992 the average number of females enrolled in the introductory programming classes is 6, and the average number of females who complete the course is 3. The average number of females that enroll in the Programming I class is about 4, while the average number of females who complete that course is 3. The average number of females who enroll in the Programming II class is about 3, while the average number of females who complete that course is 2. If the number of female students enrolled in the first three courses of computer programming could be increased, the increase would generate a higher percentage of female students graduating from the community college to the university computer science programs, and thus supply more females to the job markets in computer science.

Many studies have investigated why females drop out of the computer science discipline as their major. This study investigates some of the reasons that female computer science students choose to retain computer science as their college major.

Before 1990 the isolation of women in post-secondary computing education meant that women were dropping out of graduate programs, or entering industry, rather than enduring the patronizing behavior, doubted qualifications, and invisibility that many encountered in academia. In industry women found similar obstacles. Problems of access to computers were addressed in this early research. In general, girls were found to have less access to computers than boys did, both in school and at home. This problem was exacerbated for girls from minority groups, particularly girls and women of color. Early research also found sex bias in educational computer software and sex stereotypes associated with the hacker culture. Partly because of their lack of access and experience compared to that of boys, girls were generally found to have less self-confidence with computers and attributed their successes with computers to luck and their failures to lack of competence (Dryburg, 2000).

Since 1990 post-secondary research has focused attention on stereotyping, attitudes toward computing, gender grouping and social facilitation, experience, computing culture, and motivation (Dryburg, 2000). Studies generally agree that computer scientists are stereotyped as male, very smart, antisocial, and content to sit in front of the computer for long periods of time. Some studies have found that students' lack of interest in pursuing CS degrees is related to these stereotypical perceptions of computer scientists. Studies on post-secondary students' attitudes to computing generally do not use the same attitude measures. Different measures and non-random samples may help explain some contradictory findings among studies on attitudes toward computing. Prior computer experience is associated with greater success in computer education at the post-secondary level. All studies indicated either that females are less prepared than males entering CS or that they believe themselves to be less prepared. Female attitudes were also found to be more positively associated with any form of previous computer experience than male attitudes. The social facilitation associated with working in all-female groups also appears to have a positive influence upon females' performance at the computer. CS culture has been explored as another factor potentially deterring women from entering the field. In general, it appears that women in CS accept the masculine culture and adjust their expectations and aspirations accordingly, whereas women outside CS are deterred from entering by the anticipation of male hostility (Dryburg, 2000).

This study investigated many of these issues in an effort to ascertain reasons that computer science females stay in the computer science program. Many more females begin their college careers in the field of computer science than ultimately graduate in that field. External factors may contribute to the failure to retain female students in the computer science program. However, internal factors may play a large role. This study investigated both external and internal factors.

Theoretical Framework

Learning computer science has been compared with learning mathematics. Both disciplines use the human minds in a way that greatly increases the power of human thinking. If this view is correct, then students cannot succeed in learning mathematics or computer science unless they are taught in ways that enable them to bring their intelligence, rather than rote learning, into use for their learning of these subjects. This is very important in today's world of rapidly advancing science, high technology, and commerce (Skemp, 1987). Skemp (1987) further states:

> Much of our everyday knowledge is learnt directly from our environment, and the concepts involved are not very abstract. The particular problem (but also the power) of mathematics lies in its great abstractness and generality, achieved by

> successive generations of particularly intelligent individuals, each of whom has been abstracting from, or generalizing, concepts of earlier generations. The present-day learner has to process not raw data but the data-processing systems of existing mathematics. This is not only an immeasurable advantage, in that an able student can acquire in years ideas that took centuries of past effort to develop; it also exposes the learner to a particular hazard. Mathematics cannot be learned directly from the everyday environment, but only indirectly from other mathematicians, in conjunction with one's own reflective intelligence (p. 18).

This expresses the essence of Skemp's principles of learning mathematics, which are now directly stated.

(1) Concepts of a higher order than those that people already have cannot be communicated to them by a definition, but only by arranging for them to encounter a suitable collection of examples.

(2) Since in mathematics these examples are almost invariable other concepts, it must first be ensured that these are already formed in the mind of the learner (Skemp, 1987, p.18).

There are several consequences of the second principle stated above. In the building up of successive ideas and abstractions, if a particular level is imperfectly understood, everything from then on is in peril. This dependency is probably greater in mathematics than in any other subject (Skemp, 1987). In learning mathematics, although students must create all the concepts anew in their own minds, they are able to do this only by using the concepts arrived at by past mathematicians. There is too much for even a genius to do in a lifetime! This makes the learning of mathematics, especially in its early stages and for the average student, very dependent on good teaching (Skemp, 1987).

The formation of a single concept is embedded in a structure of other concepts. Each one, except primary concepts, is derived from other concepts and contributes to the formation of yet others, so it is part of a hierarchy. The study of these structures themselves is an important part of mathematics, and the study of the ways in which they are built up and function is at the very core of the psychology of learning mathematics (Skemp, 1987). The more structures, called schemas, students have available, the better their chance of coping with the unexpected. The schemas that students build up in the course of the early learning of a subject are crucial to the ease or the difficulty with which they can master later topics. New experiences that fit into an existing schema are much better remembered, and those experiences that do not fit in are largely not learned at all.

Both the study of mathematics and the study of computer programming are sequential in nature. New concepts and ideas are built on preceding abstractions. To understand something, then, means to assimilate it into an appropriate schema. Therefore, inappropriate early schemas will make the assimilation of later ideas much more difficult, perhaps impossible, to learn. This has very important consequences for the beginning computer science programming courses.

There is another concept that Skemp teaches about learning mathematics that has a direct impact on retention. On the surface, people communicate by exchanging words or symbols. These symbols act as conveyors of the deeper meanings or the conceptual structures they are trying to communicate. What teachers try to communicate are these conceptual structures. How they communicate these, or try to communicate them, is by writing or speaking symbols. The conceptual structures are what are most important. These form the deep structures of mathematics. However, only the symbols can be transmitted and received. These form the surface structures. Even within the mind, the surface structures are much more accessible, and to other people they are the only ones which are accessible at all. But the surface structures and the deep structures do not necessarily correspond, and this lack of correspondence causes problems. These conceptual structures, both surface and deep, can act as attractors for incoming information (Skemp, 1987). Sensory input will be structured, interpreted, and understood in terms of which conceptual structure it activates. In some cases more than one structure may be activated simultaneously, and in others one conceptual structure captures all the input. This means that all communication, written or oral, is necessarily received into the symbol system at the surface. To be understood and learned mathematically, the communication must be attracted to the deeper structures. This requires that the deeper conceptual structure of mathematics be a stronger attractor than the surface conceptual structure. If it is not, the surface structure will capture the input, or most of it. What, then, can make these deeper structures stronger attractors? The surface structures obviously have an advantage – all communicated input must go there first. As for developing the deeper cognitive structures, there is a point of no return. For too many children, the deeper structures are effectively not there. If the deeper structures are absent or weak, all input would be assimilated into the surface structures (the effort to find some kind of structure is strong), and those surface structures will grow at the expense of the deeper structures. In the years' long process of learning mathematics, if these deep conceptual structures are not formed early on, they will never get the chance to develop as stronger attractors. If the deep conceptual structures are absent or weak, the input will be assimilated into the surface symbol system. Learning at this surface level may be easy short-term, but it becomes impossibly difficult long-term because the surface structures lack internal consistency. In contrast, the deep conceptual structures are internally consistent. Of all subjects, mathematics is one of the most internally consistent and coherent. Therefore, if the deep mathematical structures become well established, they will attract much of the input. Teaching material in mathematics can be sequenced in such a way that new material is presented which always can be assimilated conceptually to those deeper structures (Skemp, 1987).

What, then, are the implications for computer science? The first courses in computer programming are strategic in their importance to the long-term success of students. The beginning courses in computer science set the stage for learning the deep conceptual structures onto which all the rest of the concepts of computer science will be attached. Since communication is by the utterance of symbols, especially in computer science, all the communication first goes into the students' symbol system. To be understood relationally, it must be attracted to an appropriate conceptual structure. What is more, the input must be interpreted in terms of the relationships within the conceptual structure, rather than within those of the symbol system. This requires that the conceptual structure be a stronger attractor than the symbol system, and that the connections between the symbol system and the conceptual structure be strong enough for the input to go easily from the symbol system to the deep conceptual structure. It is this construction of the deep conceptual structure that should occur during the first computer programming courses.

It is to be noted that, for the conceptual structure in computer science, there is a point of no return. In the process of learning the beginning concepts of computer science, if these structures are not formed early, they will never have the chance to develop as attractors. The effort to find some kind of regularity is strong, and if these deep conceptual structures are absent or weak, the input is assimilated into the surface symbol system. This increases surface knowledge, but when the more complicated concepts of computer science come along, there is no deep structure onto which they can be attached. This results in an inconsistent understanding of basic computer science concepts, and in the long term it is difficult to learn and retain these concepts. Part of the answer, according to Skemp, is to sequence the material carefully in such a way that new material can be assimilated conceptually, that is, into the deeper conceptual structures. Because of these principles of learning, this study considered the beginning courses in computer science and how the female students successfully constructed the deep conceptual structures of computer science.

Implicit in this investigation of learning is the question of learning styles. Gardner (1991) has laid the foundation of research of multiple intelligences. He asserts that there are several types of learners – the intuitive learner, the traditional student and the disciplinary expert – each of whom operates in accordance with several constraints and demonstrates his or her understanding in characteristic types of performances. The intuitive learner reflects neurobiological and developmental constraints, constraints owing to species membership and to principles of human development that operate predictably in physical and social environments around the world. These constraints are very powerful, and they prove very difficult to dissolve (Gardner, 1991). For the traditional student the constraints are of an extrinsic sort – the historical and institutional constraints that are embedded in schools. One cannot begin to master a domain, or to understand it, unless one is willing to enter into its world and to accept the disciplinary and epistemological constraints that have come to operate within it over the years. Taken together, these constraints place severe limitations on what students can learn in educational settings and how they can achieve understanding (Gardner, 1991). Since scientists cannot peer into the mind or the brain, understanding has come to be linked with performances that reflect that understanding.

Gardner mentions three varieties of performance. The first is the performance of intuitive, naïve, or natural understanding, which is often immature, misleading or fundamentally misconceived. In the school context, educators have ordinarily sought and accepted rote, ritualistic, or conventional performances. These performances occur when students respond by simply spewing back the particular facts, concepts or problem sets they have been taught. To these rote performances, Gardner contrasts performances of disciplinary or genuine understanding. Such performances occur when students are able to take information and skills they have learned in school or other settings and apply them flexibly and appropriately in a new and at least somewhat unanticipated situation. Disciplinary understanding is always changing and is never complete; expertise is manifest when an individual embodies his culture's current understanding of the domain (Gardner, 1991).

Gardner extrapolates from this that students possess different kinds of minds, and therefore learn, remember, perform, and understand in different ways. There is ample evidence, he says, that some people take a primarily linguistic approach to learning, while others favor a spatial or a quantitative tack. Some students perform best when asked to manipulate symbols of various sorts, while others are better able to display their understanding through a hands-on demonstration or through interactions with other individuals. Gardner proposes that all human beings are capable of at least seven different ways of knowing the world, ways he terms the seven human intelligences. According to Gardner's analysis, all humans know the world through language, logical/mathematical analysis, spatial representation, musical thinking, the use of the body to solve problems or to make things, an understanding of other individuals, and an understanding of oneself. Where individuals differ is in the strength of these intelligences, the profile of intelligences, and in the ways in which such intelligences are invoked and combined to carry out different tasks, solve diverse problems, and progress in various domains. People do learn, represent, and utilize knowledge in many different ways, and they learn in ways that are identifiably distinctive. The broad spectrum of students would be better served if disciplines could be presented in a number of ways and learning could be assessed through a variety of means (Gardner, 1991).

While the recognition of different ways of representing and acquiring knowledge complicates education in certain ways, it is also a hopeful sign. Not only are the opportunities of acquiring understanding enhanced if multiple entry points are recognized and utilized, but the way in which understanding is conceptualized is broadened. Genuine understanding is more likely to emerge and be apparent to others if people possess a number of ways of representing knowledge of a concept or skill and can move readily back and forth among these forms of knowing. No one person can be expected to have all modes available, but everyone ought to have available at least a few ways of representing the relevant concept or skill (Gardner, 1991).

With this in mind, this study investigated whether the female students who remain in computer science have different learning styles from those female students who, for whatever reason, decide to change to another major. Studies of cognition suggest that there exist many different ways of acquiring and representing knowledge. Different students may be reached in quite different ways, and it is possible that those students who remain in computer science have a particular combination of learning styles that are different from the styles of those who choose to opt out of computer science.

Research Questions

This study will explore the following research questions:

1. What are the characteristics of the female students who have chosen computer science as their college major?
2. Specifically, how do they perceive that they have constructed their internal knowledge of key concepts and ideas in computer programming?
3. Do female students who major in computer science have a different combination of learning styles than those female students who have dropped out of computer science?

The computer science program at the community college is primarily geared toward a university programming degree. The computer programming course sequence is comprised of Basic Programming, Programming I, and Programming II. Since content mastery of the beginning courses of computer programming greatly influences the future success of the students, this study will investigate those female students who have successfully mastered one or more of these courses. In order to provide supporting data, this study will also explore reasons that female students changed from computer science to another major.

Based on this researcher's experience over the last eight years of teaching college level computer science, there are certain hallmark concepts that must be mastered at particular points in the curriculum. Mastery of these concepts indicates that deep conceptual structures have been constructed. The female students who have successfully completed one or more computer science courses have been successful at constructing deep conceptual structures in the area of computer programming. This researcher has observed that those students who apparently master these hallmark concepts taught in Basic Programming are well able to succeed in Programming I. This researcher has also observed that the students who apparently cannot master the hallmark concepts in Basic Programming do not proceed to Programming I, and these students usually change their major to another field.

For example, in the community college, of the 24 female students enrolled in the Basic Programming classes during the fall semester of 1999, seven earned an A or B, two earned a D, and the rest either withdrew, audited, or were dropped. Of the seven who successfully passed the class, three have continued in computer science, two needed just one course for their employment, and two are majoring in another field. All those who earned a D, an F, changed to an audit, withdrew, or were dropped chose not to repeat the course, and if they had declared CS as their major, changed to another major.

Of the thirteen female students enrolled in the Basic Programming classes during the spring semester of 2000, eight earned an A, B or C; the other five either withdrew or dropped. Of those eight who successfully passed the course, one has continued in computer science. The rest have moved away, changed their major, or for some other reason decided not to pursue computer science.

From a review of the literature, as seen in Chapter II, several areas of interest have surfaced. This study will examine six of those areas and seek to answer questions in hopes of discovering some of the success strategies in computer programming of these female students. These areas include instructional environment, social interaction, content difficulty perception, complexity issues, persistence/satisfaction, and motivation. The instructional environment includes questions about teaching styles, teaching strategies, learning activities, learning environment, male/female ratio of students, and the general atmosphere of the classroom. Social interaction includes questions about study and work by self or in groups, introvert/extrovert tendencies, teamwork, and clarification/help issues. Content difficulty perception includes questions about previous computer use, difficulty ratings of the different courses, and where the students went for assistance. Complexity issues deal directly with course content, the topics that were most difficult, and coping strategies. The persistence/satisfaction area includes questions about whether content difficulty is a deterrent or an incentive to achievement, confidence and ability levels, and persistence. The motivation area includes questions about the initial choice of computer science as a major, motivation for continuing, future plans, and likes/dislikes.

Limitations/Delimitations

This study is limited to freshman and sophomore female computer science students in a community college located in a suburban region of a southern state. These female students were currently enrolled in either the Basic Programming, Computer

Programming I, or Computer Programming II course for the spring semester of the year 2001. The study was conducted toward the end of the semester, when those who were going to drop out had already done so. Five female students participated in this study: three who were enrolled in the Programming I or II classes, one who was enrolled in the Basic Programming class, one who had previously failed the Programming I course, and one who changed her major.

One other limitation to this study needs to be mentioned. At this point there is no clearly defined relationship between Skemp's theories of learning mathematics and learning computer science, although the two disciplines are clearly interrelated and intertwined.

Definition of Terms

CS is defined in this study as computer science.

Retention is defined in this study as a successful completion of the previous programming course and current enrollment in the next course of the computer programming course sequence.

Successful completion is defined in this study as earning an A, B, or C in the previous programming course.

Assumptions

This study assumes that the subjects of the study will respond honestly to questions posed both in the individual interviews and in the focus group.

For the purposes of this study, it will be assumed that the principles of learning mathematics, as pioneered by Richard Skemp, also apply to learning computer science. The disciplines are clearly interconnected, and it would appear logical and reasonable that Skemp's theories are consistent with the principles of learning computer science. Both disciplines depend upon a hierarchy of knowledge being learned in a particular order, with future success depending upon the previously learned principles.

Justification of the Study

The low level of participation of females in computer science courses is a continuing concern for educators and industry. Computer science is the only field in which women's participation has actually decreased over time. In fact, women, who make up half of the workforce, account for only twenty per cent of those with information technology credentials (Tech-Savvy, 2000). Women are an untapped source of talent to lead the high-tech economy and culture. This study further uncovered some strategies to encourage female retention in computer science.

CHAPTER II

REVIEW OF RELATED LITERATURE

Background

In October 1997, Dr. Tracy Camp, Assistant Professor at the Colorado School of Mines, published a paper that summarized a phenomenon known as "The Incredible Shrinking Pipeline," referring to the ratio of women involved in computer science (CS) from high school to graduate school. As the total number of BA/BS degrees awarded in all disciplines has continued to increase from 1980 until 1994, the percentage of the recipients who were women has also increased. In computer science, however, the percentage has decreased. (Camp, 1997)

The percentage of bachelor's degrees awarded to women has increased almost every year since 1980, while the total number of BA/BS degrees awarded in CS to women has steadily declined since 1984. This is alarming news, since the percentages of BA/BS degrees awarded to women in disciplines similar to CS has increased over the same period of time (NCES, 1996). While the number of bachelor's degrees awarded in CS in 1993-94 is almost equivalent to the number of degrees awarded in CS in 1982-83 (24, 200 versus 24,510), the percentage of degrees awarded to women is drastically smaller (28.4% versus 36.3%). Even though more women are being awarded bachelor's degrees, and even though the percentages of bachelor's degrees awarded to women in disciplines similar to CS has increased, the percentage of bachelor's degrees awarded in CS to women has decreased (Camp, 1997).

Alarmed by these statistics, the Association of Computing Machinery (ACM) launched a one-year study in 1998 to investigate the sharp decrease in women earning computer science degrees ("Computer Gender Gap," 1998). Citing more current statistics, this report found a disturbing trend that is unique to computer science. The proportion of bachelor degrees awarded to women in all disciplines has continued to increase almost every year for decades, to a high of 55.2% in 1995-96. This has had a positive effect on the percentage of women earning degrees in all science and engineering fields except CS. In CS, the percentage of bachelor's degrees awarded to women decreased almost every year from 1983-84 (37.1%) to 1995-96 (27.5%). During the same period, the percentage of bachelor's degrees awarded to women in biological/life sciences, engineering and physical sciences increased by 12.6%, 25.8%, and 30.4% respectively (Camp, Miller, & Davies, in press). The ACM report predicts only a nominal increase in the percentage of women in CS, followed by another decrease (Camp, et al., in press).

Women pursue educational and occupational careers in computer-related fields less frequently than men do. In recent years, the percentage of females studying computer science at various educational levels has declined nationwide, at a time when the proportion of women at the college level outnumbers the proportion of men. As the statistics show, the percentage of females entering the job market is increasing much faster than that of males. If the United States is to maintain its competitive position both nationally and internationally, it must educate more women to be competent in scientific and technical fields (Shashaani, 1997).

Over the past decade, educators have begun considering the issue of gender equity and its impact on their classrooms. In 1992 the AAUW Educational Foundation published *How Schools Shortchange Girls*, calling national attention to the barriers girls faced in America's schools. That report documented disturbing evidence that girls confronted gender bias and stereotypes in the classroom. Teachers and school administrators began incorporating gender equity issues into classroom materials. Educators began paying greater attention to the different ways they interacted with girls and boys (Weinman & Cain, 1999).

Six years later, the approach of the new millennium invited a new look at the status of girls in school. What was different? How had new educational reforms and new technologies impacted girls in school? The 1998 AAUW report, *Gender Gaps: Where Schools Still Fail Our Children,* is a synthesis of one thousand studies and was researched by the American Institutes for Research, based in Washington, D.C. This report found that girls still make up only a small percentage of students in computer science classes. While boys program and problem solve with computers, girls tend to use computers for word processing, the 1990's version of typing. Furthermore, boys are still more likely to enroll in advanced computer sciences courses: in 1996 only seventeen per cent of advanced placement test takers in computer science were girls (Weinman & Cain, 1999).

This lower level of participation of females in computer science courses has become a continuing concern for educators. A number of different reasons for this lack of participation have been suggested, including differences in attitudes, gender bias, role models, differential treatment by teachers and classmates, patterns of computer use, availability of computers and their associated subject areas. The literature has much to say on these issues.

Computers as a Male Domain

In a paper presented at the annual meeting of the American Educational Research Association in Boston, Massachusetts, in April of 1990, Bernard Arenz and Miheon Lee cited a number of studies that indicated most of the students working with computers at the secondary school level are male. They added that, as age increases, female enrollment in computer classes declines (Arenz & Lee, 1990). They also noted that there have been mixed opinions and research findings concerning gender equity in computer classes in secondary schools, and that where gender-related differences in secondary schools could be found, the factors that contribute to the problem are not yet fully understood. Arenz and Lee (1990) conducted three studies, two involving secondary school students and one involving middle school students, which examined potential factors affecting computer course participation. Differences in role models, attitudes, interests, and computer use between genders as well as within gender were found to exist. These differences may contribute to the continuing stereotype of computers as a male domain. This stereotypical sex bias of males and computers was found to be higher for males than for females, with a slight increase from first to second courses. Even in the middle school grades, males demonstrated more gender stereotyped views than females (Arenz & Lee, 1990). It is interesting to note that, in this study, the perception of computers as a male domain is high for beginning females and nonexistent for intermediate females, while the same perception increased for males. One student wrote that the low participation of females in the intermediate course is proof of a difference

(Arenz & Lee, 1990). The presence of friends who are computer users at home appeared to have a differential effect on males. Those with male friends who use computers at home were more likely to be computer course participants and be interested in taking a computer course. A comparable relationship was not found for females (Arenz & Lee, 1990). Hours of computer use also differentiated males and females. Males reported significantly more computer use than females in terms of the amount of time spent on the computer, especially at home and at school.

Software may be partly responsible. Software caters to violence, competition, and bloody adventure; when there are women heroes, they are scantily clad and buxom. In arcades the games are of the same sort, and a greater percentage of boys are playing them while a greater percentage of girls are watching and admiring the play (Rothschild, 2000).

Lack of Previous Computer Experience

Another study by Bunderson and Christensen (1995) at a large western university found that a key factor influencing the high rate of female attrition might be lack of previous experience with computers before entering the program. This study also found that negative experiences in the classroom and with peers reinforced the gender bias and may have contributed to the attrition rate. Their responses were grouped into five categories:

(a) Satisfaction with computer science classes and computer science as a career. Results – More females than males expressed dissatisfaction with the major. Former female computer science majors were less likely to enjoy computer science and/or feel comfortable being a computer science major, and more were likely to list these reasons as important factors in deciding to switch majors.

(b) Gender discrimination toward females. Results – Less than one in five female students felt that she had been treated differently in her computer science classes because of her gender. Eighty-nine per cent of the females thought females had as much innate ability as males to learn to use and program computers.

(c) Previous experience with computers. Results – Almost half of the respondents thought the computer science department was oriented toward students with previous programming experience. This response was given by both males and females, and by students at all levels of class standing.

(d) Effects of the traditional culture on retaining female students in the computer science department. Results – Few students listed the influences of the predominant religion as reasons for the low percentage of females in the major.

(e) Interactions with others. Results – As class standing increased, students seemed to place greater value on interacting with other students. Students in advanced classes felt more comfortable asking questions than did those in the beginning classes. Females in all classes found it more difficult to ask questions in class than did men. Furthermore, many females changed from computer science to other majors because they did not perceive computer science as being people oriented.

Computer Usage by Gender

Studies over the past ten years show that males and females differ in terms of their access to and use of computers. Kirkpatrick and Cuban (1998) compiled a synthesis of research findings on gender differences in the field of computers. Kirkpatrick and Cuban found that males and females differed in their computer use with regard to when, where, and how they used them. They found that the "when" disparities exist along three dimensions. First, males take more classes in school in which computers are used than do their female counterparts. Second, males use computers more frequently in their recreational time. Third, males not only use computers more often but also stay on them for longer periods of time. Kirkpatrick and Cuban found that another distinction between male and female computer use revolves around where computers are used. Citing Shashaani (1994), they found that the majority of females learned how to use computers at school, while the minority of males learned computers at school and that home ownership of computers was significantly lower for females than for males. Kirkpatrick and Cuban (1998) summarized the research findings with this:

> Most studies have found the gap between male and female achievement and attitude toward computers was small in the early grades but males were at significantly higher achievement levels by grade twelve. Similarly, studies have shown that, with age, males are increasingly more confident in their ability to use computers and more positive in their attitude toward computers than are females. Studies have also found differences in attitudes and achievement levels correlated to types and amounts of computer use. Differences in achievement levels most often have reflected differences in types and amount of use, not any inherent difference in preferences or ability. In fact, the more time students spend learning on computers, the fewer differences there were in male and female achievement (p. 57).

Social and Societal Factors

Several social and societal factors have been identified that contribute to the gender difference in the use of computers:

1. There appears to be institutionalized differentiation between male and female students by teachers and guidance counselors.
2. Parents encourage sons more than daughters to take computer classes and to excel in this field.
3. Males typically work with computers. Brothers and fathers, not sisters and mothers, use the computer at home.
4. More male computer-using role models exist in peer groups than female computer-using role models.
5. Computers and computer programs are culturally perceived as male enterprises.
6. Computers are often associated with math classes and math skills since math teachers often teach the computer classes in the higher grades (where the disparity between the genders is greatest), and computer labs are often located in math departments.

It is clear that self-confidence is related to achievement. Studies have shown that females are less confident in their math skills than are their male counterparts. So, if computers are associated with math and females are less confident in their math abilities than their male classmates, then females' lack of self-confidence and the resultant smaller enrollment of females in higher-level computer classes can be partially explained.

Family, school, and the larger culture shape female attitudes to limit their access to and use of computers. Attitude is also a result of previous computer experience. Experience with computers is necessary for a more positive attitude toward computers, yet a positive attitude is necessary in order to engage students in working with computers.

Previous studies have suggested that female students have fewer interactions with teachers and thus receive less attention (Ray-McCutcheon, 1996). Females also tend to benefit more from cooperative activities and activities that are less rushed than male students do (Ray-McCutcheon, 1996). Males, on the other hand, receive more teacher interaction, including more questions of higher order. However, McCullough (1996) found that female student participation rates are higher than or equal to the male student participation rates. She also found that female participation levels were slightly higher in female-instructed classrooms than in male-instructed classrooms, and that male students participated slightly more in male-led classrooms than in female-led classrooms. Her data show another trend of female students receiving more attention from the instructor than in the past.

Other studies reflect that course selection in science and mathematics may be related more to gender than academic abilities. Current research suggests that this behavior pattern is seen across all ability levels, begins as early as primary school, and continues through postgraduate education (Joyce, 2000). Biographies and autobiographies of women mathematicians indicate that many women who elect mathematics have backgrounds in a family or other small community that celebrates mathematics (Damarin, 2000). Support groups provide sharing of insights and study of common information, activities in support of group members in time of crisis, and general sharing of particular events that counter the normal silent and alone state of women in marked categories such as mathematics and computer science (Damarin, 2000).

The problem is not limited to the United States. Durndell and Thomson (1997) found in their ten-year study that males reported significantly more use than females of their own computers and of a friend's computer, while computer use at school showed no gender differentiation. Their study also reported gender bias in attitudes toward computing, still with a strong stereotype of the computer specialist, including a lack of interaction with other people. The issues underlying this study have not faded very much over the last decade. There is still international concern about the relationship between gender and science and technology. Considerable research interest also continues to be shown in this area in terms of studies of computer-related attitudes and motivations in relation to gender (Durndell & Thomson, 1997). These authors conclude that gender equality, in the sense of having no difference between males and females in attitudes towards, knowledge about, and use of computers, when looked at over the decade under study, appears to be a long way off.

The difference in attitude is not unique to this country. A study of high school students in Yugoslavia confirmed that males showed a more positive attitude toward computers than females, even when computer experience was controlled (Kadijevich, 2000).

Confidence in Using the Computer

Shashaani (1997) found in her study of college students that males scored higher than females in liking to learn about computers, enjoying work with computers, and considering the computer exciting. With regard to confidence, females felt more uncomfortable with computers and feared them more. They believed that the computer is hard to learn and had less confidence than males in dealing with it. The gender differences were significant in three areas: computer liking, confidence, and stereotyping. Males were found to have more experience with computers than females, including courses in high school. Males also reported taking more programming courses than females did, and spending more hours per week in computer labs. Moreover, a higher percentage of males than of females reported having a computer at home. Shashaani did not find any significant difference in parental attitudes toward computers by gender, but she did find a positive correlation between students' prior computer experience and their attitude toward computers. Students who had knowledge about and experience with computers expressed more interest in computers, had more confidence in their ability to work with computers, believed more strongly that the genders were equal in computer ability, and agreed more strongly that the computer is useful for the individual and society. Both male and female students who received more encouragement from their parents to become involved in computers were more interested in computers, had greater self-confidence, and were more aware of the benefits of computers in their daily life. The positive correlation between computer attitudes and experience revealed that students who know more about the computer used computers more. Students who had more access to home computers were also more interested in computers and had more confidence in working with them (Shashaani, 1997). Shashaani (1997) also found that a one semester computer training course enhanced both male and female students' attitudes toward computers and could increase their frequency of use. By the end of the semester, students were more interested in working with computers and were more confident about their own ability to use them. The gender gaps were not completely eliminated, however; female students still had less positive attitudes toward computers than male students did. The results of her study support the argument that familiarity and knowledge increase interest in computers and reduce anxiety and lack of confidence in dealing with them.

Personality Traits

In England, Charlton and Birkett (1998) compared characteristics of students taking programming-oriented (computing) classes with those of students taking application-oriented (business) higher education classes. They found that programming-oriented students exhibited more schizoid personalities in terms of greater introversion and (for the males) independence. The findings for introversion were consistent with the idea that the computing students were more likely to have object-centered, rather than person-centered, interests (p. 175). This fact is likely to explain the greater computer involvement as indicated by greater computer engagement and programming experience. In addition, the study found significant correlation between computing aptitude and introversion. The schizoid personality construct (introversion) might apply primarily to the males. While computing females were more introverted than the business females, they did not exhibit greater independence or tough-poise. The data also indicated that the greater degree of computer involvement was particularly important in determining the successful completion of the two-year course of study. The study concluded that previous programming experience was implicated in explaining the gender differences in computing course choice and that teaching programming in a non-mathematical context would be particularly helpful in stimulating female interest (Charlton & Birkett, 1998).

In their 1999 study, Charlton & Birkett defined the term schizoid, which describes solitary individuals who tend to divorce emotional from intellectual considerations. They made the notation that both introversion and tough-poise are components of the schizoid typology. Previous research, which supported their hypothesis that schizoid personalities lead them to concentrate much energy upon intellectual matters and attach importance to intellectual prowess, was cited in their study. Such reasoning led them to hypothesize that there is a direct link between both greater introversion and greater tough-poise with better intellectual performance (p. 240). However, they found that positive female computer attitudes, rather than possession of personality characteristics more stereotypical of males, had a bearing upon females' enrollment in the present course. They also found that females displayed a real interest in the subject and did not enroll because of enhanced employment prospects. They suggested further research to determine why some females develop highly positive computer attitudes.

Same Gender Classrooms

Both Melissa Koch (1994) and Anita Revilla (1998) quote Myra and David Sadker's book *Failing at Fairness: How Our Schools Cheat Girls* (1994) in listing gender equities in the public school systems. Two suggestions made in these articles are for same-gender classrooms where there is no competition with male students for computers and for beginning computer training early in the upper elementary and middle school years. For girls many feelings of incompetence and alienation from technology begin or are reinforced in school, particularly in the middle school years (Koch, 1994). Revilla (1998) reports that girls often enjoy math and attain achievement levels equal to or greater than those of boys at the elementary level. However, by the time they reach high school, many bright girls become disinterested in mathematics, enroll in fewer advanced math classes, achieve lower math scores, and are less likely to choose careers that are math-related (p.12).

It is very widely accepted in the field of middle level education (education of children aged ten through fourteen) that this developmental time period is crucial for boys and girls, and this seems to be the case in regard to computer use (Butler, 2000). Butler (2000) found in her research that seventh grade is the point at which boys generally increase their use of computers compared with girls. Butler (2000) also reported that sex differences in attitudes toward computers are strongly established by grade eight.

The results of a study by Crombie and Armstrong (1999) indicate that an all female computer science high school class can enhance the learning experiences of female adolescents. Females from the all-female class report significantly higher perceived teacher support than both males and females from mixed-gender classes. The higher levels of perceived teacher support reported by females from the all-female class may be related to their learning experiences in a classroom without male students. Both females from the all-female class and males reported similar levels of computer confidence, intrinsic motivation, and future academic intentions, and both groups reported higher levels than females from the mixed gender classes. These results indicate that an all-female computer science classroom environment may enhance the learning experiences of female students (Crombie & Armstrong, 1999).

Across all stages of education, direct or indirect benefits of all-girl groupings have consistently been found. At the elementary level, all girl groupings improve task performance and other behaviors, such as asking for help and staying on task longer. Studies at the secondary stage find that girls' attitudes are more positive in all-girl groupings than in mixed groupings of students. At the post-secondary stage, the benefits of all-girl groupings relate to social facilitation, where women encourage one another in their computer work when grouped together (Dryburgh, 2000).

"Hard" Computing versus "Soft" Computing

In the late 1990s, researchers focused on how girls interacted with the computer in different ways than boys did. These studies often sought to arrive at strategies for making the world of computers friendlier for girls (Butler, 2000). The late 1980s and early 1990s saw some promising projects that were developed by states, school corporations, or institutions of higher education in collaboration with middle schools. Whether short term or not, these projects often helped young girls to develop better attitudes and skills and broader visions about a future with technology. Those projects included Ohio's Yes, I Can: Action Projects to Resolve Equity Issues in Educational Computing, the Girls into Science and Technology Program in England, Virginia's Adventures in Technology = Options in Math and Science (ATOMS) and The Regional Summer Math and Science Institute (RSI) science, math, and technology "pipeline" program also in Virginia·(Butler, 2000).

Also in England, Clegg and Trayhurn (2000) proposed that computing involves multiple practices in different concrete settings, and they have contributed to the growing body of research that is trying to move beyond the old debate about women and computing. They define computing as a set of disparate and complex practices and technologies that include the applied as well as the theoretical. Over the decades, academic institutions have characterized computing as a reliance on mathematical formalism, emphasizing such areas as artificial intelligence, formal methods, and computer vision. They report that in many cases end-users are denigrated as not "real" computer people. Clegg and Trayhurn believe this emphasis is misplaced and computing should include end-user type applications.

Research to date has found almost universally that women are more attracted to computer courses that emphasize social issues and computer applications than to traditional science-based computer courses (Henwood, 2000). In Henwood's study, women constituted just twenty per cent of all computer science students but over fifty per cent of interdisciplinary students. He also found that women students tended to underestimate their own technical skills, both relative to other measures of their technical competence and relative to equally competent men. Also, women's expertise was less likely to be recognized by other students than men's expertise (Henwood, 2000).

The linking of computers to formal methods and artificial intelligence is too limited a vision of what computing is. The Clegg and Trayhurn (2000) study confirmed that women in different areas of computing appeared not to be lacking in either confidence or skills. The problem, they stated, is that the computing that men do ("hard coding") counts, but that the computing that women do (often working with large data sets and systems) does not count and so contributes to the gender stereotyping that prevails in the academic world.

The existing canon in the computer science curriculum places an inappropriate emphasis on "hard" areas such as mathematical formalism. More incorporation of arts-based skills and skills traditionally associated with women would actually result in a curriculum more likely to meet the needs of the computing and information technology industries. There is a myth that working as a computer scientist and working with people are mutually exclusive. This "soft" area involves contact and concern with users and is usually regarded within the computing profession as being easier and somehow less "pure" than the more technical areas (Mahoney & Van Toen, 1990). A "Women and Information Technology" report, based on information from the British Computer Society, predicts that most of the information technology (IT) jobs of the future will "demand business aptitudes, people-oriented skills and multi-tasking management potential more than technical ability" (Mahoney & Van Toen, 1990). The courses in which more emphasis is placed on the social, business and communications aspects of computing are being found to attract more women. Degrees that combine business, social sciences, or humanities with computing are also being found to have much more attraction for women than single computer science degrees (Mahoney & Van Toen, 1990). Mahoney and Van Toen (1990) conclude that it is these types of courses that will breed success in turning out graduates who will offer skills that meet the real needs of the current computing industry.

Gender Bias and Stereotyping

Conscious or unconscious gender bias does exist in many of the cognitive and affective classroom interactions between teachers and students (Gipson, 1997). Gipson reported in her Educational Technology article that studies reveal that teachers speak to boys more frequently and boys are asked higher-order thinking questions more frequently than girls are. In addition, it appears that primary school teachers tend to praise boys for the quality of work and girls for form and neatness. In project work teachers more often provide boys with instruction on how to complete the project, while teachers tend to show girls how to do it, or even do it for them (Gipson, 1997).

Margolis, Fisher and Miller conducted a four-year study in the computer science department at Carnegie Mellon University (2000). They focused on the process by which women students who enter with high enthusiasm and interest in computing quickly lose both faith in their ability and interest in the subject. They found that women are concerned with the usefulness of computers and that their interest in computers developed gradually among many other interests when they were growing up. This study found no correlation between prior computer experience and success. They did find, however, that a wide range of psychological, pedagogical, and socialization related factors contributed to the women's perceiving themselves as "picking up ideas slower" and perceiving their male counterparts as being "so good with computers without even trying." They cite studies that show that even with the same grades, women in science showed less confidence in their abilities and appeared more depressed about their academic performance than men did.

Other studies have found a similar pattern of the unraveling of undergraduate women's confidence during the first year of college (Margolis et al., 2000). Margolis et al. (2000) report:

> Twenty per cent of the female computer science majors we interviewed have questioned whether they belong in computer science, because they feel they do not share the same intensity of focus and interest they see in their male peers. Women describe wanting to talk about other things besides computers, feeling estranged from those who are myopically focused on a machine (p. 113).

Research shows that both males and females believe that males are better at computing. Comments from male peers, which may appear incidental or random, accumulate to make women feel undervalued and unwelcome (Margolis et al., 2000). The MIT Artificial Intelligence Laboratory report "Why are there so few female computer scientists?" concludes that such comments and behaviors are the symptom of a more fundamental problem: lower expectations for females (Margolis et al., 2000).

Women's under representation in technical fields is not primarily due to direct discrimination, but to subconscious behavior that tends to perpetuate the status quo (Spertus, 1991). This behavior includes the different ways in which boys and girls are reared, the stereotypes of female engineers, subtle biases that females face, problems resulting from working in predominantly male environments, and sexual biases in language (Spertus, 1991).

Prior Computing Experience

Computer achievement is directly related to lack of computer anxiety, which is a significant predictor of achievement in computing (Taylor & Mounfield, 1994). The amount of computer anxiety that an individual has is directly related to the prior experiences of the individual. In fact, Taylor and Mounfield (1994) report that the most reliable predictions of computing attitude and achievement are based on knowledge of the amount of prior computing experiences. Further studies have indicated that the real problem is lack of participation by females, not lack of ability. Given equal opportunity, females do as well as males in computing courses and achievement, but those who initially choose to pursue computing courses or careers are not persisting in these choices.

This lack of persistence starts early in the educational process, and gender by itself is a significant predictor of persistence in computing (Taylor & Mounfield, 1994).

Another study investigated whether student academic achievement in college computer science programs in Taiwan could be predicted by factors reported to be effective in United States studies. Subjects were 940 college students enrolled in five universities offering computer science programs. The close relationship between performance in introductory computer science courses and performance in computer science programs was validated. Model predictive powers were significantly improved when performance in introductory computer science courses was included in the models. Also, female subjects outperformed male counterparts in course performance at both the high school and the college levels (Fan, Li, & Niess, 1998).

Retention of females in computing courses or computing curricula after they have chosen to enroll in the course of study has historically been a problem. Past studies have shown marked differences in the ways in which males and females tend to view their own inadequacies in computing courses. Females tend to blame failure on their own personal inadequacies, while males tend to blame external factors (Taylor & Mounfield, 1994).

Most pre-college computing experiences proved to be significant for the females, but only computer programming courses were significant for the males. In the study of Taylor and Mounfield (1994), eight per cent more of the males who had prior high school computer science coursework were successful than the males who had no such experience. But there was a thirty per cent difference in the female rate of success when compared by prior high school computing experience. For both sexes, computer ownership was the single most significant factor in course success, and females who owned computers were the subgroup in this study with the highest success rate.

In a longitudinal study, all students were provided with network access and laptop computers over a four-year period. Results from this study indicated that women were less positive about computers than men were, and the use level of computers by women was less frequent than that for men. This change in relationship is a throwback to the earlier days of computing, when research indicated that men were more positively disposed toward computers than women were (Mitra et al., 2000).

Learning Styles

Katz, Maitland, Hannah, Burggraf, and King (1999) examined gender and academic program groups on measures of perceived usefulness of computers, course confidence and content, and learning styles. Their study found that overall the undergraduate students demonstrated a strong preference for visual learning (i.e., pictures versus words). In a comparison of learning styles with "comfort in working with computer technology" (i.e., general computer use, Web, software applications), students who preferred detail to overall structure were more at ease with computers. The study also revealed a tendency for visual learners to be more at ease with computers (Katz et al., 1999).

Severiens and Dam (1997) reported that men showed a greater preference than women for the abstract conceptualization mode of learning, while women experienced more anxiety about their study success than men did. They also reported that men seemed to be more often interested in the courses for the qualifications they offered, and women, by contrast, were more often interested in learning for learning's sake. They found in their study that women more often used memorizing and rehearsing strategies; they depended on the teacher or the school to organize their learning processes, and they defined learning more often as taking in knowledge. Men, on the other hand, appeared to be more ambivalent as to why they are studying, and they lacked a certain kind of regulation, more so than women do (Severiens & Dam, 1997).

Philbin, Meier, Huffman, and Boverie (1995) found that traditional educational settings may not be the best learning environment for females. Traditional education is directed towards and appeals more to males since it is primarily abstract and reflective. Females learn better in hands-on and practical settings, emphasizing the realm of the affective and doing. Based on the results of their study, Philbin et al. concluded that if females were watching and feeling, or doing and thinking, they learned best; and if males were thinking and watching, they learned best. Females generally felt that they did not fit in with traditional education learning styles. Teaching a subject in the traditional style (abstract and passive) to females who prefer a more concrete and active approach could contribute to the discrepancy in teaching and learning results, particularly in mathematics (Philbin et al., 1995).

Magolda (1989) found that more women preferred concrete experience than abstract conceptualization and that more men and women preferred reflective observation than active experimentation. She reported that women took less initiative in learning than did men, relied more on authorities, were more prone to collect others' ideas rather than debate opinions, and placed greater emphasis on personal interpretation than did men. She also found that women prefer concrete experience to abstract conceptualization, a result that matches other findings in the literature (Magolda, 1989).

Productivity style theorizes that each individual has a biological and developmental set of learning characteristics that are unique. Productivity will improve when training and instruction are provided in a manner that capitalizes on each individual's learning preferences. This theory is based on the generally accepted concept that individual students at every age level differ in the way they learn new and difficult information. The concept of individual differences is well established in the psychological and educational literature and has been corroborated by the extensive research conducted with the Dunn, Dunn and Price Productivity Environmental Preference Survey (Price, 1993, p. 16).

Productivity style, as a model, embraces several general principles in the form of philosophical assumptions:

1. Most individuals are capable of learning.
2. The learning conditions in which different individuals learn best vary extensively.
3. Individual learning preferences exist and can be reliably measured.
4. Most students are self-motivated to learn when they have the option of using their learning style preferences and experience success.
5. Most teachers can learn to use individual learning styles as a basis for instruction.
6. Use of individual learning style strengths as the basis for instruction increases learning and productivity (Price, 1993, p. 16).

Howard Gardner is the leading advocate of the theory of Multiple Intelligences. He proposed that there are at least nine different kinds of intelligences. These are linguistic, logical/mathematical, spatial, kinesthetic, musical, interpersonal, intrapersonal, existential and naturalistic. Long before Gardner redefined intelligence, Funk and

Wagnalls' definition was rather widely accepted. It described intelligence as being directly related to an individual's ability to perceive; to comprehend meaning; to adapt to new situations; to learn from experience; to seize the essential factors of a complex matter; to demonstrate mastery over complexity; to solve problems; to critically analyze; and to make productive decisions (Dunn, Denig, & Lovelace, 2001).

These thinking skills do not necessarily relate to the talents that some people may have with language, mathematics, spatial manipulation, and so forth. It is possible to be extremely intelligent but to have no developed talent. It is also possible to be extremely talented in any of these areas but to demonstrate only limited intelligence (Dunn et al., 2001). Intelligence is different from and more than talent. When potential has been developed and demonstrated over time, it is recognized as talent. But how humans perceive, comprehend, adapt to new situations, learn from experience, seize the essential factors of a complex matter, demonstrate mastery over complexity, solve problems, critically analyze and make productive decisions is much more closely related to how well they think – which is how most dictionaries define intelligence (Dunn et al., 2001).

Gardner's theory of Multiple Intelligences advocates developing children's innate potential or talent; allowing children's natural talents, intuition and interests to guide them toward learning through comprehension; and providing an environment in which children can learn to think rather than to memorize. The Dunns (Dunn et al., 2001) define learning style as the way in which each person begins to concentrate on, process, internalize, and remember new and difficult academic content. Their model addresses twenty elements that can be classified into environmental, emotional, sociological, physiological and psychological variables. Strongly analytic learners often tend to prefer concentrating in brightly illuminated, quiet, formal seating without breaks or snacks, whereas strongly global learners often tend to prefer concentrating in a softly lit, casual (informal) environment with music, periodic breaks and snacks (Dunn et al., 2001).

The basic difference between learning style theories and multiple intelligences theories is that learning styles are concerned with the differences in the process of learning while multiple intelligences center on the content and products of learning. Some researchers believe that by integrating the theories of learning styles and multiple intelligences, the classroom teacher will be provided with some very practical suggestions that will prove to benefit the students (Snyder, 2000).

Differing Views of Technology

The study at Carnegie Mellon School of Computer Science conducted by Margolis, Fisher, and Miller (1999) presents examples of how women students' passion for computer science, their excitement about the field, is often presented and articulated in terms of the context of larger, often people-oriented issues. This study quotes J.B. Miller's *Toward a New Psychology of Women:*

> It is true that women, like everyone, are motivated out of the well-springs of their own being. In that sense, we all, at bottom, act on what is moving us individually. It is also true, however, that women feel compelled to find a way to translate their own motivations into a means of serving others and work at this all their lives. If they can keep finding ways to do this, they are often comfortable and satisfied – and they do thereby serve others. (Margolis et al., 1999, p. 15).

The feminine perception of technology looks right through the machine to its social function, while the masculine view is more likely to focus on the machine itself.

Thus, a major gender difference in interests in math and science is the perceived usefulness of the computer. Even the mass media have popularized a similar conception, the toy/tool dichotomy – the idea that boys and men see computers as toys with which to play, while girls and women use them as tools with which to accomplish a task. Twenty percent of the female computer science majors interviewed in this study questioned whether they belonged in computer science because they felt they did not share the same type of focus and intensity of interest they saw in their male peers (Margolis et al., 1999). These women spoke of struggling to maintain their own confidence and breadth of interest in the computer science environment where joy, play, fun, and love are wrapped up in intense, singular focus on computing and on the computer itself. This lifestyle is one in which hackers seem to spend nearly every waking hour at the computer, talking incessantly about computers, eating and sleeping in front of the computer. "True" members of this computer subculture can be found at the computer at all hours of the day or night (Margolis et al., 1999).

Many women students are hesitant to join this computer science world in which they sense that the links to other interests in their lives will disappear. The goal is not to devalue or derail some students' single minded pursuit of technical virtuosity, but rather to validate a wider array of ways to think about computer science, to engage students' interest and motivation to become computer scientists, particularly female students. Margolis et al. (1999) found that women who persist in computer science are those who have found a way to get the grades they are satisfied with and are able to reconcile a "different" relationship to computing.

In light of these considerations, this study will investigate how and why the female students have persisted in keeping computer science as their major and, at the same time have maintained good grades without changing themselves into "computer geeks" in the process.

CHAPTER III

METHODOLOGY

Participants

The subjects in this study were selected from the female computer programming students who were currently enrolled for the spring semester of 2001 in Computer Programming I with C++ or Computer Programming II with C++. These students were enrolled at a southern community college in a suburban area, and participation in the study was completely voluntary. Traditionally, small numbers of female students enroll in these programming courses. At the time of this study, there was a total of four female students enrolled in these two programming courses. All four students were included for participation in this study. In addition, one female student who had completed Basic Programming was selected. If more female students had been available from whom to choose, the criteria for selection would have included the following:

1. One female with a firm grasp of the subject matter, indicated by an A average in the programming class.
2. One female with an average grasp of the subject matter, indicated by a C average in the programming class.
3. One female struggling with the content, indicated by a D average in the programming class.

4. One female who had dropped the programming course and had changed her major from computer science to something else. This does not necessarily mean the female was struggling with the content in the programming course.
5. One female with a different learning style, as indicated by the Productivity Environmental Preference Survey (PEPS).

Design

This was a qualitative study designed with the phenomenological approach. Phenomenology declares that multiple ways of interpreting experiences are available to each person through interaction with others and that it is the meaning of experiences that constitutes reality. Phenomenology provides an interpretation of reality that is useful in understanding the human condition. What is produced, then, is a particular rendering or interpretation of reality grounded in the empirical world (Bogden & Biklen, 1998).

This study explored the meaning of certain events and interactions to ordinary students in the computer science program and emphasized the subjective aspects of the participants' behavior. This study also explored the conceptual world of the participants in order to understand what meaning they construct around events in their daily lives pertaining to the computer science program and how that meaning is constructed.

The data were collected by a modified version of standardized, open-ended interviews. The researcher pursued the modified, standardized, open-ended interview format in the early part of the interview and then, time providing, pursued any subjects of interest during the latter part of the interview. Since the interviews with the participants lasted for a limited period of time, the questions asked during the first part of the individual interviews were written out beforehand. This technique also minimized interviewer effects by asking the same questions of each participant, ensuring the interviews were systematic. The modified, standardized, open-ended interview was chosen as the design for this study for the following reasons:

1. The exact instrument used was available for inspection and modification by the dissertation committee;
2. Consistency across questions was maintained by the interviewer's use of the exact same questions;
3. The interview was highly focused so that the participants' time was carefully used;
4. The interviewer was permitted more flexibility in probing and more decision-making flexibility in pursuing any subjects of interest during the last portion of the individual interviews.

By controlling and standardizing the first portion of the open-ended interview, the researcher obtained data that were systematic and thorough for each respondent. Since the data gathered were used for comparative purposes, the issues of legitimacy and credibility were minimized by carefully collecting the same information from each participant who was interviewed (Patton, 1990). The researcher herself interviewed each participant, thus ensuring consistency in questioning techniques across interviews.

The first part of the study consisted of administering the Productivity Environmental Preference Survey (Dunn, Dunn, & Price, 1996), which was given during the latter part of the semester in the programming classes. This comprised the first data set.

Each individual interview followed the modified, standardized, open-ended interview format in the early part of the dialog between interviewer and participant, leaving the interviewer free to pursue any subjects of interest during the latter part of the interview. The participants in this study responded to the questions in their own words to express their own personal perspectives. The purpose of using the modified, standardized, open-ended interview was to understand how the participants view the computer science program, to learn their terminology and judgments, and to capture the complexities of their individual perceptions and experiences (Patton, 1990).

The interviews were videotaped, and the interviewer took field notes. After each interview, the interviewer wrote down observations about the interview itself, allowing time for reflection and data clarification, elaboration, and evaluation. Any ideas and interpretations that emerged following an interview were written down and clearly marked as such.

A good interview lays open thoughts, feelings, knowledge, and experience not only to the interviewer but also to the participant. The process of being taken through a directed, reflective process affects the persons being interviewed and leaves them knowing things about themselves that they did not know, or at least were not aware of, before the interview. There were no risks, physical, emotional or psychological, associated with these interviews. The data will be kept confidential, and informed consent forms (see Appendix K) were presented before the interview. Only the interviewer, and possibly the graduate committee, will have access to the data, and the names will be changed to protect the identity of the participants.

Data Collection

The Productivity Environmental Preference Survey was administered to all students in the programming classes during the last month of the semester. Each student in the classes was asked to complete the Productivity Environmental Preference Survey (Dunn, et al., 1996), which took about ten to fifteen minutes. After that time the female students selected for this study participated in an individual interview that followed the predetermined format, discussing the questions posed on the interview agenda in the order listed (see Appendix C). The female students were interviewed separately and they were asked to keep the interview confidential until the time of the focus group interview. Each interview lasted from forty-five minutes to one hour and was videotaped with the permission of the participant. There was time allowed after each interview for the interviewer to reflect and to write additional field notes. Each interview was then transcribed for data analysis, assigning pseudonyms to protect the privacy of the participants. Additional comments from the field notes were inserted at this time. The interviews were conducted on the campus of the community college in one of the empty computer labs at a time determined by each participant.

After all participants were interviewed individually, a focus group interview session, in which the participants discussed some of the topics already presented by the interviewer, was held. The focus group interview session also had a predetermined format that followed a predetermined agenda (see Appendix D). The interviewer set the guidelines for the focus meeting in terms of protocol and time restraints. After that, the actual questioning portion of the meeting began. First, factual questions were asked to introduce the group members, because not all the participants knew one another.

Second, introductory questions that introduced the general topic of discussion and provided the participants with an opportunity to reflect on past experiences and their connection with the overall topic were asked. Third, transition questions were asked to move the conversation toward the key questions. At this point the participants became more aware of how the others viewed the topic under discussion. Fourth, key questions that drove the study were asked of the participants. These were the questions that required the most attention in the analysis. Fifth, ending questions that brought closure to the discussion were presented. The researcher used the questioning route format for the focus group meeting.

The focus group interview session was conducted in an available community college computer lab and was videotaped. The tape was then transcribed. New ideas emerged from the group that was prompted by the group interaction. Focus groups possess the capacity to become more than the sum of their participants and to exhibit a synergy that individuals alone cannot achieve (Krueger, 1994). The focus group meeting lasted one hour and fifteen minutes.

Instrumentation

To investigate the students' learning styles, the Dunn, Dunn and Price (1996) Productivity Environmental Preference Survey (PEPS) for adults was administered to each individual student. This took from fifteen to twenty minutes to complete and was scored by Price Systems, Inc. The Dunn, Dunn and Price Productivity Environmental Preference Survey for adults has been validated and has been shown to be reliable. DeBello, in his paper presented at the 1989 national conference of the Association for Supervision and Curriculum Development, stated, "Extensive research employing the

LSI has made it the most widely documented assessment instrument. Dr. Curry's 1987 review of 21 different learning/cognitive style models through psychometric analyses reported that the Dunn and Dunn model had among the highest reliability and validity ratings" (p. 4). DeBello (1989) further reported that Ohio State University's National Center for Research in Vocational Education published the results of a two year study of instruments. It reported that the LSI had "impressive reliability, and face and construct validity" (pp. 4-5). Ninety percent of the reliabilities are equal to or greater than .60, as shown in Table 1.

Table 1

Reliabilities for PEPS (N=504, 1/8/96)

Area	r	Area	r
Sound	.86	Several Ways	.67
Light	.91	Auditory Preferences	.81
Warmth	.86	Visual Preferences	.71
Formal/Informal Design	.76	Tactile Preferences	.33
Motivated/Unmotivated	.65	Kinesthetic Preferences	.67
Persistent	.63	Requires Intake	.88
Responsible (Conforming)	.76	Evening/Morning	.87
Structure	.71	Late Morning	.84
Learning Alone/Peer-Oriented	.86	Afternoon	.88
Authority-Oriented	.48	Needs Mobility	.83

Note. From PEPS Manual (p.40) by Gary E. Price, 1993, Lawrence, Kansas: Price Systems, Inc. Copyright 1996 by Gary E. Price, Ph.D. Reprinted by permission.

According to the creators of the instrument, they began with 360 items in the LSI. After conducting a content and factor analysis of each of the questionnaire forms and isolating those that achieved a 90 per cent consistency, they developed the shortened form of the LSI, which has been proven to be both reliable and valid. The LSI is the result of 14 years of school- and university-based research involving more than 20,000 students (Dunn, Dunn, & Price, 1981).

The PEPS identifies an adult's personal preference for each of twenty different elements and identifies how adults prefer to function, learn, concentrate, and perform their occupational or educational activities in the following areas:

1. immediate environment (sound, temperature, light, and design)
2. emotionality (motivation, responsibility, persistence, and the need for structure or flexibility)
3. sociological needs (self-oriented, peer-oriented, authority-oriented, or learn in several ways)
4. physical needs (perceptual preferences, time of day, intake and mobility)

Questions in each of these areas are presented and tend to reveal the way in which an individual prefers to work or concentrate (Price, 1996). Other factors that may influence learning are pace of work, mobility, amount of activity, reinforcement, variety, and persistence ("Learning Styles," 1986, pp. 1-3).

The interviews, both individual and the focus group session, had components that touched on learning styles. For example, in the individual interview, the following four questions involved learning styles.

1. Of all your schooling, elementary through college, who has been your favorite teacher/instructor? Why?
2. Of all your high school or college classes, which type of learning activities do you prefer? From which learning activities have you learned the most? Why?
3. Did the fact that the majority of your classmates are male affect your learning or understanding of the material? If so, how?
4. With mostly males in the class, did you feel intimidated? Did you feel hesitant to ask questions? Why or why not?

In the focus interview, learning styles were addressed with the following two questions.

1. What is your idea of a good instructor?
2. Describe your ideal classroom or learning environment.

The data gleaned from the Productivity Environmental Preference Survey was related and compared to the data gathered from the interviews.

Analysis of Data

First, the learning styles of the five subjects were compared to see if a common learning style prevailed among the participants of the study. The learning styles of the five subjects were then compared with the learning styles of the students from three classes of computer science, which contained a majority of males. Finally, the learning styles of the four computer science majors were compared with the learning styles of the lone non-major to see if there were any significant differences.

After that, the individual interviews were analyzed in a case study type of analysis. Where appropriate, the responses of each individual participant were compared and contrasted with the information gleaned from the PEPS. In each individual interview the responses to the first two Teaching/Learning questions were related to the results of the five learning styles variables on the PEPS – tactile, auditory, visual, kinesthetic, and learns in several ways. The responses to the four questions included in the Social Interaction section were related to the results of the five environmental variables on the PEPS – noise level, light, temperature, design, and mobility. The responses to the first two Content Complexity questions were compared with the results of three PEPS variables: works alone, wants authority figures present, and needs structure. Finally, the responses to the Persistence/Satisfaction questions from the individual interview were evaluated against the results of the motivation and persistence variables from the PEPS instrument. Then, the learning theory information was analyzed for common themes. This analysis gave the researcher a clear picture of the perceptions and viewpoint of each individual subject pertaining to the eight categories under investigation: Teaching/Leaning, Content Difficulty, Social Interaction, Content Complexity, Persistence/Satisfaction, and Motivation.

Additionally, the answers to the individual interview questions were compared in a cross-case analysis to discover the common threads present in the answers. All the answers to the first question were analyzed and compared to one another. Then, all the answers to the second question were analyzed and compared to one another, and so forth for all the questions in the individual interviews. At this point some common themes, which were color coded and grouped, appeared in the data. The common themes that emerged were also compared and related to the results that were reported by the participants on the Dunn, Dunn and Price instrument. This information was used to help analyze the focus group interview responses.

After all the individual responses were analyzed, coded and grouped, the focus group interview responses were analyzed. Central themes that emerged from the data were then coded, analyzed, and combined with the information gleaned from the Productivity Environmental Preference Survey. The responses of the focus group interview questions were amalgamated into areas of concern for educators and possible suggestions for encouraging more females to enter the computer science field.

CHAPTER IV

ANALYSES OF DATA

Productivity Environmental Preference Survey

The data were gathered in three stages. First, the Productivity Environmental Preference Survey was administered to three programming classes as well as to the females who consented to be interviewed. The answer sheets were collected and sent to Price Systems, Inc., for scoring. Price Systems, Inc., returned individual analyses for each student as well as a group analysis. The results are discussed below.

The Productivity Environmental Preference Survey (PEPS) is designed to identify an adult's individual productivity and learning style (Price, 1996). The areas identified by this instrument are seen in Table 2 below. The raw scores for each area are converted to standard scores between 20 and 80, with a mean of 50 and a standard deviation of 10. The standard score is based on a random sample of 1000 subjects from the national database who have taken the PEPS. Individuals having a standard score of 40 or less or 60 or more find that area important when they study or work. Individuals having scores that fall between 40 and 60 are varied with respect to how much that area is important to them (Price, 1996). For example, a score of 38 in the area of Noise Level would indicate that it is significantly important to that person to have quiet in their study or work environment. A score of 53 in the area of Time of Day would indicate that it is not significantly important to that person what time of day he or she works or studies.

Table 2

Profile of Programming Students

PEPS Area	Low	<40%	>60%	High
Noise Level	Prefers Quiet	0%	36%	Prefers Sound
Light	Prefers Dim	44%	12%	Prefers Bright
Temperature	Prefers Cool	20%	12%	Prefers Warm
Design	Prefers Informal	24%	8%	Prefers Formal
Motivation	Low	8%	8%	High
Persistent	Low	0%	4%	High
Responsible	Low	12%	4%	High
Structure	Does not like	4%	32%	Wants
Learning	Prefers Alone	4%	40%	Prefers with Peers
Authority Figures	Does not want Present	4%	16%	Wants Present
Learns in Several Ways	Does not learn in several ways	28%	0%	Prefers variety
Auditory	Does not prefer	0%	28%	Prefers
Visual	Does not prefer	8%	8%	Prefers
Tactile	Does not prefer	4%	40%	Prefers
Kinesthetic	Does not prefer	0%	16%	Prefers
Intake	Does not prefer	4%	60%	Prefers
Time of Day	Prefers evening	44%	0%	Prefers morning
Late Morning	Does not prefer	40%	4%	Prefers
Afternoon	Does not prefer	0%	64%	Prefers
Mobility	Does not prefer	4%	8%	Prefers

For the research sample of 25 students enrolled in Basic Programming, Programming I with C++, and Programming II with C++, the following PEPS profile emerged. The "typical" programming student from this sample prefers sound to quiet and dim light to bright light; prefers a cool, informal environment; tends to be persistent; desires to work with peers in a structured atmosphere with authority figures present; prefers auditory, tactile, and kinesthetic activities; prefers some sort of intake; and prefers afternoon and evening. These findings are presented in Table 2 and are characteristic of the five females who volunteered to be interviewed for this study.

The first analysis of the data was a one-way ANOVA to compare the means of the variables tested on the PEPS with regard to gender. Analysis of the standard scores showed no significant difference in the two groups, males and females, with regard to all variables in the PEPS except for temperature. The females preferred a warmer temperature than the males (sig. = 0.034, F = 5.097, df = 1,23). The mean for the females was 54.25, and for the males the mean was 45.59. The temperature mean for the group, which is displayed in Table 3, was 48.36.

The next analysis was another one-way ANOVA to compare the five female participants in the qualitative part of the study with the rest of the students. There was no significant difference in the two groups, the five female participants compared to the rest of the programming students, with regard to all variables in the PEPS with except for light and authority. The five female respondents preferred a brighter light than the rest of the programming students (sig. = 0.007, F = 8.621, df = 1,23). The mean of the five female respondents was 49.2, and the mean for the rest of the programming students was 22.95. The mean for the group on the light variable, as shown in Table 3, was 28.2. Unlike the rest of the programming students, the five female respondents preferred the presence of authority figures (sig. = 0.039, F = 4.808, df = 1,23). The mean of the five female respondents was 58.80, and the mean for the rest of the programming students

Table 3

Sample Descriptive Statistics

PEPS Area	Minimum	Maximum	Mean	Std. Deviation
Noise Level	41.00	64.00	55.08	6.4156
Light	00.00	59.00	28.20	20.5244
Temperature	30.00	71.00	48.36	9.6819
Design	28.00	62.00	45.84	9.0263
Motivation	26.00	60.00	48.76	7.2127
Persistent	42.00	61.00	51.44	6.1106
Responsible	22.00	62.00	48.88	9.0015
Structure	37.00	67.00	55.32	7.2210
Learning	40.00	80.00	56.52	10.5322
Authority Figures	37.00	73.00	53.00	7.1181
Learns in Several Ways	26.00	60.00	45.04	10.0145
Auditory	41.00	70.00	54.28	9.0394
Visual	40.00	61.00	50.48	6.6653
Tactile	38.00	69.00	55.12	8.8474
Kinesthetic	42.00	63.00	53.00	5.9372
Intake	37.00	67.00	58.08	8.3213
Time of Day	28.00	53.00	42.20	6.7454
Late Morning	29.00	60.00	44.76	7.3727
Afternoon	45.00	79.00	62.52	9.1975
Mobility	37.00	61.00	52.68	6.3948

was 51.55. The mean for the group on the authority variable was 53.00, as shown in Table 3. Table 3 presents the descriptive statistics for the sample chosen for this study and includes the mean, standard deviation, minimum standard score and the maximum

standard score for each variable. The means for this sample group follow the national norms except in two areas – light and time of day. This programming group strongly prefers dim light to bright light, indicated by a mean score of 28.20, and strongly prefers afternoon to other times of day, indicated by a mean score of 62.52. The large standard deviation for the variable of light indicates that there is wide variability among the responses. That finding is also reflected in the range of scores, from 0 to 59, for that particular variable.

The Productivity Environmental Preference Survey (PEPS) indicates that a characteristic is significant for an individual when the standard score is greater than or equal to 60 or less than or equal to 40. For the five female respondents, the means and ranges for all characteristics are listed in Table 4. The means of all characteristics fell between 40 and 60 with the exception of preferring afternoon, which was 63. Other than significantly preferring afternoons, there are no other significant indicators that appeared in the data. Scores that range between 55 and 60 or between 40 and 45 indicate a leaning or tendency toward that characteristic. The females somewhat preferred evening to morning (41) and late morning (45), sound to a quiet environment (56), working alone (59), structure (56), and the presence of an authority figure (59). There was also a slight leaning toward tactile learning (56). There are no other strong leanings that appeared in the PEPS data for the five female respondents.

Table 4

Respondents' Descriptive Statistics

Characteristics	N	Minimum	Maximum	Mean	Std. Deviation
Noise Level	5	52.00	62.00	55.6000	4.9800
Light	5	40.00	58.00	49.2000	7.5631
Temperature	5	41.00	71.00	54.8000	13.4239
Design	5	37.00	62.00	45.8000	9.9850
Motivation	5	36.00	52.00	45.6000	6.6933
Persistent	5	44.00	56.00	52.4000	4.9295
Responsible	5	50.00	56.00	52.4000	2.1909
Structure	5	50.00	64.00	56.4000	7.1274
Learning	5	43.00	67.00	58.8000	9.3113
Authority Figures	5	54.00	63.00	58.8000	3.4205
Learns in Several Ways	5	43.00	51.00	47.0000	4.0000
Auditory	5	46.00	58.00	52.6000	5.3666
Visual	5	40.00	53.00	45.8000	5.1186
Tactile	5	38.00	69.00	56.2000	11.3004
Kinesthetic	5	42.00	63.00	51.4000	7.5366
Intake	5	37.00	62.00	52.8000	11.2561
Time of Day	5	28.00	52.00	41.2000	10.8950
Late Morning	5	40.00	50.00	45.0000	5.0000
Afternoon	5	49.00	79.00	63.0000	11.9791
Mobility	5	43.00	61.00	53.4000	6.8411

Illustration 1 shows the summary data for the PEPS for the five female respondents.

Means Graph for Female Respondents

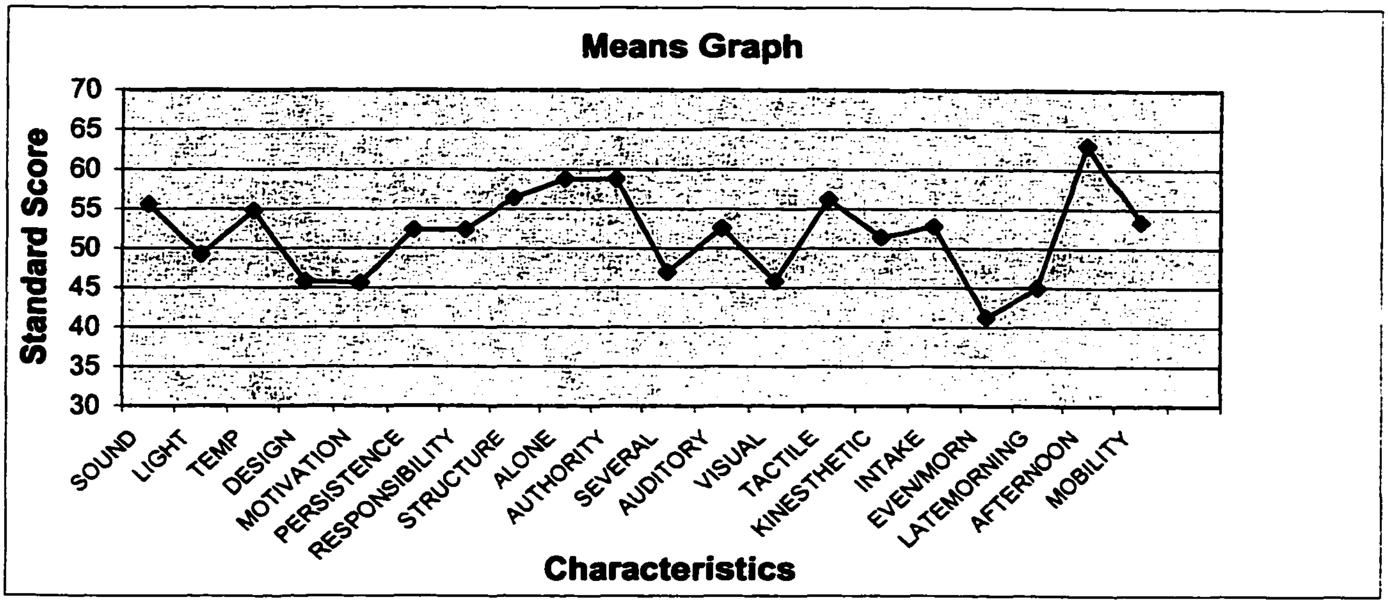

Figure 1. Means Graph for Female Respondents

<u>Individual Case Studies</u>

The next stage in the data gathering was the individual interviews. All the data presented in this section came strictly from the individual interviews. Of the five females chosen for this study, one had successfully completed Basic Programming (Diane), one had successfully completed Programming I (Abby), one had successfully completed Programming II (Betty), one had unsuccessfully attempted Programming I twice (Carol), and one had changed her major (Evelyn). Each female was interviewed in a computer lab using a Web cam with a built-in microphone. Each video interview was permanently recorded on a computer hard drive and subsequently copied to a blank CD-ROM. The researcher and two assistants individually transcribed each interview.

Abby

The first participant to be interviewed was Abby (pseudonym), who is a twenty-nine year old Caucasian and a single mother with three children. Abby has successfully completed College Algebra and Trigonometry, as well as Basic Programming and Programming I with C++. Abby was articulate and relaxed during the interview.

Abby's favorite teacher was her high school Power Tech teacher because that class allowed hands-on types of activities in which the students took things apart and put them together again. Abby stated that she learns the most from the hands-on type of activities and that she prefers this type of learning activity. She stated, "It's hard for me to read something and always ... understand it. And at the same time, I can't listen to someone stand up there and explain step by step and me still remember everything ... so it's easier for them to explain while I am doing it, 'cause then I remember it easier." Her PEPS sub-scores in these areas reflect a tactile (57%) and auditory (58%) type learner. Other sub-scores indicated that she tends not to learn in several ways (43%) and is a moderate visual (42%) and kinesthetic (50%) learner.

Abby indicated that the males in the class did not make her feel intimidated or hesitant to ask questions. She stated that she felt comfortable with males because she had been reared with four brothers and that she did not take seriously the gender-biased joking that occurred periodically. Abby did not have much computer use prior to beginning the computer science program at the community college, but she rated the first two programming courses, Basic Programming and Programming I with C++, as relatively easy. She related learning computer science to learning mathematics by saying, "I am a very logical person ... it seems a lot like Algebra ... and I'm really good at math, and it makes it real easy for me. ... Some people pick up French; I pick up computer language."

Abby felt comfortable going to the instructor for help rather than to the other students in the class, and she worked in the college computer lab on her programming assignments, rather than at home. These attitudes are supported by her PEPS sub-scores that she prefers a moderate noise level (52%), warm temperature (71%), moderately bright light (56%), and a moderately informal environment (48%) where she can walk around (61%). She stated that she did not have the C++ compiler at home, and since the programs did not "take the extra time on a normal basis," she just preferred working on them in the lab. Abby also stated that she was comfortable working either in an environment with other people or working alone, although her PEPS sub-score indicated that she prefers working with her peers (67%). She had a study partner in the Basic programming class, but not in the Programming I with C++ course. The course content that gave her the most difficulty was the argument list. Abby stated that she encountered difficulty with arrays, but in analyzing her responses, the underlying difficulty was the argument list. When asked about this difficulty she stated, "Because you have a lot of, uh, different ways of putting the arguments in and ... which one had to have which ... some of them had to have the name and the dimension, and some of them just had to have the name. And sometimes you had to have the name, the dimension and how many you wanted in it, or how many you wanted on a line. ... That's harder for me to remember which ones go together." Her primary method of overcoming difficulties in learning the programming concepts was to rely on the handouts that were given out by the instructor rather than to confer with other students. Her PEPS sub-scores in two categories also indicated that she wants structure (68%) and wants authority figures present (60%).

When asked about the different topics in the course content, Abby related them to something in her everyday life or mathematics. For example, she related variables to "something that changes on a regular basis. ... if you have Algebra, or Beginning Algebra, you know what a variable is so that was real easy." For if-then statements, she said, "See, I have children, so you have to say a lot of if-thens. ... There's a lot of if-thens at home." The condition for stopping looping structures gave her difficulty, but she finally related it to if-then statements in a mathematical context. "I didn't have a lot of information to relate looping structures to, so it was more like a math thing, where I put it in an if-then sort of context." Abby had no problem with sub-programs, and she related them to something she had done previously in programming, but she again reiterated her difficulty with argument lists. She stated that she related the passing of arguments to "electrical lines. ... If you use the local [variable], then the electrical circuit is broken, and if you used the global [variable], it wasn't." She stated that she did not relate argument lists or arrays to anything with which she was familiar, and "... that's probably why I don't like arrays much."

In both programming courses, Abby stated that she "breezed through the whole semester [in both programming courses] until I got to the last program, and then I was just sort of lost." She stated that in this situation, her self-confidence was decreased. "When you get lost on a program or just go blank, you start to wonder, am I in the right field? ... So you get through it, you turn the program in, you get the grade, and then you go, 'Oh, I did better than I thought I did.'" Getting through the last program boosted her self-confidence and she said that the difficulty really did not hurt her determination to succeed; rather she was even more determined to finish the course. Abby stated that her motivation for choosing computer science as her college major is money. "That might sound a little self-serving, but I'm a parent with three children... and I had to find a way ... to support them on a regular basis." Her score on the motivation portion of the PEPS instrument reflected medium low motivation (44%) and moderate persistence (53%). Her reasons for staying with computer science are "It's fun! And I enjoy creating programs on the computer." She also mentioned versatility and the many career choices available in computer science.

Abby did not offer any suggestions for encouraging females to choose computer science as a college major because "you either pick it up very easily or it's very, very hard, and ... I don't really think gender makes a difference as far as how you pick it up. ... I think on a general basis men are more mathematical and logical than women are, or think they are any way." She also stated that she persisted through the courses when other females dropped out because, "... it doesn't seem that hard to me, and it's a goal that I've set for myself and I intend to finish." Her recommendation for encouraging females to enter computer science was to display a sign that says, "If you are female and good at math, come to Computer Programming."

Betty

The second participant to be interviewed was Betty (pseudonym) who is Oriental, twenty-three years of age, and married with one child. Betty has successfully completed College Algebra, Trigonometry, Calculus I and Calculus II, as well as Basic Programming, Fortran 90 Programming, Programming I and II with C++. Betty's command of the English language is very good, and she was articulate and relaxed during the interview.

Betty stated that her favorite teacher was one who was gifted in telling stories, so that there was no effort to learn. She just listened to his stories, and she remembered. Betty reported that she has no preferred method of learning but that she has learned the most from hands-on types of activities. Her PEPS sub-scores reflect that Betty is able to learn in different ways (51%) but does not necessarily prefer variety and that she is a moderate auditory learner (52%), a low visual learner (40%), and a high tactile (60%) and kinesthetic (63%) learner.

Betty reported that her male classmates do not intimidate her, even though she is the only female in the class. However, she did say that she was hesitant to ask questions because "I guess I didn't want to look stupid ... but most of the time it's because they answer so much faster than me." She also made reference to a male attitude of superiority making her hesitant to ask questions in class. Betty said, "I'm mostly hesitant because ... what if they ... kind of look at me like [makes a sneer], like that."

Betty had no computer use before she began her college programming classes, and she responded that the Basic programming course was very simple for her. She said, "It's just all logic. You ... have to think a little bit, but not that much. You know, normal arithmetic and basic skills. You don't have to work very hard." The next course in the sequence, Programming I with C++, Betty rated moderately difficult. She stated, "It needs a lot more math. And ... the language was a little hard for me. I think one was because I didn't have as much contact with it. ... But it's mostly the math that kind of stumbled me." The Programming II with C++ she rated very difficult due to pointers.

Betty sighed, "Pointers is what killed me. It took me a month, or a few months, to finally … understand it. … It took me a while, but I think I got it."

Betty reported that when she was having difficulty, she went to the instructor for help, although she did admit to asking her classmates for assistance once in a while. "Mostly suggestions, you know, and comparing notes and stuff. Comparing programs, and seeing what they do and what I've done wrong." Betty said that when she encounters content difficulties, she prefers asking the instructor or trying to figure it out on her own rather than asking her peers. This attitude is supported by her PEPS sub-scores that show Betty prefers working alone (43%), that she wants authority figures present (60%), and that she strongly desires structure (64%).

When asked about her social interactions, Betty replied, "I have to break it down because before I started really getting into my studies, I was, you know, like a social butterfly. I'd make friends with a lot of people, but most of my friends were males. I guess that is why I don't feel uncomfortable around males. But since I started school here, I have not talked to very many people, and I guess I'm a loner now. The only people I talk to is my son and my husband." Betty indicated that she mostly works on her programs and assignments at home rather than in the computer lab, and that she does not desire a study partner. Her ideal working environment is "Quiet, um, no distractions … that's really the two most important things. … When I'm really concentrating and something distracts me, I get really irritated." Her PEPS sub-scores indicate that she prefers surroundings with a moderate noise level (52%), moderately bright light (58%), a warm temperature (67%), and a fairly formal structure (62%) where she can be moderately mobile (55%).

The topic that gave Betty the most difficulty in Programming II with C++ was pointers and string arrays. To overcome the difficulties, Betty "drew, you know, pictures and stuff, to try to let me understand what actually happened. That really, really helped." This is not reflected in her visual sub-score (40%) on the PEPS. When asked about how she understood certain topics in computer science, Betty related most of them to some topic in mathematics. For example, when asked how she understood the concept of a variable right away, she replied, "Um, by comparing it to algebra. … It just clicked. You know, there's no difficulty there." She also had no problem with if-then statements, looping structures or sub-programs. Concerning if-then statements she said, "… if something happens, then something else happens. You made it really clear. I had no problem with that." With looping structures she had a similar comment, "… you know, do while a condition is present … and then stops when it's not. … It's just common sense to me." She compared sub-programs to food. "You know, food is … a main idea, and you have like main courses, entrees and different things, and then from the different foods, the entrees, you have … ingredients … like a hierarchy." She also had no difficulty with argument lists. Her standard reply was, "I just feel like it is all common sense. Nobody should have any problems with it," or "It all just makes sense to me." The only topic of any significant difficulty to her was pointers. Betty thought, "pointers should be introduced a lot earlier instead of being just pushed onto us like that. It shocked a lot of us. And almost half the class dropped after the first pointers assignment. It was just a shock…"

Betty's self confidence was bolstered when the course content became more difficult. "It felt good that I could do something that some guys … dropped out of." Her main reason for persisting in the face of difficult course work was her son Zachary. She stated, "I want to graduate. You know, my Dad has been very forceful in this. ... I feel like even if it gets really hard, I just have to stick in there and try to fight my way through it. So when I graduate with my bachelor's, then Zach will have a good life." Her main strategies for understanding difficult and complex ideas were "mostly drawing my way through it, and sometimes just laying in bed thinking about it ... Yeah, just thinking and drawing by myself helps." Betty's reason for choosing computer science as her major was "because I like to type. ... Oh, and because of my cousin. She graduated with MIS and I misinterpreted. I thought it was computer science, so I took computer science and she told me it was easy. But I took computer science and I liked programming ... so I decided to stick with it."

Betty's reasons for staying with computer science are "I like it. I don't like to admit it, but it is fun and it is interesting." Her future goals are to "make lots of money and maybe do something that nobody else has done before – be a Bill Gates person." What Betty likes most about computer science is "being in control. You decide what happens in your programs... you are in control of everything that you do and when you want to do it and how fast you want to progress with it." She repeated that she likes "being by myself, you know, concentrating." Her PEPS sub-scores show moderate motivation (44%) and moderate persistence (53%). The thing Betty likes least about computer science is the mathematics, even though she has just successfully completed Calculus II. "It just kind of hurts me that I have to take all this math. But sometimes it's interesting ... but other times it's just, you know, a lot of work. Sometimes I like it, but other times ... I don't really want to be doing this, and I hate it."

Betty suggested that all-female classes be offered because the females in the classes will "be more open about voicing their difficulties and ... you know, females bond a lot easier and quicker, and when they have difficulties like that, they will ask the person next to them or even they'll do a huddle, like, you know, a group thing. And that helps solve problems a lot faster." She also mentioned forming support or study groups but noted "I'm different because I'm with guys all the time." She said that does not bother her because "I guess I was a tomboy, and I've always had male friends ... so I'm used to guys. I don't really care what they say around me ... guys don't bother me." When asked why she has persisted in the program, Betty replied, " ... I'm really stubborn, and I would be somebody who would fight, you know ... most females are really submissive and they don't like to project ... I'm not afraid of standing up for what I believe ... in computer science where it is all males, you have to be strong, you know, just be yourself and be strong." Betty observed, "Females in computer science don't talk to each other. That's why they don't stick together." Her suggestion to help alleviate this problem was group study, even though she does not like it. Betty noted that Orientals have the idea that you have to stick it out because of family expectations, but she said, "I'm not really like that. I think I'm in it because I feel that I owe Zach. I want him to have more than I did... a couple of times I felt like quitting ... but I sit down by myself and I think about Zach. That's really what helped me stick through it."

Carol

Carol, the third participant in the study, is twenty-one years of age, black, and single with no children. Carol has successfully completed College Algebra, Basic Programming, and Cobol Programming. She has attempted Programming I with C++

twice unsuccessfully. Carol was very reticent during the interview and often gave one-word responses. It was difficult to elicit much information from her, and she seemed to need prompting to help her articulate her answers.

Carol stated that she really did not have a favorite teacher, but she liked her English teachers from high school and college because they were very nice and helped her when she needed help with her papers. She stated that primarily liked hands-on types of activities because "it seems like you learn more than just listening or reading. Because if you make mistakes, you can learn how to fix them as you go." She had no experience with hands-on classes except her high school and college computer classes. Her strong preference for hands-on activities is not supported by her PEPS sub-scores in tactile (38%) or kinesthetic (42%). Carol does show moderate auditory (49%) and visual (48%) learning preferences, and another PEPS sub-score indicated that she moderately learns in several ways (47%).

Carol said that having so many male classmates and so few female classmates in her computer science classes sometimes seemed as if it affected her learning or understanding of the material. She said, "… it just seemed like most of the guys did group together, like the women were by themselves." Carol also commented, "It seemed like it was harder to talk to the guys because they seem like everything was so … complicated." She reported that she did not feel intimidated, uncomfortable, or hesitant to ask questions, because "I felt like the guys probably had questions, and what I had to ask … they probably need help on too. It was like they was just in there, but it didn't make a difference."

Carol declared that she had three years of computer experience in high school. This was composed of introductory type material, with simple programming interspersed. She rated the Basic programming class as moderately difficult for her because she said it was "hard, but as I got into the, um, upper ones, it seemed really easy." The Programming I with C++ class she rated very difficult because "I had a hard time with functions, and I didn't really understand what went in it." Carol said that she goes to the instructor for help, but sometimes does ask her female classmates for assistance. She reported that she asked her male classmates for help only once or twice because they responded, "Like I should have known it" and their responses made her feel "Kind of stupid, I guess." Her PEPS sub-scores show that Carol moderately wants structure (50%) and desires authority figures to be present (54%).

Socially, Carol described herself as in between because "I don't really talk to a lot of people. And sometime I just talk to certain people." She reported that she does not have a group of friends at school or at home, but she primarily works on her programs in the lab at school. Carol stated, "... because at school I know I am going to do it; at home I just put it off." She said she prefers not working with a partner because "it just seems like more complicated to me when somebody else is working with me. ... It's like I don't want to be with nobody. ... I don't want to work with nobody." Carol reported that it did not matter to her about the working environment, and the PEPS sub-scores bear that out. She scored in the moderate range for noise level (52%), light (46%), temperature (45%), and preferred a rather informal atmosphere (43%). She was fairly ambivalent about working with other students and emphasized that she likes working alone: "It doesn't matter. Sometimes I just want to be ... off in a corner. Sometimes it's better when people aren't around me." Her PEPS sub-score indicated that she prefers working with her peers (60%) and prefers mobility (57%).

Carol prefers to try to figure out difficulties on her own because "it seems like I learn more if I figure it out by myself." The topic that gave her the most difficulty was functions, but an analysis of her responses indicated that the real difficulty was the passing of arguments and the argument list. Carol said, "I just didn't get it … you are supposed to pass certain things in to get certain things out. That was hard for me." When asked how she tried to overcome this difficulty, she responded, "Just tried to get where it would work … if I passed something in that did not work I'd try something else."

Carol did not have any trouble with the beginning topics in computer programming, such as variables or if-then statements. She related the idea of a variable to Algebra, and if-then statements to every-day life "because it's like saying, ok, if this doesn't work then you can do this, so that's the way I got that." Looping structures gave her a bit of trouble "because it was hard to figure out … how many times the loop should occur. So that was a problem for me." Carol said that sub-programs made sense to her because she could "break the program down into different parts, like the introduction and then the calculation part," and she also related it to "writing a research paper or something and you break it down like different parts." She reported that arrays made sense to her and that she had no difficulty with the subscripts, but the argument list, the notation in C++, and passing the arguments confused her. She said, "I got a lot of errors and I couldn't work it."

This affected her self-confidence, and she said she wanted to get out of computer science because "it was just there were certain things that I couldn't figure out …

functions, and ... certain loopings and stuff. That was hard for me to get it. So I just felt like I wanted to get out. Sometimes I just want to forget it. But, then I figure that I have done all this, so I might as well just go on through the rest." When she accomplishes something difficult in computer science, "it makes me feel like I can do the rest of the program." When she encountered difficulty in programming, she just worked harder at it by reading the textbook, going back over her notes, and sometimes looking at the handouts. However, she said that sometimes the handouts confused her more.

Carol said that she chose computer science originally because she was very interested in computers and just wanted to see how certain things worked. When asked if she would stay with computer science, she replied, "No," but when asked what her new major is, she stated, "It's still computer science, but I am not sure what or if I am going to continue with it. I don't have any [future goals] right now. I guess just to get through with computer science." Carol scored low in motivation (36%) but had some thoughts on why females dropped out of computer science. She said, "Most of the girls I know got out of it because it was too complicated to them, and they felt that they had a hard time with Basic, so they were going to have a hard time with everything else. So they just got out." When asked why she has stayed in, she replied, "I don't know why I stayed in really." In response to the question of liking computer science, Carol sighed, "No. I just have a hard time ... when I do something I just have a hard time not finishing it. ... I just didn't like the idea of giving up." Her PEPS sub-score in persistence (56%) certainly supports this. Carol had few positive comments for other females about a computer science career. "It will be more money. And they'll learn to like it, and I don't know. It's hard for me to say that. It's ok, but it just gets hard ... plus all the classes you have to take with programming is hard." When asked if she had any encouraging words for other females interested in computer science, her answer was, "No."

Diane

The fourth participant, Diane (pseudonym), is a twenty-two year old single Caucasian with no children. She has successfully completed College Algebra, Trigonometry, and Calculus I as well as Basic programming. She completed Fortran programming at a university prior to attending the community college. Diane was very articulate and comfortable during the interview.

Diane's favorite instructor was her gifted instructor in high school because "She treated us all with respect. ... She allowed us free time... She gave us mind benders, things that tested out computational and analytical skills. ... She always would really get me into heated discussions with her in class..." Diane said, "I like hands-on, you know, tactical skills or anything taught through lecture or experience. ... I like my computer and my science classes and in my math, you know, you learn it and you do it. That's it, I just like to learn it and then do it." Her PEPS sub-scores reflect a strong tactile (57%) and moderate kinesthetic (52%) learner. The other sub-scores reveal that Diane does not prefer auditory (46%) or visual (46%) learning activities.

Diane did not feel uncomfortable or intimidated by the large number of male students in the classroom and reported that she was never hesitant to ask questions. Her previous computer experience consisted of a required typing class, and although her family owned a computer while she was growing up, she said she never touched it. She rated the Basic programming class moderately difficult and said that it would have been more difficult without the Fortran background from a university.

When asked about where she goes for help, Diane said that first she talks it out with her cat, then hits the books, and finally goes directly to the instructor. This response is reflected in her PEPS sub-scores, which show she wants moderate structure (50%) and an authority figure present (63%). She displayed good insight when she noted, "But I do in the future plan to collaborate a lot more with my colleagues, my peers in the classes and in the future at work because I feel that is a very integral part of this…." She also stated that she has not found a "study buddy" and that she's been "quite the loner for years." She works on her assignments "at home, for the first fifty percent of the assignment, then I come into the lab for the next thirty percent … and then once it's done, it's back home again for a double check." She repeated that she prefers to work alone, but her PEPS sub-scores show that she much prefers working with her peers (60%), prefers sound to quiet (60%), prefers an informal environment (37%) with limited mobility (43%), a moderate amount of light (46%) and a moderate temperature (50%).

Diane said that she never hesitates to ask questions in class if something is not clear to her. She said, "I think that it's important to try and get it first, and then for everyone's benefit there are always questions, and there may not always be correct answers, but it's important to pose possibilities. When asked about the most difficult topic she encountered in computer science, she replied, "I think that it is a science-based subject which involves math in such a way that it's hierarchical, I think. You start with the basics and you build and build. So I think the last thing we learned, arrays, comes to mind. … I'm sure that the next thing we get into will be the most difficult thing I've ever encountered." She clearly saw the relationship between mathematics and computer science. She related variables to Algebra and topics in every day life such as figuring out her budget and figuring her daily nutritional intake. If-then statements were also no problem for her. "If, you know, you tell your Mom you're gonna be home by six and you're not, then you'd better expect to be asked for an explanation." Diane related looping structures and sub-programs to everyday tasks and had no difficulty at all understanding the concepts. She was thrilled to discover the syntax and quickly related it to experiences in her own life. Diane said, "The only thing I know is what I have lived. I only know about the path that I have walked down. ...You have to relate those things to other things in your life... It's become absolutely necessary that I take every aspect of my life and integrate them in a way that I can handle them."

Diane did not think the course content became more difficult as the course progressed, but rather she "just found it to be the next logical step...." She said of herself that she is very determined and "spent long hours, long nights...and got the satisfaction of knowing I completed something that I had put time and effort into, that I had learned something new...." She also said she is not the type to give up and that she did "what was necessary and maybe a little bit more. ...What was it that I needed to know in order to complete the objective, both in application and examination." She also mentioned outside research, buying and borrowing books and reading them, going on-line, and finally sorting through all that information. Her PEPS motivation sub-score was (52%), showing that her motivation is moderately high, but her persistence sub-score was moderately low (44%).

Diane related that when she was in the seventh grade, an engineer came and told her class that there were hardly any females in engineering. In her words, "That was really, well, shoot, that floored me!" She said that statement ignited her desire to become a chemical engineer, but she knew it was "very, very, very important to know about computers." She is interested in computers because, to her, "computers are the model of the human brain ... in an electrical sense, it has its own formation, yet every brand is different, [it has] the same, certain inherent qualities..." Her future goals are "to provide for myself, ... have a family, ... a comfortable salary." Diane showed great insight into computer science with this observation. "That's why I like computer programming. I think it's a way to help other people. I don't see why I necessarily have to sit in an office and interact with you all the time when I can do two things at once. I could be there at my computer and doing something on a grand scale, something that can be distributed. And then on a personal level, you know, kick back and ask someone how their day is going."

Diane said that she has not really found anything that she dislikes about computer science and that she really would not change anything. In conversing about the problem of few females in computer science, she offered no immediate suggestions. "For a long-term solution," Diane commented, "we have to start at home."

Evelyn

The fifth participant in the study, Evelyn (pseudonym), is a forty-year-old Caucasian, who is married and has three children. She switched her major from computer science to business. Evelyn's favorite teacher was her Latin teacher in high school because "she was very encouraging, very demanding." Evelyn stated that she prefers hands-on types of learning activities. Evelyn described hands-on activities as including group discussions in which the students express different viewpoints. She said she is not fond of team projects because "invariably you have one person that doesn't want to do any work, and they want to take advantage of the grade," but she reported that team projects have helped her learn how to get along with different types of people and "get the best out of everybody." Eveyln stated that she learned the most from the written drills on prefixes and suffixes in her Latin class. This assertion is supported by her PEPS inventory, which shows that Evelyn strongly prefers working with peers (64%), that she is primarily a tactile learner (69%) and an auditory learner (58%), with some tendencies toward visual learning (53%) and kinesthetic learning (50%). Her scores also indicate that she tends not to learn in several ways (43%).

Evelyn said that she felt intimidated in a way with the large number of male classmates because "you couldn't phrase a question the way that they would answer you the way you wanted it. You would have to go at it a couple of times before they finally understood what you were confused about." She also stated that she felt hesitant to ask questions of a male instructor because "he would make you feel like you were incredibly slow, or this is something you should have caught on to." She also reported that she went primarily to the instructor for help when she ran into difficulty. This tendency is supported by her PEPS scores, which show she prefers structure (54%) and prefers the presence of an authority figure (57%).

Evelyn had no previous computer experience before she entered college. She has taken several computer language courses: Basic Programming, Fortran, Pascal, Cobol and C++. She rated the Basic programming course of medium difficulty because it was a new concept for her. She rated the Fortran course very difficult, because "that's the one where I had the instructor that read straight from the book and we basically taught ourselves." Evelyn reported, "I can be a loner, I can get things done on my own, but I do prefer to be with other people..." She stated that she prefers to work at home or in the library on computer, but that she had study partners. "Yeah, survival in numbers was better. If you had a partner, you did a lot better." This observation also is confirmed by her PEPS scores, which show that Eveyln prefers sound to quiet surroundings (62%), with a dim light (40%), cool temperatures (41%), and a rather informal design (39%) with mobility (51%). She also prefers to ask the instructor questions rather than ask questions in the class because "you're either treated like you're a know-it-all, or you're treated like you're a total idiot by the other class members... It's a little intimidating 'cause they look at you like [makes a face of disdain], 'Can't you shut up?' So, at least that is what I've found in the [classes] I've taken."

Evelyn said that the topic in programming that gave her the most difficulty was passing arguments. She said that the variable idea was not very difficult for her because "I'd had some biology classes and all, and associating things with other things was not hard for me. That part females seem to get. The males had more trouble with that, I think, than the females did, for some reason." When asked to speculate on this idea, Evelyn responded, "Well, women are always, when you're talking about a car or car parts, the thingamabob. [Women] know what that is when you're talking about it. You already have another name for it. That's a variable name for it. But, see, guys are very ... literal in some ways. A cup is a cup. A plate is a plate. So that one, I didn't have any problems with." Evelyn connected if-then statements with consequences to actions from her everyday life. She also stated that she had no trouble learning looping structures or subprograms. She related arrays to post office boxes, and several times she stated that she had to visualize or "see it in my head" in order to understand a particular concept. The argument passing gave her difficulty in the syntax of the language (C++), and she stated that she is more comfortable in a more structured language such as Fortran.

Evelyn agreed that the higher programming courses became more difficult for her, but she stated that the stumbling block that made her change her major was the higher math requirements. She said, "One thing I noticed got a lot of females out of [computer science] though was before you could even take a certain class, they told you that you had to have ... this higher math and that higher math. ... I think the emphasis on the higher math was too much, maybe." She expressed some definite ideas on higher math. "The criteria is being able to think really, in a logical manner, not necessarily to write a formula ... If you can program the whole program and all you need is a two line piece of code for the math, you can always go find someone that's had the higher math or is a math-oriented person, and they'll talk them through that ... You can go get books, you can go get help from other people. I think that is a big stumbling block for a lot of people."

Evelyn discussed how her confidence level was positively affected by the difficulty of the concepts when she grasped the concept before others in the class did, and also how it was affected negatively when she could not grasp the concept and the others in the class did. She commented that the general male attitude in her Pascal class simply promoted some friendly rivalry among the class members.

Evelyn finished all the programming classes except the C++. She said that during the other classes, "Sometimes I'd feel like I'd want to give up, but then I would get mad and be determined but, at least, try ... so, it usually made me more determined to go ahead and finish it." She also stated that she intends to go back and finish the C++ class when she completes her degree. Her PEPS scores indicate moderately high motivation (52%) and persistence (56%), corroborating her statements on determination and persistence.

She stated that when the content became more difficult, she included visuals such as a diagram or flowchart to help her understand her notes. She was initially attracted to the computer programming courses because "they were always changing and interesting, and I just liked it better." Her reason for changing was "because they still have that higher math requirement. That's my main one, more than anything ..." She liked computer programming because "it can be very imaginative, it can be exciting, it can be different. It's not the usual." What did Evelyn like least? "It's got to be the math." To encourage more females to enter computer science, she would take out the higher math and substitute a test for logical reasoning. She also reported that if there had not been the higher math requirements, she would have stayed in computer science.

Cross-Case Analysis

All five respondents reported that they preferred hands-on types of learning activities and that they all went to the instructor for help when they had difficulty with the course material. All indicated that they rarely asked classmates for help, although Abby and Evelyn reported having study partners. The other three indicated that they preferred to work alone.

Four of the five did not feel intimidated in any way by having a large number of male students in the classes. The one who did feel intimidated was Evelyn, the one who changed her major from computer science to business. Three of the five were not hesitant to ask questions in the predominantly male environment. Both Betty and Evelyn felt hesitant to ask questions because they did not want to appear stupid or slow, but none of them felt uncomfortable with most of the classmates being male.

All but Abby rated the Basic programming course as medium difficulty. Abby rated it slightly easier in difficulty. Four of the five had difficulty with argument passing notation in C++. Both Abby and Carol (Basic programming and Programming I) had difficulty with passing arrays in C++. Evelyn, who did not finish Programming I, had difficulty with passing arguments to and from functions in general. Betty (Basic programming, Programming I and II) reported great difficulty with the pointer notation in the function argument lists of C++. Diane has not experienced the C++ language yet and reported no difficulties with argument passing in the Basic programming language.

Three of the five had no computer experience before college. Abby had no previous classes in high school, but she reported six months of computer experience at home and Carol had three years of computer classes in high school. All the respondents reported high levels of determination and a reluctance to quit or give up. Both Abby and Betty are mothers, and they reported their primary motivation as being able to provide a quality life for their children. Diane also reported a desire to have a better life, and Carol reported that she does not particularly like programming but that she has a hard time not finishing something she began.

Betty, Carol, and Evelyn all agreed that the one thing they disliked the most about computer programming was the math requirements. They saw no relevance or purpose for requiring the higher math courses. All of the respondents but Carol liked the variety and creativity of computer programming. Abby said, "The many different things you can do with it." Betty replied, "Being in control. You decide what happens in your programs." Diane said, "I think it's a way to help other people." Evelyn stated, "[You can] use your imagination and put a lot of yourself into it." Carol just liked getting the programs completed.

To summarize, then, all five respondents stated that they preferred hands-on type learning activities and that they learned the most from hands-on type activities. Abby, Betty, and Evelyn had no computer courses before college. Diane had one typing course and Carol had three years of computer classes in high school. All five said that they went primarily to the instructor for help in the computer programming courses and asked their classmates for assistance only on occasion. When asked about study partners, Evelyn and Abby both responded that they could work alone or with a study partner or partners. The other three stated firmly that they preferred to work alone. Only Abby actually had experienced a study partner in the first programming course.

Both Abby and Diane expressed that they never felt intimidated or hesitant to ask questions with the large number of male classmates in the programming classes. Betty responded that she did not feel intimidated but did feel hesitant to ask questions because she did not want to appear slow in front of the males in the class. Carol said that sometimes it seemed that having mostly male classmates affected her learning and understanding of course content because the males tended to group together and exclude the females, and it was harder to talk to them when she had a problem. She also stated that she did not feel intimidated or hesitant to ask questions. Evelyn shared that she felt both intimidated and hesitant to ask questions in her programming classes because one male instructor made her feel "incredibly slow" if she did not catch on immediately and the other male instructor was so bright that when he answered her question she felt more in the dark than before the question was asked.

When asked about how the difficult concepts affected their self-confidence, all the female respondents expressed that when they ran into difficulties, feelings of doubt about their ability to do computer science surfaced. Each one declared that at one time or another the thought of quitting arose. But each one stated her intention of finishing the program, expressing determination and a distinct aversion to giving up. Abby and Betty both have children, and this responsibility emerged as a very strong motivating factor. Diane also possesses a very strong internal motivation. Carol expressed that she is not sure she will continue with computer science, that she does not know why she has stayed in it, and that she does not like computer programming. Evelyn stated that she intends to return and finish computer programming as soon as she has completed her other educational goals.

Focus Group Interview

The third, and last, stage of data gathering was the focus interview. This was conducted after all the individual interviews were completed. Evelyn was not included in the focus group because her major was not computer science. The other four females were interviewed in the Computer Lab using a Web cam and a microphone. The Web cam was too far away to pick up the voices, so a microphone was used. The microphone was passed from person to person as each one gave her responses. This interview was also permanently recorded on a computer hard drive and subsequently copied to a blank CD-ROM. The researcher transcribed this session. All data in this section were gleaned strictly from the focus interview, although many responses from the individual interviews were repeated here.

Included in the focus group interview were Abby, Betty, Carol, and Diane. Before enrolling in the community college where they are now attending, Betty, Carol, and Diane reported experience with a spreadsheet program and a word processing program. Abby reported no previous experience except six months on a home computer many years ago. Diane also reported taking a typing class and some experience with a presentations package. When asked about difficulties experienced in the programming classes, Abby reported difficulty with arrays; Carol, with looping and functions; Diane with finding time; and Betty, with pointers and string arrays.

Their composite view of a good instructor is one who "can make the class fun" and who can not only explain the material so that the class can understand it, but who can give the perspective as to why the class should know the material; one who "has a lot of patience" and is willing to spend quality time outside of class; and one who is friendly and female, because "they tend to understand female students and they ask female students questions." When asked to describe their ideal learning environment, the consensus was that it would include friendly people, both students and the instructor, a not-too-quiet place where sarcasm and intimidation were absent and where students could talk to each other.

When asked about the predominantly male environment in the classrooms, Betty reported that she does not talk as much and participates less when there are large numbers of males in the class. Carol replied that the mostly male presence did not make any difference to her. Abby and Diane said that they interacted more with the class when its composition was primarily male. The general consensus about asking questions in class was that it depended on the instructor. The participants all replied that, if the instructor was friendly and encouraged questions, they asked questions in class when they did not understand. Betty and Abby stated that they needed help in approximately two out of three new concepts introduced in programming. Carol said she needed help "every so often, because I would forget the concept while I was doing the program and so I would have to be reminded." Diane said on those concepts that were not covered in available literature, she tended to ask for help.

None of the four reported using a study partner during the past semester in the computer programming class in which they were enrolled. All four preferred using the lab setting as opposed to working at home, but none wanted to work in a group. For Betty and Carol, the lab is quieter than home because of their children, and instructors are around to answer questions.

When asked if they thought the math courses had helped them understand programming, Betty said, "I finished Calculus II. I don't feel like it did very much to help, although in the earlier courses the equations are ... what I used to help me to understand." Abby stated, "I just finished Trig and ... the math classes at the beginning, the Algebra, College Algebra, it does help. A lot. You get past that, I don't see where it's helping at all." Carol had just finished College Algebra, and did not see that it was any help to her at all. Diane had just finished Calculus II, and said, "I actually think they've all helped, mostly in the Algebra... But I've found that as the scenery grows in math, as I learn more, it just tends to broaden my horizons, my perspectives, and it opens my mind to a new way of learning which permeates to all other courses, and in the case of computers, that's not an exception."

When asked about how they integrated the computer programming concepts into their internal schemas, the general response was that they "added on to what was already in there," or "it was just memory work and connecting what you already know. And so you just add it to, or connect it with, something you've learned that's in just everyday life." They added the new ideas to things they had already learned, not necessarily to math. Diane made a very insightful observation, "To me it seems that the computer science is different; in a way it's an applied science. It's as conceptual as all the other fields that we study. We don't just study the English language, but we study it in order to use it. We do the same here except it's so much more involved because it takes in what we've learned in learning language skills, what we've learned in math. It's a science, a working component and a philosophy, you know, a thinking machine." While Betty was explaining that she did not understand pointers because she could not relate it to something she already knew, an "aha!" moment occurred and pointers "clicked." Her own words affirm, "I relate the different concepts in programming to what I already know, except for pointers. Because it doesn't exist, I mean you can't, [pause] you can't, [another long pause] well, actually, I guess you could. Because a sign on the street points to something else, so I guess I got it!" Betty excitedly laughed as everyone cheered. She then said, "I had initial problems with pointers because it didn't make sense."

Some of the things that made the respondents want to quit computer science were two-dimensional arrays (Abby), the amount of time it took for the programs (Carol), pointers and calculus (Betty), and the week of finals (Diane). Some of the biggest obstacles that they have had to overcome personally to remain in computer science are lack of transportation and lack of a computer at home (Diane), finding time to do the work (Betty), not having the compiler at home (Abby), and not being good at math (Carol). Some of the other obstacles they surmise that other females have had to overcome to remain in computer science include lack of reassurance, the math, and the course content difficulty. All four respondents have had significant people in their lives who encouraged them to stay with computer science, even when difficulties arose. All of them had support people to encourage them.

Some of the suggestions for encouraging more females to enroll in computer science included having all-female classes, having more female-oriented classes, stating the need for more females and the job opportunities, having friendly instructors who make the course fun and relate it to "female" topics, such as groceries or cleaning house, and having instructors who do not make fun of the females and who ensure that other students do not make fun of the females. Betty had one closing comment about being a female in computer science, "I like knowing that I can do something very few other people can do. And it makes me feel powerful."

In questioning the female respondents, the researcher was surprised to discover that most of the concepts taught in computer programming are not connected to math ideas as they are being learned, but rather to other things that have been learned in everyday life. The only concept that was connected to mathematics at all was the concept of a variable. All the respondents except Evelyn related the idea of a variable to the concepts they had learned in Algebra. Evelyn, the non-computer science major, related it to biology. The respondents related the rest of the concepts presented in the programming classes to personal experiences, not to mathematics, and those concepts or ideas that they could not connect to a personal experience were difficult, at best, to remember. As Diane said so succinctly, "I only know what I have lived."

All of the respondents related decision statements to consequences, and loops to some type of repetitive action. Abby, Carol and Evelyn said that they had difficulty learning argument passing in the programming language C++ because of the notational complexity. Diane related argument passing to a personal experience and said she understood it perfectly in the Basic language. Betty said argument passing simply made sense to her and did not consciously relate it to anything in her personal life. However, Betty also reported having great difficulty with pointers, because she could not relate it to anything in her personal experience until the focus group interview, when she connected it to something with which she was very familiar – signs that point to another location. It was at that point that Betty connected this concept to her deep internal schema by associating it with something familiar to her. Carol, who had just finished College Algebra, had the most difficulty relating programming concepts to things in her personal experience and could not see the relationship between mathematics and computer programming. Betty, who had completed Calculus II, had the least difficulty understanding the concepts presented in the first three programming courses, but she also did not see any relationship between mathematics and computer programming. Only Diane, who also had completed Calculus II, could see the relationship between mathematics and computer programming, even though she had taken only one course in programming. The respondent who had the most difficulty with programming was also the one with the least amount of mathematics, and those who had the least amount of difficulty were the ones with the most mathematics. Table 5 summarizes this data.

Table 5

Highest Courses and Concept Difficulty

Name	Highest Math Course Taken	Highest Programming Course Completed	Concept Difficulty
Abby	Trigonometry	Programming I with C++	Argument Notation
Betty	Calculus II	Programming II with C++	Pointers
Carol	College Algebra	Basic Programming	Argument Notation
Diane	Calculus II	Basic Programming	None so far
Evelyn	Business Calculus	Basic Programming	Argument Notation

When the respondents were asked how they integrated the computer programming concepts into their internal knowledge framework, Diane immediately responded with comments that indicated her understanding of the inter-relationship of computer programming with language and mathematics. Carol stated that she simply added the new knowledge on to what was already in there. Abby replied that she added the new ideas to or connected the new ideas with something similar that she had already learned in everyday life. Betty related the different concepts in programming to what she already knew.

CHAPTER V

SUMMARY AND CONCLUSIONS

Summary

The question of females majoring in computer science has resurfaced as the percentage of women earning degrees in computer science is still decreasing. This low level of participation of females in the computer field is a continuing concern for educators and for industry. There is a labor shortage in the field of computer science, and although fifty percent of college students are women, they are vastly underrepresented in the field of computer science (Bunderson & Christensen, 1995; Camp, 1997; Camp et al., in press; Dryburgh, 2000; Miller, 2000). Camp (in press) reports that this trend will continue and that ACM has strongly urged direct action to attract and retain more women to computing and computer science.

The main problem seems to be retention. The vast numbers of females who drop out of computer science have been attributed to many causes, but since 1990, research has focused on gender bias and stereotyping (Arenz & Lee, 1990; Gipson, 1997; Margolis et al., 2000; Spertus, 1991), societal factors (Ray-McCutcheon, 1996; McCullough, 1996; Joyce, 2000; Damarin, 2000), attitudes toward computing (Shashaani, 1997; Charlton & Birkett, 1998), gender grouping (Koch, 1994; Revilla, 1998; Sadker & Sadker, 1994; Crombie & Armstrong, 1999), previous computer experience (Bunderson & Christensen, 1995;Kirkpatrick & Cuban, 1998; Shashaani, 1997; Taylor & Mounfield, 1994; Fan et al., 1998; Mitra et al., 2000), "hard" computing versus "soft" computing (Butler, 2000; Clegg & Trayhurn, 2000; Henwood, 2000; Mahoney & Van Toen, 1990), learning styles (Katz et al., 1999; Severiens and Dam, 1997; Philbin et al., 1995; Price, 1996; Dunn et al., 1977; Dunn et al., 2001), and motivation (Dryburgh, 2000).

This research study investigated the characteristics of female computer science majors, their perceptions of how they have constructed their deep internal knowledge schemas, and their learning styles. Specifically, it investigated three research questions, which bear repeating at this point and which will be addressed individually.

First Research Question

The first question asked was, "What are the characteristics of the female students who have chosen computer science as their college major?"

Previous Computer Experience

Many studies have documented that males traditionally have more computer experience than females before entering a college-level computer science program (Kirkpatrick & Cuban, 1998; Shashaani, 1997). Of the five females interviewed, three had no computer courses before college. A study by Bunderson and Christensen (1995) concluded that a key factor influencing the high rate of female attrition might be lack of previous experience with computers, although in this study it did not seem to be significant. The females in this study did not express any anxiety or lack of confidence in using the computer, and this result corroborates the findings of Margolis, Fisher and Miller (2000), who found no correlation between prior computer experience and success.

Social Characteristics

All five respondents reported that they went to the instructor for help and asked their classmates for assistance rarely or never. This finding contradicts the study by Ray-McCutcheon (1996), which reported that female students have fewer interactions with teachers and thus receive less attention. Three of the five females interviewed preferred to work alone rather than with a study partner, and the other two responded that they could work alone or with a study partner or small group. This result correlates with the study done by Charlton and Birkett (1998), which found that programming students tended to be more introverted than business students.

Gender Bias

There is evidence that gender bias still exists in the male-dominated computer science classrooms. Of the five female students interviewed, two of the five felt comfortable in the male surroundings, were not hesitant to ask questions, and did not feel intimidated by the males; however three shared that the predominantly male environment affected learning, self-esteem, or both. They reported that they felt intimidated or were hesitant to ask questions in class because they "did not want to appear slow in front of the male students," that the male students "tended to congregate together and exclude the female students," and that the male students were difficult to consult when a female student had a problem with a program or concept. Similarly, Bundersen and Christensen (1995) found that students in advanced classes felt more comfortable asking questions than did those in the beginning classes and that females in all classes found it more difficult to ask questions in class than did the male students. This finding also corresponds with the gender bias found in the study of Margolis, Fisher and Miller

(2000), which reported that the women perceived themselves as "picking up ideas slower" than their male counterparts. It also reflects the same type of subtle biases that females face and subconscious behaviors that were found in the study by Spertus (1991).

Self-Confidence

All five of the female respondents expressed feelings of doubt about their own ability to do computer science. These feelings surfaced when the respondents ran into difficulties with programs or concepts. This result corresponds to Taylor and Mounfield's (1994) study that found that females tend to blame failure on their own personal inadequacies or lack of ability while males tend to blame external factors. All of the females in this study declared that at one time or another the thought of quitting crossed their minds. Shashaani (1997) also found in her study that females felt more uncomfortable with computers and had less confidence than males in dealing with them. Additionally, Henwood (2000) found that women students tend to underestimate their own technical skills relative to other measures of their technical competence.

Motivation

All of the female respondents in this study firmly stated their intention to finish the computer science degree, showing strong internal motivation. Two of the respondents have children, a factor which appeared as a very strong external motivational force. All the females reported some type of affirming relationship by a family member or close friend, which helped support and motivate them to continue in the computer science program. Charlton and Birkett (1999) also found that positive female computer attitudes had a bearing upon females' continued enrollment in the computer courses. It is interesting to note that when asked what they liked best about computer science, the females who have been successful in computer science did not respond by naming enhanced employment opportunities. They responded with adjectives such as "fun," "exciting," "variability of applications," and "imaginative." These responses support Charlton and Birkett's (1999) finding that successful female computer science majors display a real interest in the subject and they do not enroll in computer science courses because of enhanced employment prospects. The one female who is struggling with Programming I did not display much enthusiasm or liking for programming.

The emphasis in research has been primarily on external reasons that females drop out, and very little has surfaced concerning learning theory. Both mathematics and computer science are sequential in nature: new ideas and concepts are built on preceding knowledge (Skemp, 1987). Since foundational knowledge is of paramount importance, the first courses in computer science are strategic in building that foundation. It has been a common practice to use mathematical examples in teaching the concepts and ideas of computer science since they are so closely related. There is also a fairly rigorous mathematics requirement to fulfill for all computer science programming majors. In computer science, as well as mathematics, the higher order concepts and ideas are often very abstract, and these concepts and ideas must be built on the foundation of previously learned foundational theory and principles. To retain these higher order topics, the foundation must already be formed in the mind and deep schema of the learner, and the new topic must be firmly associated with some already learned object. This makes the learning of computer science and mathematics, especially in the beginning courses, very dependent on good teaching and leads directly to the second research question.

Second Research Question

The second research question asked, "Specifically, how do the female computer science majors perceive that they have constructed their internal knowledge of key concepts and ideas?"

Computer skills are often associated with mathematics skills because both depend on logical and sequential thinking abilities. It is not uncommon to find the computer science department joined with the mathematics department in many educational institutions. In this study, however, the connection between computer programming concepts and mathematics did not emerge. All five female respondents related the concept of a variable to the same idea in mathematics, and one female related the idea of looping to a mathematical concept. Of all the other programming concepts taught, all of the females related it to something in everyday life, not a mathematical concept. In fact, only one of the respondents could see any relationship between mathematics and computer programming at all. In every case, difficulty in understanding a programming concept was directly related to the respondent's not being able to connect it to something already familiar in her everyday experience. It is very interesting that Charlton and Birkett (1998) concluded that teaching programming in a non-mathematical context would be particularly helpful in stimulating female interest. Clegg and Trayhurn (2000) reported that over the last decades, academic institutions have typified computing as a reliance on mathematical formalism, emphasizing such areas as artificial intelligence and formal mathematical methods. They believe this emphasis is misplaced, and this study supports their position. ACM has traditionally emphasized "hard" areas such as mathematical formalism in its approved computer science curriculum for university programs.

Learning theory also includes the possibility of different learning styles (Gardner, 1991). It is possible that female computer science majors have different learning styles than their male counterparts, or that computer students have different learning styles than do students in other disciplines. This study also researched the question of learning styles as it pertains to female students majoring in computer science, which is the third research question.

Third Research Question

The third research question inquired, "Do female students who major in computer science have a different combination of learning styles from those female students who have dropped out of computer science?"

In the interviews, all of the female respondents reported themselves as being hands-on type learners. Evelyn, the business major, had higher auditory, visual, and tactile scores on the PEPS than the means of the other four respondents, but she scored lower on the "learns in several ways" variable. There was no significant difference in the kinesthetic scores. These results are shown in Illustration 2.

The lower visual scores do not agree with Katz et al. (1999), who found a tendency for visual learners to be more at ease with computers. However, it does correlate with the study by Philbin et al. (1995), which found that females learn better in hands-on and practical settings and that traditional educational settings, which are abstract and reflective, may not be the best learning environment for females. Magolda (1989) reported that women prefer concrete experience to abstract conceptualization, matches the results of this study. In this area of learning styles, no significant differences were found between the non-computer science major and the four computer science majors. Since all scores except one were in the 40 to 60 range, no conclusion could be drawn.

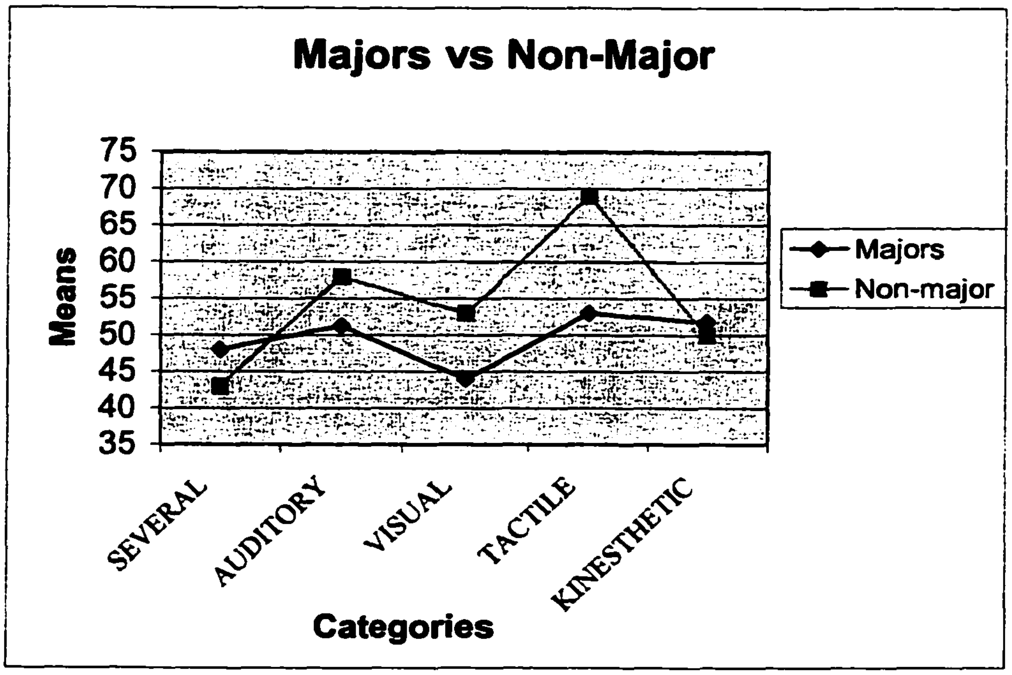

Figure 2. Comparison of Learning Styles

Conclusions

This study found that previous computer experience was not a significant factor in the success of females in beginning programming courses. This finding suggests that females with no previous computer experience can do well as computer science majors, and that this lack of experience should not be a detriment to their success. The isolation reported by the females should be a cause for concern among educators, and several suggestions were proposed by the subjects to alleviate this situation. Study partners, group study, and some type of support group were all mentioned by the subjects as ways to alleviate the isolation. Ray-McCutcheon (1996) reported that females tend to benefit more from cooperative activities and activities that are less rushed than male students do, correlating with their proposal of a support group for female students in computer science.

Gender bias is still a major issue in computer science. The male environment in computer science is still perceived as cold and unresponsive to female students. Female students hesitate to ask questions in the male-dominated classes, and especially of male computer science teachers. Both learning and self-esteem are affected by these constraints, and these contribute to the attrition of females. Several suggestions were made by the subjects of this study for all-female classes taught by female instructors. This concept is soundly supported by research studies conducted by Melissa Koch (1994) and Anita Revilla (1998), which suggest same-gender classrooms in the public school systems. Crombie and Armstrong (1999) report that females enrolled in an all-female high school science class perceived significantly higher levels of teacher support, computer confidence, intrinsic motivation, and future academic intentions than females from mixed gender classes. Across all stages, the benefits of all female groupings include improved task performance and more positive attitudes (Dryburgh, 2000). At the post-secondary level, Dryburgh (2000) found that the benefits of all-female groupings relate to social facilitation where women encourage one another. Such groupings would also help assuage the tendency toward isolation that was reported by the females in this study. McCullough (1996) found that female participation levels were slightly higher in female-instructed classrooms than in male-instructed classrooms, supporting the suggestion of more female instructors in computer science.

Since the females in this study did not relate the majority of computer science topics to any mathematical concepts, instructors of beginning computer science courses must rethink the examples used in class. Formal mathematical methods are not appropriate in teaching female students in beginning computer programming courses. This study found no relationship between mathematics and beginning computer science topics. Examples for these students, then, need to be pulled from everyday experiences, not from mathematics. The concepts that were related to everyday experiences were cemented to a deep conceptual schema, albeit not a mathematical one, but those concepts that had no connection in the subject's experience were not learned well at all and gave the female subject much difficulty. Examples, then, to be effective for these female students, need to be drawn from common human experience and not from abstract mathematical logic.

It is well documented that women are attracted to computer courses that emphasize social issues and computer applications (Henwood, 2000). Degrees that combine business, social sciences, or humanities with computing have been found to attract more women than traditional computer science degrees (Mahoney & Van Toen, 1990). This would indicate that teaching these beginning programming classes in a non-mathematical context would seem to be part of the solution to retention of females in computer science.

Encouraging more females to major in computer science has been an ongoing effort for over two decades. Many times females must overcome almost insurmountable odds to pursue a college degree. Coupled with the gender bias still found in computer science, it is not surprising that many females become discouraged.

With the conclusions, implications, and recommendations for further research that follow, this study proposes a working model to increase the retention of female students in computer science.

Support Network

This study brought to light the need for some type of support network targeted specifically for females in the computer science area. Although the females interviewed expressed no desire for study partners, they did describe the ideal computer lab setting as being warm and friendly, a place where casual conversations could take place to help one another and where a friendly person who knows the programming language is available for questions.

Female Mentors

Another suggestion proposed by the respondents was the presence of at least one female instructor who would be willing to spend time mentoring those female students in the computer science department. This would provide a "safe" place for the female students to ask questions and not feel that they are "stupid" or "slow" if they did not understand a programming concept right away. Gender bias is still a problem in the male-dominated classrooms, where sarcasm and intimidation are still found. Some female students still feel hesitant to ask questions in that environment, and these females tend to participate less in the class. This might be a contributing factor as to why the female students prefer to work alone.

Non-mathematical Context

One other important result of this study is that female students do not relate the new concepts and ideas of computer programming to mathematical ideas, with the exception of the concept of a variable. The deep mathematical schemas proposed by Richard Skemp as the foundational internal structures onto which students connect new mathematical ideas do not appear to be the foundational schemas for computer science in the females studied. The foundational schemas for these computer science females are the deep internal structures created by everyday life experiences, and it is these experiences that are used to understand the new, and often confusing, concepts of computer science. The females who cannot find some experience in the deep structures with which to relate the new concept or idea find these new concepts or ideas very difficult to grasp. The importance of finding something in everyday life to use as an example to teach the computer programming ideas and concepts cannot be overstated. So many times computer science instructors use mathematical examples to explain new ideas or concepts, and, at least for the females interviewed in this study, these examples are not meaningful.

Female-Friendly Environments

In order to increase the number of females graduating with computer science majors and entering the field of computer science, computer science courses and departments at all educational levels must be perceived as being more friendly to women. If instructors were more cognizant of the fragile position of women in computer science departments, teaching methods could be adapted to encourage more involvement in class discussion, and class instructors could take more care to encourage and respond positively to questions asked by female class members. The challenge of increasing the number of females who have positive feelings toward mathematics, who enjoy programming and interacting with computers, and who look forward to acquiring the skills needed for computing and associated careers must be fully and eagerly embraced by the educational community at all levels (Bunderson & Christensen, 1995).

Implications

In this study, the way female computer science students learned the beginning concepts and ideas followed the design proposed by Richard Skemp. In building up successive levels of ideas and abstractions, if a particular level was imperfectly understood, everything from then on was in peril. It was very clear from the interviews that each programming concept was connected to or embedded in a deep internal structure of other concepts and was part of a hierarchy. The internal structures that the female students constructed in the beginning programming courses were crucial to the ease or difficulty with which they mastered the later topics. New ideas and concepts that found a connection within the existing internal structure were much better learned and remembered, and those ideas and concepts that found no connection were largely not learned at all.

This study found that learning computer science is as sequential as learning mathematics and that new ideas and concepts must be assimilated into an appropriate schema. The first programming courses were found to be strategic in their importance to long-term success of the subjects. The beginning concepts and ideas were foundational for learning the rest of the concepts of computer science and must have been integrated into a deep internal knowledge structure. This study found that when these deep connections were not constructed, the student did not progress. Such was the case with the subject who had attempted Programming I with C++ two times with no success.

This illustrates the importance that the connections between the internal structure and the new idea or concept being presented must be strong enough for the new idea or concept to be attached to this deep structure. If not, the surface knowledge of the subject is increased, but when the more complicated concepts of computer science come along, there will be no deep structure to which they can be attached. The result for this one subject in particular was an inconsistent understanding of basic computer science concepts and long-term difficulty in retention of these concepts. The internal knowledge structure to which the computer ideas and concepts were attached was not mathematical, although it followed exactly the learning pattern as described by Skemp. For the female subjects, the deep attractors in their own internal knowledge structures were previous life experiences that were related in their minds to the new idea or concept being taught. It was very clear in the interviews how the subjects connected the programming ideas to personal experiences in their own lives, and when those connections could not be made, then incomplete or nonexistent understanding was the result.

Recommendations

Mentors

Several areas of interest have emerged from this study that could be worthy of further investigation. First, further investigation needs to conducted at the undergraduate level into having female instructors, female graduate students, or upper class female students act as mentors for all levels of female computer science students to see if this practice would result in more retention of females in the computer science program.

Support Networks

The isolation of female computer science students is a cause of great concern, and an investigation into support networks is a second area of further research. It is possible that forming support networks of female students would result not only in less isolation, but ultimately in more female students completing the computer science degree.

All-female classes

The third area proposed for further research is all-female classes taught by female instructors. The numbers of female instructors in computer science at the college and university levels are slowly increasing, and the reality of this type of research is closer than several years ago when males dominated the instructional arena. It is possible that further data could reveal that creating "female friendly" environments is another key to attracting and retaining females in computer science.

Math Requirements

It would behoove the educational community to rethink the mathematics requirements for a degree in computer science, or at least to consider postponing the calculus series until the junior or senior year when the relationships between computer science and mathematics will be more readily visible to the female students. The prerequisite for discrete mathematics is college algebra, well within the reach of the female computer science students. A suggestion for further research would be the placement of discrete mathematics and the calculus series in the computer science curriculum. A possible scenario would include the beginning computer science courses (introductory programming, programming I and II), college algebra, and discrete mathematics at the freshman/sophomore level. Then the programming curriculum would continue, and the calculus series and/or other mathematics courses would be integrated at the junior/senior level.

This calls for additional research, although the new university computing guidelines proposed by the Association for Computing Machinery (ACM, 2001) in their Steelman draft of the Computing Curricula 2001 question the importance of and necessity for the extensive mathematics courses. In section 7.4, *Integrating discrete mathematics into the introductory curriculum*, the ACM document states, "...the CC2001 Task Force believes it is important for computer science students to study discrete mathematics early in their academic program, preferably in the first year." (http://www.acm.org/sigcse/cc2001/ steelman/cs-introductory-courses.html)

Later in section 9.1.1, *Mathematical Rigor*, the ACM lists recommendations for the mathematics necessary for a degree in computer science. It states

> The CC2001 Task Force makes the following recommendations with respect to the mathematical content of the computer science curriculum:
>
> *Discrete mathematics.* All students need exposure to the tools of discrete mathematics. When possible, it is best for students to take more than one course in this area, but all programs should include enough exposure to this area to cover the core topics in the Discrete Structures (DS) area. Strategies for integrating discrete mathematics into the introductory curriculum are discussed in section 7.4.
>
> *Additional mathematics.* Students should take additional mathematics to develop their sophistication in this area. That mathematics might consist of courses in any number of areas including statistics, calculus, linear algebra, numerical methods, number theory, geometry, or symbolic logic. The choice should depend on

program objectives, institutional requirements, and the needs of the individual student.

(http://www.acm.org/sigcse/cc2001/steelman/cs-completing-curriculum.html)

Learning Theory

Much more investigation is needed in the area of learning theory in computer science. Do males relate computer science topics to everyday experiences, or do they relate them to mathematical concepts and ideas? Little is known of learning theory in computer science, and this lack opens up an entirely new vista in education. As Richard Skemp paved the way for the Mathematical Standards of the new millennium, perhaps the discovery of new learning theory in computer science could forever eliminate the problem of gender inequities in computer science. Is this science fiction? Possibly, but what was impossible less than one hundred years ago now takes astronauts into space, to the moon, or to a space station that was not even imagined except in the minds of dreamers.

It is fitting that computer programming was begun by August Ada Byron, a female mathematician who wrote programs for a computer that did not exist, except in someone else's mind. Computer programming was furthered by another female dreamer, Rear Admiral Grace Murray Hopper, who created the first compiler, forever changing the way computers were programmed from machine language to English, paving the way for all high-level languages that followed and winning the computer science "Man of the Year" award in the process. It is time for another dreamer to take up the baton that is being offered and forge ahead in the territory called gender inequity to call young females into a discipline that is theirs by birthright.

APPENDIX A

PERMISSION TO USE HUMAN SUBJECTS

THE UNIVERSITY OF SOUTHERN MISSISSIPPI

HUMAN SUBJECTS PROTECTION REVIEW COMMITTEE
NOTICE OF COMMITTEE ACTION

The project has been reviewed by The University of Southern Mississippi Human Subjects Protection Review Committee in accordance with Federal Drug Administration regulations (21 CFR 26, 111), Department of Health and Human Services (45 CFR Part 46), and university guidelines to ensure adherence to the following criteria:

- **The risks to subjects are minimized.**
- **The risks to subjects are reasonable in relation to the anticipated benefits.**
- **The selection of subjects is equitable.**
- **Informed consent is adequate and appropriately documented.**
- **Where appropriate, the research plan makes adequate provisions for monitoring the data collected to ensure the safety of the subjects.**
- **Where appropriate, there are adequate provisions to protect the privacy of subjects and to maintain the confidentiality of all data.**
- **Appropriate additional safeguards have been included to protect vulnerable subjects.**
- **If approved, the maximum period of approval is limited to twelve months.**
  Projects that exceed this period must submit an application for renewal or continuation.

PROTOCOL NUMBER: **21042301**
PROJECT TITLE: **Gender underrepresentation in beginning computer programming courses**
PROPOSED PROJECT DATES: **05/07/01 to 09/30/01**
PROJECT TYPE: **Dissertation**
PRINCIPAL INVESTIGATORS: **Margaret West**
COLLEGE/DIVISION: **COST**
DEPARTMENT: **Science Ed**
FUNDING AGENCY: **N/A**
HSPRC COMMITTEE ACTION: **Exempt - Approved**
PERIOD OF APPROVAL: **05/02/01 to 05/01/02**

Mitchell E. Berman, Ph.D. — 5/6/2001 (Date)
HSPRC Co-Chair
The University of Southern Mississippi

RESEARCH AND SPONSORED PROGRAMS
Box 5157 · Hattiesburg, MS · 39406-5157
Phone (601) 266-4119 · Fax (601) 266-4312
www.usm.edu
Hattiesburg • Long Beach • Ocean Springs • Biloxi • John C. Stennis Space Center

APPENDIX B

PERMISSION TO REPRODUCE TABLE

Subject: Re: Permission to reproduce table

Date: Sun, 15 Apr 200120:17:31 -0500

From: "Gary E. Price" <gprice@ku.edu>

To: William West wgwest@bellsouth.net

You have permission to reproduce the table. Gary

William West wrote:

> Dear Dr. Price,
>
> I am a doctoral student in Science Education at the University of
> Southern Mississippi. I am studying gender inequities in the field of
> computer science. My dissertation includes a section on learning styles
> and I am using the PEPS in my study. I need permission from you to
> reproduce the reliability numbers (r values) table from your PEPS
> manual. This table is on page 40 of the PEPS manual, and I would like
> to include the entire table in my dissertation.
>
> Thank you very much for your consideration. I look forward to hearing
> from you.
>
> Sincerely,
>
> Margaret L. West
> 2098 Woodfield Lane
> Biloxi, MS 39532-3349 > (228) 388-8243
> wgwest@bellsouth.net

APPENDIX C

INDIVIDUAL INTERVIEW AGENDA

1. Welcome

2. Overview of the topic and ground rules

3. Teaching/Learning questions

   a) Of all your schooling, elementary through college, who has been your favorite teacher/instructor? Why?

   b) Of all your high school or college classes, which type of learning activities do you prefer? From which learning activities have you learned the most? Why?

   c) Did the fact that the majority of your classmates are male affect your learning or understanding of the material? If so, how?

   d) With mostly males in the class, did you feel intimidated? Did you feel hesitant to ask questions? Why or why not?

4. Content difficulty questions

   a) How much previous computer use did you have prior to your first college programming course?

   b) How do you rate the Introductory Programming course in terms of difficulty on a scale from 1 to 5, with 5 being the most difficult and 1 being the easiest? Why?

   c) How do you rate the Programming I course in terms of difficulty on a scale from 1 to 5, with 5 being the most difficult and 1 being the easiest? Why?

   d) Where do you go for help when you need it?

5. Social Interaction questions

   a) How would you describe yourself socially?

b) Where do you primarily work on your programs and assignments?

c) Who works with you on the programs and assignments? Describe your working environment.

d) Do you ask questions in your computer class when you do not understand or do you just try to figure it out on your own?

6. Content complexity issues

a) What topics in computer science gave you the most difficulty?

b) What did you do to overcome the difficulty?

c) Talk about the following topics in computer science in terms of difficulty: variables, if/then statements, looping structures, subprograms, arrays, argument passing.

7. Persistence/Satisfaction questions

a) As the programming course progressed, did you think that the content became more difficult? If so, how did this affect your self-confidence?

b) How did the content difficulty make you determined to succeed?

c) What steps did you take to understand the concepts and ideas as the programming course became more complex and difficult?

8. Motivation questions

a) Why did you originally choose computer science for your major?

b) What are your reasons for staying with computer science as your major?

c) What are your future goals?

d) What do you like most about computer programming?

e) What do you like least?

f) If you could change one thing about the programming courses, what would it be?

g) What would you suggest to encourage more females to enroll in computer programming courses and possibly consider it for a major?

APPENDIX D

FOCUS GROUP AGENDA

1. Welcome
2. Overview of the topic
3. Ground rules
4. Opening question
    a. Tell us your name and one thing you enjoy doing.
5. Introductory questions
    a. How much computer experience did you have before enrolling in the community college?
    b. What are some of the difficulties you have encountered in learning computer programming?
    c. How do you rate the programming courses in terms of difficulty?
    d. What is your idea of a good instructor?
    e. Describe your ideal classroom or learning environment.
6. Transition Questions
    a. How does having mostly male students in your computer programming classes make you feel?
    b. Do you ask questions in the programming classes? Other classes? Why or why not?
    c. How often do you require help with new concepts or programs?
    d. Where do you go for help with the concepts? The programs?
    e. Do you have a study partner or group?

f. Do you like to work in a group computer lab setting or do you prefer to go home and work by yourself?

g. Are you satisfied with your level of understanding or do you feel you have missed something along the way?

7. Key Questions

a. How did you build the necessary mental framework for understanding computer programming?

b. Which topics were most difficult for you to grasp? Describe your difficulties and how you overcame them to achieve understanding.

c. Did you encounter any difficulties that made you want to quit computer science? If so, what were they? Why didn't you quit?

d. Was your confidence in your own ability affected by these difficulties? If so, how?

e. What are the biggest obstacles you have had to overcome to remain in computer science?

8. Ending Questions

a. What is your motivation for continuing with computer science as your major?

b. What do you like best about computer programming? Least?

c. If you could change one thing about the way computer programming is taught, what would it be?

d. What would you suggest to encourage more female students to enroll in computer science?

APPENDIX E

Interview with Abby (Excerpts)

R – Researcher A - Abby

R … of all your schooling, elementary through college, who has been your favorite teacher or instructor?

A … I would have to say my Power Tech teacher when I was in high school.

R Okay, why?

A He kept it interesting. … I had a lot of fun with it. We got to do things hands-on, and take things apart and put them together, and all kinds of things. … I would have to say he was my favorite.

R Of all your high school or college classes, which type of learning activities do you prefer?

A Hands on.

R From which learning activities have you learned the most?

A … again hands-on.

R … Explain that.

A It's hard for me to read something and always, you know, or just unless it's like a novel or a story, it's hard for me to read it and understand it. And at the same time, I can't listen to someone stand up there and explain step by step and me still remember everything and get everything, so it's easier for them to explain while I am doing it, 'cause then I remember it easier.

R That makes sense. Did the fact that the majority of your classmates are male... affect your learning or understanding of computer science material?

A Not at all.

R How did you feel in a class with mostly males?

A Comfortable. Actually, I am more comfortable around men than I am women.

R Why is that?

A I was raised in a house with four boys. So, I am used to boys. [Laughs]

R ...With mostly males in the class, did you feel intimidated?

A No.

R Did you feel hesitant to ask questions?

A No.

R So it didn't affect you in any way?

A Nope, not at all.

R Okay. Did they ever say anything to make you feel ... less important or did they put you down because you are female in any way?

A In a joking manner. You know, I mean it wasn't intended personally or even very seriously. But that happens everywhere, unfortunately.

R ... how much previous computer use did you have prior to your first college programming course?

A ... about six months, and that was like ten years before I started college.

R ... Talk about that.

A ... when I was twelve, we got a Commodore 64 computer for Christmas, and I spent about six months playing with the programming and stuff like that. But that was like fifteen years before I started college. Long time ago, and I don't remember a lot of it.

R You've had how many computer science courses?

A Uh, two – Basic Programming and Programming I.

R All right. How would you rate the Beginnning or Basic Programming course in terms of difficulty on a scale from 1 to 5, with 5 being the most difficult and 1 being the easiest?

A I would have to put it on a two.

R A two. Okay, explain...

A I am a very logical person. ... it seems a lot like Algebra ... and I'm really good at math, and it makes it real easy for me. ... some people pick up French; I pick up computer language.

R That's great. So it's just really naturally easy for you. Okay, how do you rate the Programming I course, the C++, in terms of difficulty on a scale from 1 to 5?

A ... I rate it a three. It was a little bit harder. We had a lot more programs to do and ... changing from one language to the other made it a little bit more difficult, because once you've already got that one language set in your mind, it's sometimes hard to remember the newer information.

R Okay. Where do you go for help when you need it in your programming classes?

A To the instructor.

R Do you ever go to the other students in the class for help?

A Occasionally. ... but there were only one or two that could really explain the answer to me. [Laughs]

R Was that because there were only one or two that knew, or just one or two that could explain it in a way you understood?

A One or two that really knew for sure.

R … How would you describe yourself socially?

A Quiet. I don't go out much; I don't do much. [Laughs]

R … Where do you primarily work on your programs and assignments?

A In the lab. Here at school.

R You do not work on them at home?

A … no. I did with Basic because I have Basic on my computer. But I … wasn't able to get a hold of … a compiler for C++. So, … I just worked on them in the lab.

R Right. And that way, Mrs. Scafide or I sometimes would be available …for help.

A … yeah.

R Does anyone work with you on the programs and the assignments? In other words, did you have a study partner or a programming partner?

A … in Basic I did. Pete and I were study partners and we usually did most of our programs together. But in C++, not really, no.

R Describe your working environment. How do you like to work on programs or assignments?

A … that's kind of hard for me because I get along either way. I mean, I can tune things out or I can sit and talk with other people and do it at the same time... I wouldn't want a TV sitting next to me, or a radio sitting next to me playing, because that would sidetrack me, but as far as in a working environment with other people, either way works.

R So that's why you could work here in the lab and the noise didn't bother you.

A Right.

R Do you ask questions in your computer classes when you do not understand or do you just try to figure it out on your own?

A A lot of times, I'll ask questions when I don't understand, but I also ask questions when she hasn't explained the next step and I am curious about it…, and I'll ask a question about an aspect that she hasn't gone over yet but it seems logical to me as the next step and I want to make sure.

R … What topics in computer science gave you the most difficulty?

A Arrays.

R … Why?

A Because you have a lot of different ways of putting the arguments in and what type of [arguments], like the name, and some of them had to have the name and the dimension, and some of them just had to have the name. And sometimes you had to have the name, the dimension and how many you wanted in it, or how many you wanted on a line. It made it harder… for me to remember which ones go together.

R Right. What did you do to overcome the difficulty?

A Um, She gave out some … handouts, and I would look back on my handouts and check to see how I had to put it in to make sure I had it right.

R ... Talk about the following topics... The first concept that you are taught is the concept of a variable. ... Was it difficult? Did you understand it?

A Oh, yeah, yeah, because you have variables in just about everything in life. I mean, variable is just something that changes on a regular basis. You know, ... if you have even Algebra, or Beginning Algebra, you know what a variable is, so that was real easy.

R Okay, so you related that to something you already knew in math and in life? [She nods] Okay, what about If-Then statements?

A Oh, well, that's easy. See, I have children, so you have to say a lot of if-thens. If you do this, then you get this, if you don't, then you don't. [Laughs] There's a lot of if-thens at home.

R All right. What about looping structures?

A Looping structures were not too difficult except figuring out what was going to break the loop. And that gave me a little trouble. ... she put a statement up there one time that had ... one statement under it, no brackets, no nothing, and I'm like, "Well you don't have a test statement there," and she said, "Well, it automatically knows," and I'm thinking, "Why?" [Laughs] But, I had a little trouble figuring out when it was going to break the loop.

R Okay. How did you relate, or what did you relate looping structures to in your own life?

A ... I didn't have a lot of information to relate looping structures to, um, so it was more like a math thing, where I put it in an if-then sort of context.

R ... What about sub-programs? Like functions and subroutines?

A	I don't like functions. [Laughs] I can do them, and they're pretty easy, um, except when you're having to put prototypes up at the top and the parameters change for the prototypes and the call and function headers, but, other than that as far as the actual function, that was easy.

R	Okay, why was that easy for you?

A	Because it was regular programming. It's the same thing you did in the main program; you just put it in a different section.

R	Okay, so you related that to something you had done previously in programming.

A	[She nods] Right.

R	Okay. All right. What about the argument or parameter passing?

A	Arguments ...That...I had trouble deciding when ... it would change. ... I knew the difference between a local and a global variable, but I could never figure out when it was going to change the global variable. [Laughs] I had trouble with that.

R	All right. And then, arrays.

A	I hate arrays, but they are a necessary evil because ... you can try to read in... a list of five hundred names when you have no structure to reread it or to put it in a position or okay, I can go back and find this. Then it would take you ten to fifteen times as long as putting it in an array. I mean, you can have up to... many different dimensions in an array when, to try to do it on a regular read and go back and find that information, you may never be able to find it again. So, I don't really care for arrays, but they are a necessary evil.

R	Okay. ... I noticed once we got to looping structures, there wasn't too much in everyday life to relate that to, and subprograms you were relating to previous programming. ... the passing of parameters, did you relate that to anything you were familiar with?

A ... electrical lines. It's kind of like that. When you have a global variable and use it in the argument list, it's ... the difference between using a global and a local variable. If you use the local, then the electrical circuit is broken, and if you used the global, it wasn't.

R Okay, what about arrays? Could you relate that to anything in your everyday life?

A Yeah. But I wouldn't necessarily say everyday life, it's more like... I could relate it to a calendar, but it was more like a list of names on a schedule or a list of words, like a list of grades on a grading sheet.

R Is that how you learned the concept of arrays?

A ... not exactly. We did the row and column thing, but ...if I had to think about how to relate it, you know, that would be the best way for me to do it.

R So you never actually related it to anything you were familiar with when you were learning it.

A Not at the time of learning, no.

R Okay, that's interesting.

A That's probably why I don't like arrays much. [Laughs]

R You know, that's a thought. As the programming course progressed, did you think that the content became more difficult?

A Yeah, yeah. It seemed like I breezed through, and this is in both courses, it seemed !ike I breezed through the whole semester until I got to the last program and then I was just sort of lost. [Laughs]

R How did this affect your self-confidence?

A … it took it down a notch. When you get lost on a program, or just go blank, you start to wonder, you know, am I in the right field? You know, am I sure I can do this? So, you get through it, you turn the program in, you get the grade, and then you go, "Oh, I did better than I thought I did."

R And what did that do for your self-confidence?

A … it took the notch back up. [Laughs]

R That's what I wanted to know. Since you have successfully completed two classes, what I want to know is [how] did the content difficulty…[affect] your determination to succeed?

A … it didn't really, I mean it touched on it when… I had trouble with the last program. And when I got through with the last program I thought, "Well, I can do this," so… it didn't really hurt my determination to succeed at all.

R Right. Then once you made it through that rough spot, then how did you feel about that?

A … all that more determined to finish it out.

R Okay. What steps did you take to understand the concepts and ideas as the programming course became more complex or difficult?

A … it was a lot of looking back at the handouts that were given. … I didn't really open the book and do the reviews a lot, but… the handouts were easily found. I had them in my notebook so that I could refer to them and remind myself about that.

R Did you ever take them to a computer and work on them, because you said at the beginning you were hands-on, so did you take those worksheets and work them out on the computer?

A Actually, no, I never actually had to put the programs that were on the handouts in the computer. I would ... just glance ... back, like at a load array ... mostly it was with arrays and with the randomize [function] that I would have to use the handouts.

R Just to remember the syntax or the structure?

A Right, right. [She nods]

R All right. Here are some questions about motivation. Why did you originally choose computer science for your major?

A I enjoy playing with computers, and ... computers is where the money is. That might sound a little, you know, self-serving, but I'm a parent with three children and everything in the world is going to computers. And I had to find a way to make sure that I could support them on a regular basis, you know, and support their future, and that being a way to do it with money.

R What are the reasons for staying with computer science as your major?

A It's fun! And I enjoy creating programs on the computer. I am still worried about graphics, 'cause I'm not very good at drawing, but I'll get there. [Laughs]

R You'll get through that. What are your future goals?

A ... my goals are pretty easy. ... to get a position that I can retire at, that would support me and my family, and just enjoy the rest of life.

R ... What do you like most about computer programming?

A The many different things you can do with it.

R Such as?

A ... you can be a graphics design person... you can specialize in programming, or in particular areas of programming like bank programming or... in the scientific field, you could be an analyst. There are thousands of different varieties of specialties in the computer industry, all of which have a computer science degree.

R What do you like least about computer programming?

A ... pseudocode. Having to write the pseudocode or flowchart or however you want to put it. You know, like I think there is ... a third way, but I don't remember what it was. [Laughs] But... oh, yes I do. Top down charts. But that's having to go back after you've written the program... and take the time to write the pseudocode. I don't enjoy that at all.

R If you could change one thing about the programming courses, what would it be?

A ... make each one have ... a game as a program, like a video game... but to put an aspect of at least five of the different fields in each programming class so they can see how each one would be used.

R ... a good idea. Okay. What would you suggest to encourage more females to enroll in computer programming courses and possibly consider it for a major?

A ... I don't have a lot of suggestions as far as females go because you either pick it up very easily or it's very, very hard... and ... I don't really think gender makes a difference as far as... how you pick it up. One way or the other, I think... that on a general basis, and ... there's no statistics for this that I know of but, I think on a general basis men are more ... mathematical and logical than women are, or think they are, any way. [Laughs] ... I can't think of anything that would make it more interesting to a specific gender than it already is.

R Have you noticed that in the beginning of the computer science classes there are ... I wouldn't say a lot, but a goodly number of females. And then, as the classes progress, it's the females that drop out?

A Yeah... when we went through Basic, there were... I'd like to say like four women in our class...

R Uh huh, four or five.

A And then ... when I got to Programming, there was only two of us, and the one wasn't one of the ones that were in my Basic class. [Laughs]

R So, out of that whole class, you were the only one that went on to Programming I?

A Yes, I was.

R Why do you suppose that's so?

A ... I think the ones who took the Basic, although I don't know ... the reason for them taking it to begin with, but I think the ones who took it decided that it was harder for them than they thought it was going to be. ... and I think that's what made them drop out 'cause they figured that the next step in the programming would be harder than what they were doing then. Which, to an extent it was, but not, not a lot. Yeah. [Nods]

R Okay. Well, I believe that's it, unless you can think of anything that you want to say about the number of females in computer science and why you have stuck it out where others haven't.

A Well, it doesn't seem that hard to me, and it's a goal that I've set for myself and I intend to finish. You know, I may not finish at the top of my class, but I will finish it. So… that's about it … I could think of a sign set up in the lab "If you are female and good at math, come to Computer Programming."

R Well … thank you very much

[End of interview]

APPENDIX F

Interview with Betty (Excerpts)

R – Researcher B - Betty

R ... Of all of your schooling, elementary through college, who has been your favorite teacher or instructor, and why?

B ... I guess my world lit teacher in... Keesler, because he was so good in telling stories. It didn't take me any effort to learn. I just... listened to his stories and I remembered.

R Okay. Of all your high school or college classes, what type of learning activities do you prefer? [Puzzled look] Do you understand what I mean by learning activities?

B Yeah, it's just hard for me because ... I wasn't very interested in studying when I was younger, and it's only been this past two years when I really tried. So... I guess it's ...reading, sometimes, and then writing, and, you know various things. There's really no fixed method that I prefer.

R From which learning activities do you think you have learned the most?

B ... hands-on.

R Hands on?

B Yeah.

R Did the fact that the majority of your classmates in computer science are male... affect your learning or understanding of the material?

B No, but… I find that now there is only just me, I don't talk. And…I just listen, but I still learn from what they say because they say a lot weirder stuff than what… I'm normally used to. And it's been interesting.

R With mostly males in the class, do you feel intimidated?

B No.

R Do you feel hesitant to ask questions?

B Yes.

R Okay, why? Talk about that.

B … I guess I didn't want to look stupid… but most of the time it's because they answer so much faster than me. You know, I didn't have to ask any questions. … I'm mostly hesitant because… what if they… kind of look at me like, [makes a sneer] like, yeah.

R Look down on you?

B Yeah.

R All right. This is about content difficulty. … you've noticed, I'm sure, that when you were in Basic, there were more females. And as you've traveled through the computer science… you notice the females tend to drop by the wayside.

B Yeah.

R … so I want to explore some content difficulty issues. How much previous computer use did you have prior to your first college programming course?

B … when you say college programming course, do you mean QBasic?

R Any college programming course.

B None.

R Okay. ... how do you rate the basic programming course in terms of difficulty, on a scale from 1 to 5 with 5 being the most difficult and 1 being the easiest?

B One.

R One. Okay, talk about that. Tell me why.

B It's just all logic. You don't really have to...you have to think a little bit, but not that much. You know, normal arithmetic and, you know, basic skills. You don't have to work very hard. [Laughs]

R Okay. How do you rate the Programming I course, the C++ Programming I course, in terms of difficulty – same scale.

B Um, about three.

R About a three. Okay, talk about that.

B It, it needs a lot more math. And ... the language was a little hard for me... I think one was because ... I didn't have as much contact with it. I didn't have it at home. But it's mostly the math that kind of stumbled me.

R Okay. What about Programming II? How would you rate that?

B That was difficult. That was like a five. ...Pointers.

R Pointers.

B Pointers is what killed me. It took me a month, or a few months, to finally understood it, understand it. ... at first, you know, I couldn't quite grab the concept of what, you know, memory, you know, it pointing to memory. I was trying to assign things to it, and it didn't work out. It, it took me a while but I think I got it.

R Good. When you were having difficulty, where do you go for help when you need it in programming?

B [the instructor]

R The instructors, then. Okay. Do you ever ask your classmates for help?

B Once in a while.

R Okay. What kind of help do you ask from them?

B Mostly, like, suggestions, you know, and comparing notes and stuff. Comparing programs, and seeing what they do and what I've done wrong.

R All right. Now this is about your social interactions. How would you describe yourself socially? Are you … a loner, or a social butterfly? Talk about that.

B … I have to break it down because… before I started really getting into my studies, I was…like a social butterfly. I'd make friends with a lot of people, but most of my friends were males. I guess that is why I don't feel uncomfortable around males. But since I… started school here, I have not talked to very many people, and I guess I'm a loner now. The only people I talk to is my son and my husband.

R Why do you think that's so?

B … I guess because I'm always thinking about stuff. I'm not really interested in talking to people really that much now.

R You think maybe you're concentrating on school work?

B Yeah. I find myself thinking a lot about school work and even when I don't know that I'm thinking, I'm actually thinking about problems I have to solve, and…I'll wake up in the middle of the night and go Ah! [Laughs] That was what's wrong!

R I've done that myself.

B And … one time I dreamt about school work, about… what was wrong with it. That was pretty bad.

R When you woke up, was that the answer?

B Yeah! [Laughs]

R Where do you primarily work on your programs and assignments?

B Mostly at home.

R Okay. Talk about that.

B … when I have time by myself, I will sit by the computer and type it out. I don't like writing; I like typing, the clicking noise. So, I do my algorithms and everything at home.

R … does anyone work with you on the programs and assignments? In other words, do you have like a study partner?

B [Shakes her head no] No.

R Describe your working environment – your ideal working environment.

B Quiet… no distractions, and… that's really the two most important things. …when I have distractions, when I'm really concentrating and something distracts me, I get really irritated.

R Do you ask questions in your computer classes when you do not understand, or do you just try to figure it out on your own?

B I try to figure it out on my own.

R Okay, and what do you do if that doesn't work?

B I go to the teacher.

R All right. What topic in computer programming gave you the most difficulty?

B Pointers.

R … Okay, but before that?

B … string arrays.

R … What did you do to overcome the difficulty?

B I drew… pictures and stuff, to try to let me understand what actually happened. That really, that really helped.

R Yeah, that's really a good way to do that. Now what I want you to talk about is the following topics, which are presented in Programming I and also in the Basic programming class in terms of difficulty. Did you have difficulty learning them? The first topic that is introduced to the student is the concept of a variable. Did you understand that right away?

B [Nods her head yes] Um hmm.

R How?

B Um, by comparing it to algebra.

R Okay.

B That, it just clicked. You know, there's no difficulty there.

R Okay. What about if-then statements? Did you have any difficulty with those?

B [Shakes her head no]

R Okay, what did you relate that to?

B Um, if something happens, then something else happens. You made it really clear. I had no problem with that.

R Okay. So you related it to, like, everyday life.

B Um hmm. [nods]

R … What about looping structures?

B The same thing applies. … do while a condition is present … and then stops when it's not. I don't know; it's just common sense to me.

R Okay. So that was just something you were already familiar with?

B Yeah.

R It wasn't strange to you.

B [shakes her head no]

R Okay, what about the idea of a subprogram? Did you have difficulty understanding that?

B No.

R Okay, what did you relate that to in your own life?

B I don't know. I guess, food. You know food is … a main idea, and you have like main courses, entrees and different things, and then from the different foods, the entrees, you have… various ingredients … where it came from and stuff, and … like a hierarchy.

R … So you related that to something in your life. Okay, what about the argument passing?

B Um, I didn't have any problem with it. [a little apologetically]

R Okay, that's fine. … what did you relate that to, the parameter or argument passing?

B Um, I didn't really relate it to anything. You know, I just looked at the examples given to me and that's what they did so I followed it.

R Okay.

B But... you know... whatever was passed from the main program, it has to be of the same type and so... I just feel like it is all common sense.

R Okay.

B Nobody should have any problems with it.

R ... It didn't present any difficulty to you at all?

B No.

R I am trying to get you to tell me what you related it to in your own life so that it was easy for you to learn.

B I don't think I related it. I just looked at it.

R And it made sense.

B Yeah.

R What about arrays?

B Arrays was interesting. ... It's really funny because it just all makes sense to me. ... when I looked at it, it just never gave me any problems. So, sometimes, you know, when I have difficulty initially understanding it, I try to compare it. But most of the time it didn't, so arrays is something that just makes sense... At ... first... you allocate that amount of space to it and you use it. It just makes sense.

R ... the next group of questions is about your satisfaction and your persistence in staying in computer science. As the programming courses progressed, did you think that the content became more difficult?

B Yes.

R How did this affect your self-confidence?

B It felt good that I could do something that some guys… dropped out of.

R Right. How did the content difficulty make you determined to succeed?

B Well, [long pause] it didn't.

R Okay. What did it do then?

B It just made me tired. [giggles]

R Because you're one of the females that has persisted through the computer science courses, …tell me, when the content became difficult, why you stuck it out.

B Because of Zach. [her son]

R Okay. Talk about that.

B I want to graduate. You know, my dad has been very forceful in this. And… I don't want to waste any time, so… I feel like even if it gets really hard, I just have to stick in there and try to fight my way through it.

R Why?

B So when I graduate, and hopefully it will be soon, with my bachelor's, then… Zach will have a good life.

R Okay. As the programming courses became more complex and difficult, what steps did you take to understand the ideas and concepts that were presented?

B Um, mostly drawing my way through it, and sometimes just laying in bed thinking about it. And…talking to my husband about it, even though he don't understand and [gives a blank look] Yeah, just thinking and drawing by myself helps.

R And talking it out sometimes.

B Yeah.

R ... Why did you originally choose computer science for your major?

B Um, because I like to type. ... Oh, because of my cousin. He, she graduated with MIS and I misinterpreted, I thought it was computer science, so I took computer science, and she told me it was easy. But I took computer science, and I liked programming, I liked Basic programming, Fortran, so I decided to stick with it.

R ... What are your reasons for staying with computer science as your major?

B I like it. I don't like to admit it but it is fun and it is interesting. It's just the down side, the very difficult problems we get that kind of makes me to cringe.

R What are your future goals?

B To make lots of money! [giggles] And, um, maybe do something that nobody else has done before – be a Bill Gates person. ... Maybe, hopefully, do something.

R Any particular area in computer science?

B No, I've not really given it much thought, but I want to do something that will benefit everybody, and not require lots of money. [giggles]

R What do you like most about computer programming?

B Being in control. You decide what happens in your programs, what you want it to look like... basically everything. You are in control of everything that you do and when you want to do it, and... how fast you want to progress with it. And the thing I like about it the most is being by myself... concentrating. It is just like a little hole you stick yourself in, and I like that I guess.

R What do you like least about computer programming?

B The math.

R Talk about that.

B ...I did not like math when I was little. It just kind of hurts me that I have to take all this math. But... sometimes it's interesting when... I go, whoa, I didn't know it does things like that! But other times it's just... a lot of work. I guess I just don't like the amount of thought that has to go into it. But sometimes, I guess I'm kind of contradictory with myself. Sometimes I like it, but other times I'm like, you know, I don't really want to be doing this, and I hate it. But, other times I like it. I don't know.

R What level of math are you in right now?

B Um, I finished Cal II.

R Okay.

B You know, I have to do the rest at UCCS.

R UCCS?

B University of Colorado at Colorado Springs.

R That's right. All right, if you could change one thing about the programming courses, what would it be?

B The introduction of pointers. It should be... introduced a lot earlier instead of being just pushed onto us like that. It shocked a lot of us. And almost half the class dropped after the first pointers assignment. It was just a shock, and you can't do that to people. [giggles]

R That's tough. It's a tough topic.

B I think the concept should be introduced a lot earlier, about... what it means, because it took me forever.

R Where would you introduce that topic?

B In QBasic I would do it, because ... even though [the instructor] was talking about ... the people in charge, trying to argue about weeding people out sooner. But... like she said, even in database you should understand it and when you're working with a computer, you have to understand why your program is not taking... enough stuff, like it's supposed to. And, it's just important, even if you're not in programming... even if you don't want to major in programming, that you understand a little bit about pointers.

R ... what would you suggest to encourage more females to enroll in computer programming courses and possibly consider computer science for a major?

B Maybe having an all-female course.

R Okay. Why would that help do you think?

B Because they'll be more open about voicing their difficulties and... females bond a lot easier and quicker, and when they have difficulties like that, they will ask the person next to them or even they'll do a huddle, like... a group thing. And that helps solve problems a lot faster. And when they have people they know, they will stick in a program. They'll really stick in. [giggles] But if... all their friends drop out of it... they don't want to be by themselves. I'm different because...I'm with guys all the time.

R And that doesn't bother you at all?

B No. [shakes her head no]

R Why not?

B Because I guess I was a tomboy, and I've always had male friends and my female friends were the ones that were limited. So, I'm used to guys. I don't... really care what they say around me, and... I can ask them things. It's just I don't like to be heard a lot. Guys don't bother me. I just don't want to... portray the wrong image.

R ... you're one of the few females that has stuck it out through three, actually four, programming courses. ... what is it about you that is different from those females who don't do that, who don't last, who drop out?

B I don't know. ... I guess because I'm really stubborn, and I would be somebody who would fight... like I would argue my way out of something, or try to. And, um, most females are really submissive and, you know, they don't like to project ... Oh! This kind of contradicts something I said earlier about not wanting to be different, but I... I'm not afraid of standing up for what I believe... And, I don't know, like in computer science where it is all males, you have to be strong you know, you have to... just be yourself and be strong. But most females can't take that. They don't like, I don't know what it is that they don't like. But I find that most females go into accounting where all the other females are and... they go into roles that ... require domination. You know in accounting... there is somebody presiding over you, but ... in programming, it's almost like you're your own boss even though it's really not. And, I guess that's it.

R Do you think that your family expectations might have something to do with it, being that you were raised in an Oriental culture? Do you think that might have something to do with it?

B About being submissive?

R No, about persisting because your mom and dad expect you to do it.

B ... Orientals have that idea, but I'm not really like that. I think I'm in it because I feel that I owe Zach. I want him to have more than I did, and ... that really has a lot to do with it because a couple of times I felt like quitting, and my husband, you know, kind of tried to talk me out of it, but I sit down by myself and I think about Zach. That's really what helped me stick through it.

R So, your child is the main reason.

B Yeah.

R Okay. Can you think of anything else that you would like to talk about? About females in computer science? Anything you have observed?

B Females in computer science don't talk to each other. That's why they don't stick together.

R Hmm. That's interesting. Do you want to talk about that for a little bit?

B Well, [laughs] I find [giggles] ... Well, maybe it's because I'm taking night classes, but most of the females that are a lot older, they don't really talk, you know, to the ones that know it... Um, they stick with maybe one of their friends who don't understand it just like they don't, and they don't get help. So...they meet that hurdle and they can't cross it, so they dropped out.

R Right. What would you suggest to help them?

B Group study, even though I don't like it. But for, I think it should be only implemented for people who has problems, but females don't like admitting they don't have problems. They don't go to teachers. So maybe when... the teacher sees... a dropping grade... there is difficulty in their programs, maybe they should assign somebody who is doing good, another female who is good, to that person. Because, you know, males, they dominate over what they suggest... they just do it for you. They don't let you think.

R Ah. So, study buddies might be a good suggestion. Okay. Anything else you want to add?

B Um, not really. But, uh, yeah! I think there should be more labs than...lecture, because labs really are the ones that help people understand. And... lectures kind of help, notes kind of help, but hands-on helps more.

R What is it about hands-on that helps?

B You learn your mistakes. You see what really happens. You know, you listen and you write it and you know that it's supposed to happen, but when you do the programs, you know, and you run, and it crashes, and you're like, Oh my gosh! What happened? And then you debug it and you understand ... what it is you're doing wrong. And when you do something hands-on, it stays in your memory a lot faster and a lot harder.

R ... I sure thank you for being a part of my study.

[End of Interview]

APPENDIX G

Interview with Carol (Excerpts)

R – Researcher C – Carol

R ... Of all your schooling elementary through college, who has been your favorite teacher or instructor and why?

C I don't really have one.

R Were there any instructors you particularly liked?

C Yes.

R Which one?

C ... basically, my English teachers.

R Ok. Was that in high school or middle school?

C It was... I guess high school and college.

R Ok. What did you like about them?

C They was just very nice and they helped me very much when I needed help. And stuff like that and they helped me a lot with my papers.

R ... Of all your high school or college classes which type of learning activities do you prefer? Do you know what I mean by learning activities?

C No.

R Oh, like listening to the teacher talk, reading something, hands-on, that kind of thing.

C ... basically hands-on.

R Ok. Why did why do you like that the best?

C … because it seems like you learn more than just listening or reading. Because if you make mistakes you can learn how to fix them as you go.

R … did you have any classes that were hands-on in high school?

C … only in my computer classes. The rest was just listening.

R And taking notes.

C Yeah.

R … From which learning activity do you think you have learned the most?

C Hands on.

R … In computer science I am sure you noticed that most of the classes are composed of male students?

C Um.

R … Did the fact that the majority of your classmates were male… affect your learning or understanding of the material?

C Sometimes it seemed liked it did.

R Ok, how? Talk about that.

C I don't know. It just seemed like most of the guys did group together, like the women were by themselves.

R So that affected, um, your learning and your understanding. In what way did it affect that?

C Well, it seemed like when you had a problem, it seemed like, [pause] it seemed like it was harder to talk to the guys because they seem like everything was so, I don't know, complicated.

R Oh, so it was easier to talk to the girls.

C Easier to talk to the girls. Yeah.

R With mostly males in the class did you feel intimidated?

C No.

R No. Did you feel hesitant to ask questions?

C No.

R Ok, talk about that. Why did you not feel intimidated or afraid to ask questions?

C Because I felt like the guys probably had questions, and what I had to ask probably, they probably needed help on too.

R Um hum.

C Or something like that.

R So that never stopped you from …asking questions?

C No.

R Did it make you feel uncomfortable with mostly males in the class?

C No.

R How come?

C It just didn't you know. It was like, they was just in there but it didn't make a difference.

R Now we are going to talk about content difficulty. How much previous computer use did you have prior to your first college-programming course?

C I had programming classes in high school for about three years.

R Ok, talk about that. Tell me what you did.

C It was just like basic stuff. Like working with… simple programming and stuff.

R It was programming for three years?

C Uh no, it was like...intro. But you had to figure out how to get it to do what you wanted to do; write a little program and stuff.

R Oh, that's good; and you did that for three years?

C Uh hum. [Nods her head]

R All right. Do you remember the Basic programming course?

C Yes.

R Okay. How would you rate that Basic programming course in terms of difficulty on a scale from one too five with five being the most difficult and one being the easiest?

C A three.

R A three. Okay. Talk to me about why you'd rate it a three.

C Because it was hard. As I got into the... upper ones, it seemed really easy.

R Looking back on it...how would you rate the Programming I course, the one you just got finished with, in terms of difficulty? Same scale.

C A four.

R Okay. Talk to me about that.

C Because I had a hard time with functions, and I didn't really understand what went in it.

R Okay, so that made it real difficult.

C Yeah.

R Where do you go for help when you need it?

C The instructor.

R … Do you ever ask your classmates?

C I ask sometimes.

R All right. Males or females?

C Females.

R Females. Do you ever ask the male classmates for help?

C I maybe had done once or twice.

R Um hmm, and how did they respond?

C Like, like I should have known it.

R How did that make you feel?

C Kind of stupid, I guess.

R … Now we are going to talk about some social interaction questions. How would you describe yourself socially? … do you make a lot of friends easily, or are you a loner, or somewhere in between? Talk to me about that.

C I guess I am somewhere in between.

R Okay. Why do you say that?

C Because I don't really talk to a lot of people. And sometimes I just talk to certain people. So I guess I am in between.

R Okay. Do you hang out with a group of friends here at the school?

C No.

R No. At home?

C No.

R … Where do you primarily work on your programs and assignments?

C At school.

R ... In the lab?

C Un huh. [Nods her head]

R Ok. Do you have a computer at home?

C Yes.

R But you prefer to work at school.

C Yes.

R Okay, talk to me about why you do that.

C I guess because at school I know I am going do it. At home I just put it off.

R Okay. Who works with you on the programs and assignments? Do you have like a study partner or a programming partner?

C No.

R No, so nobody works with you?

C No.

R Ok. Do you prefer it that way?

C Sometimes

R Ok. Talk to me about that.

C It just seems like, more complicated to me when somebody else is working with me.

R Okay, more complicated. Well, how would it be more complicated?

C Um, it just seems like the way they want to do it, I don't want to do it that way. It's like I don't want to be with nobody. I don't want any work with nobody.

R Their way would be different than your way?

C Yeah.

R Yea, that happens. All right describe your working environment when you work on programs and assignments.

C Describe it in what way?

R … when you are in the lab, where do you sit? Do you have people around you? Do you want to be off in a corner, or does it matter? Just talk to me about your working environment.

C It doesn't. Sometimes I just want to be like off in a corner.

R Ok, so you can work really in both kinds of environments.

C Yes.

R Does the noise in the lab bother you?

C No.

R That's great. You don't have to have it like totally quiet.

C No.

R That's great. Do you ask questions in your computer class when you don't understand or do you just try to figure it out on your own?

C Basically I try to figure it out in my own

R Ok. Why do you do that?

C It seems like I learn more if I figure it out by myself.

R Do you work on the computer when you are trying to learn it or try different things?

C No.

R Then what do you do?

C I like, either read, or just try to figure out what will happen if I do certain things differently.

R Okay. Do you do that with pencil and paper?

C Ah ha. [nods]

R All right. Now we are going to talk about content. What topic in computer science or computer programming gave you the most difficulty?

C Functions.

R ... Ok, talk to me about functions.

C I just didn't get like you are supposed to pass certain things in to get certain things out. That was hard for me.

R ... What did you do to try to overcome the difficulty?

C Just worked at it. That's basically it.

R What kinds of things did you do when you were working at it?

C Just tried to get where it would work, you know. Like if I passed something in that did not work I'd try something else.

R ... Now I want you to talk to me about the following things in computer science. When we teach computer science there are certain key concepts that have to be learned, and so I want you to talk to me about each of these. When you get into your first programming class, the first concept that we teach is the idea of a variable. Did you have difficulty learning the idea of a variable?

C No.

R Why not?

C I just didn't for some reason.

R Ok. Did it make sense to you?

C Yes.

R Did you relate it to anything in your everyday ordinary life?

C No.

R How about did you relate it to algebra?

C Yes.

R Ok, talk to me about that.

C Because you was trying to get … it's like you were trying to solve for x and you, um, do the calculations to get x. That's the way I thought about it.

R So that variable was not difficult for you to learn?

C No.

R Ok, that was the first one. Now the second one presented to the students are if-then statements. Did you have trouble with the if-then statements?

C No.

R Ok. Why not?

C Because it's like saying, "Ok, if this doesn't work then you can do this. So that's the way I got that."

R Ok, so you related that to your everyday life.

C Um, yes.

R Did you relate it to anything in math?

C Not really.

R Ok. What about looping structures? Did that present difficulty when you were learning it?

C Yes.

R Ok, talk to me about that.

C Because it was hard to figure out like how many times the loop should occur. So that was a problem for me.

R Ok, what did you do to try to learn that?

C I don't know. I don't remember what I did.

R Did it all of a sudden make sense to you?

C No. It just, I guess, it was just something that I got used to. In trying to calculate how many times the loop should occur.

R Ok, so that was the difficulty - figuring out how many times?

C Yes.

R Did you ever have difficulty figuring out when it would stop?

C Yes.

R ... Now, what about sub programs? Just the sub programs. We're not talking about passing the parameters - just the sub program idea.

C That made ... that made sense to me.

R Ok, talk to me about why it made sense, or how it made sense.

C It made sense because you could break the program down into different parts. Like the introduction and then the calculation part. That made sense. But the other part ...that part didn't make sense.

R Ok. When you were learning about sub programs, did you relate that to anything in your ordinary life to help you learn it?

C No.

R Ok, you didn't have to?

C No. It just seemed to make sense the first time it was introduced.

R Ok. I am trying to figure out why it made sense to you.

C It just made sense. It's like ... kind of like when you are writing a research paper or something, and you break it down like different parts. That's how it made sense to me. But the functions part didn't.

R ... What about arrays?...Did you have difficulty learning the idea of an array?

C No.

R Okay. Did it make sense to you?

C Yea.

R Okay. Talk to me about how you learned arrays, or any difficulty you had with it.

C It's just like making sure the numbers were in the right places. ... making sure they are ... in the right order that made sense.

R ... What was the hardest part of learning arrays? Do you remember?

C Um, gosh, I can't think of a part that was complicated.

R ... Arrays just made sense to you then.

C Yeah.

R And you were able to just put it into your knowledge. I was going to say assimilate but I didn't know if you'd understand that. Ok, arrays you had no difficulty with.

C Yes.

R Looping structures, variables, pretty much all of that was okay?

C Yea.

R All right. Now talk to me about the parameter or the argument passing.

C That part confused me.

R Okay. Talk to me about it. How did that confuse you? Why did it confuse you? What's confusing about it?

C It's like... when you first have the variable and the parameters, and you are not supposed to use the same ones when you are, um, doing the functions. Calling it. That part confused me and you have to have like those little "and" marks in there. That was what confused me.

R Oh, the notations.

C Yeah.

R Okay. Did you have trouble, um, with linking up the names in the main program and in the functions?

C Yes.

R Okay, talk to me about that.

C It's like when I had the name of a function, I always, for some reason, I didn't make it the same as the one earlier. I got a lot of errors and I couldn't work it.

R ...Oh, okay. As your programming courses progressed did you think the content became more difficult?

C Yes.

R ... Did this affect your self-confidence at all?

C In a way.

R How?

C I wanted to get out of computer science.

R Ok, what happened to make you think that?

C It was just there were certain things that I just couldn't figure out.

R Ok. Such as?

C Um, functions and, um, like certain loopings and stuff. That was hard for me to get it. So I just felt like I wanted to get out.

R ... Well, I'm sure you have noticed that as you go along in computer science the first or the beginning course has a good amount of females. I wouldn't say a lot, but there's some. Then in the next course it gets really slim. Have you noticed that?

C Uh hmm. [nods]

R Okay. Um, in computer science, when you hit these difficult spots does it make you want to succeed? Does it make you really determined to get through or does it make you want to throw in the towel and say forget it?

C Sometimes I just want to forget it. But then I figure that I have done all this, so I might as well just go on through the rest.

R When you learn something that is difficult or you manage to do a difficult program, what does that do for your self-confidence?

C It makes me feel like I can do the rest of the program, because on the rest of the programs, you are going to have to do some of the stuff that you did on the previous programs.

R Does that help you want to finish it? Finish the course I mean?

C Yeah.

R Okay. As your programming courses became more complex and difficult, what steps did you take or what did you do to try to understand the concepts and ideas? What things did you do as it became more difficult and more complex? What things did you do to help yourself understand?

C I guess work harder at it.

R Ok, in what way? What did that mean?

C Um, read more, um, just work at it harder.

R Okay, by work at it harder, um, that means reading the textbook?

C Yes.

R Ok, what else?

C Um, like going back over my notes, and trying to figure out what I did wrong.

R In your programs?

C Yes.

R Ok. What about, um, handouts?

C Sometimes they help and sometimes they don't because sometimes they make me confused.

R I understand. All right, why did you originally choose computer science for your major?

C Because I was very interested in computers. I just wanted to see how certain things worked. So that's why.

R Okay. Have you decided to stay with computer science?

C No.

R Ok. What have you decided to change to? What is your major now?

C It's still computer science, but I am not sure what or if I am going to continue with it.

R Ok, what are your future goals?

C I don't have any right now. I guess just to get through with computer science.

R ... What are you going to do when you get done with computer science?

C I don't know. My mind is just occupied with computer science so I can get through.

R Ok, all right. What courses have you taken in computer science?

C Um, Basic, Cobol and Programming I.

R Ok, what do you like most about computer programming?

C When the programs are done.

R ... And what do you like least about programming?

C Just figuring out how the program should run. That's the like difficult part because you have to have to make it look a certain way. That's the hardest part. That's what I hate.

R ... By making it look a certain way, what do you mean by that?

C That's like when you get ... like when you get a handout in programming, and this is what the output should be, and just getting the correct numbers, and getting them in the right places.

R Okay. So you are talking about the format of what the output looks like, as well as getting the correct answers.

C Yes.

R Okay. All right, that makes sense. If you could change one thing about the programming courses, what would it be?

C The idea of functions.

R What would you do with the idea of functions?

C Delete them.

R Okay. Now, you've noticed that as you go along in computer science, the number of females really decreases. [She nods] Do you remember how many females there were in the Basic class you were in?

C Um, I would say about three or four.

R Okay, and then when you went into Programming I, how many were in there?

C Two.

R ... What thoughts do you have as to why females get out of computer science?

C Um, most of the girls I know got out of it because it was too complicated to them, and they felt that they had a hard time with Basic so they were going to have a hard time with everything else. So they just got out.

R Okay. What made you different? Because you stayed in.

C I don't know. I thought it was like ... I don't know why I stayed in really.

R ... Do you like it?

C Programming?

R [researcher nods]

C No. I don't know. I just have a hard time, like, uh, when I do something I just have a hard time... not finishing it.

R … What would you suggest to encourage more females to enroll in computer science?

C It will be more money, and they'll learn to like it, and I don't really know. It's hard for me to say that.

R Because you don't really like it.

C It's ok. It's ok, but it just gets hard. Plus all the classes you have to take with programming is hard. But I don't know.

R If someone were to ask you… what are some of the reasons why you chose computer science, why you chose to stay in it, what would you tell them?

C I don't know why to tell you the truth. Gosh, I don't know. I just didn't like the idea of giving up.

R Giving up. So you have a high persistence?

C Not really. Not really.

R So do you have any thoughts or any ideas on why you think females don't go into computer science?

C A lot… of females I know, they don't like computers.

R Okay, why do you think they don't like computers?

C I don't know. Like, I know some think they are too complicated, …so they just don't want to go around them.

R … How much math have you had?

C Just College Algebra.

R … Did the math help you in the computer programming?

C No.

R … Ok, can you think of anything else that you'd like to tell me about why you think women are under represented? In other words why there aren't so many women in computer science? What are your thoughts on the matter?

C I just think a lot of them think it's too hard, and too complicated, so they don't want to go through it, just like there's not a lot of women who are… in math. So they figure this is just the same thing. They just don't go in computer science.

R Ok. Would you have any encouraging words for them to go into computer science?

C No.

R … Well, thank you very much.

[End of interview]

APPENDIX H

Interview with Diane (Excerpts)

R – Researcher D – Diane

R ...Of all your schooling, elementary through college. Who has been your favorite teacher or instructor?

D Uh, goodness, uh the first name that comes to mind is, uh there are three. Of course, Miss Bailey. She was the gifted instructor for special studies at Central Elementary School in Gulfport. And, well ...she treated us all with respect. And uh, she really stood out among the other teachers in that she allowed us free time, me time, She gave us mind benders, things that tested our computational and analytical skills. ... but then there were Miss Thornhill and Miss Petry. Miss Petry was the continuation of the gifted program at the junior high level. [and then there]...was another teacher, Miss Thornhill who ...was a person who was just, um, very positive and optimistic soul. She always would really get me into heated discussions with her in class, because I would feel comfortable with her and speak my mind, and so we would get into topics about the theory of the open universe versus the closed universe. We actually went into that once about thermals in the earth's core and theories on what's really down there. She was just, um, there were three women that were very much a part of my life. And then her brother, Mr. Prior. Well ... he really pushed me in Algebra and sent me along into uh, a couple regional contests, um where I didn't put a lot of effort in, but I was just glad to have been chosen.

R ... Of all your high school or college classes, which type of learning activities do you prefer?

D ... I like hands-on, you know, tactical skills or anything taught through lecture or experience, seeing it, um, but I can kinda understand the way different classes require different ways. Sociology, uh lecture, you read, you listen, and then you go out and observe in real life but I like in my computer and my science classes and in my math, you know, you learn it and you do it. That's it, I just like to learn it and then do it And then be given critiques on that to tell me how I can improve upon it.

R From which ones of those learning activities do you think you've learned the most?

D ... I have to say a mixture of the two. ...

R So that's a combination...

D A combination of all the learning experiences that I have been against.

R The lecture and then the hands-on type of thing? Is that what you're trying to say?

D Yes.

R Okay. All right. Um, your class did not have any males in it, so I'm going to skip these two questions, unless, did you have any computer classes at ...Tulane?

D ... I did.

R All right. Then, let me ask this. Um, was the majority of your classmates male?

D You know, I think so.

R Did that fact affect your learning or understanding of the material?

D Not at all ...

R Okay. With mostly males in the class, did you feel intimidated?

D Never.

R Did you feel hesitant to ask questions?

D Not at all. I knew that there were some things I just did not understand. In fact, the only thing that would ever keep me from asking questions is not knowing what question to ask.

R All right. Tell me about the courses that you had at Tulane, the computer courses. What did you have there?

D ... I think it was just the one. It was Fortranning, uh, computer programming in Fortran. It was a lecture - three days a week. It was early in the morning. It was, um, let's see, composed of probably thirty-five to forty-five students, all on tables with a raised platform and a podium for the teacher from which she spoke where she could also utilize the blackboards, which spanned the entire wall door to door. It was just, uh, you know, bring the book, look through it, familiarize yourself with the material beforehand, and see it in class. There was always an assignment. This assignment was due by this time and we would learn those, uh, the modules that would help us with that assignment as the days went on in time to complete it and turn it in by the main deadline. Now, the difference with that class and the one I have just taken was that it was strictly lecture. We were expected to use our free time, our outside-of-class time to do all of the work that was required for the, um, anything but testing. ... we were required to... learn and use flow charts with every program. Um, that I found rewarding. It was like... kind of a mini-architecture design. I would go through pages upon pages with the white computer paper till I got it right with my straight edge and pencil. I liked that, ... but, the, the best thing was the...computer labs...you could go to the Open lab where lots of students came from all walks of the school of life, and if you wanted privacy you could go to the one that was mostly for the engineering/computer science students.

R ...how much previous computer use did you have prior to your first college programming course?

D I had the required typing class. ... that's the only computer class that I had taken. ... and though we had a computer at home, I was never allowed to use it. So, um, that was really it. Not much.

R ... the introductory course, the BASIC programming that you just completed, how would you rate that course in difficulty on a scale from 1 to 5, with 5 being most difficult and 1 being the easiest?

D ... I'd just have to give it an average reading of three.

R A three?

D If I had not had previous programming experience... I'm sure I would have given it a four, but with my background in Algebra and Calculus being as strong, and knowing the basics of this... I would say a two or a three in my case.

R Okay. How would you rate the Fortran course that you took at Tulane in terms of difficulty on that same scale from 1 to 5?

D It was hard. Um, four and a half.

R Okay.

D Yes. I, um, spent many hours, countless hours. I didn't understand... See, I would give a first time go at Physics a 7 on that scale. In this case it would still be a 4 ½.

R ... where do you go for help in computer science when you need it?

D ... I talk to my cat, it, it relaxes me, and, then that gets me to the point that I can usually figure this out. Then I hit the books, and then after that point I go directly to the instructor, um, because I'm at a level here where I don't feel like, um, the others in the class have had much experience. They're probably around the same level as I am, maybe with a few more classes, but, um, just in general observation from ... the program that was first due, it seemed like the other students had some outstanding abilities that I could... really benefit from more than just being friends and maybe learning it together... But I do in the future plan to collaborate a lot more with my colleagues, my peers in the classes and in the future at work because I feel that that is a very integral part of this.

R That's right. These are some questions on social interaction. How would you describe yourself socially?

D Gosh. I don't know ...

R Do you understand the question?

D I really don't understand the question.

R I was really... trying to get at... you a social butterfly or do you tend to be a loner...or do you talk a lot?

D All of the above. ... I've been quite the loner for a few years. I've found that it's gotten me through some tough times, but I feel comfortable in most environments, that's safe environments. Once I've established that whole idea in my mind then I just go with the flow.

R Where do you primarily work on your programs and assignments?

D At home for the first 50 percent of the assignment, then I come into the lab for the next 30 percent, and then another 10 and 10. And then once it's done, it's back home again for a double check. It's a mixture of where I need to be.

R Does anyone work with you on the programs and assignments? In other words, do you have a study partner?

D I haven't found a study buddy. I tried that with a girl who, uh she asked my help actually. She seemed like she was having a rough time, but then again, she had a problem with, it seemed with my enthusiasm at one point.

R When you work do you prefer to work alone or do you prefer to work in the lab with other people around?

D I'd have to say alone, because I'm more comfortable with my headphones on, but I don't mind other people around. It's just, um, we're all here together doing our thing, and then I'm more than happy to answer any questions, or, um, it's helpful to know I can get up and ask them, but I actually prefer to work alone because I have a very peaceful environment in the home. I know that I can get much done and it won't bother me if I don't finish.

R ... do you ask questions in your computer classes when you do not understand or do you just try to figure it out on your own?

D I ask questions Um, I think that it's important to, to try and get it first and then for everyone's benefit there are always questions and there may not always be correct answers, but it's important to pose possibilities I think...

R ... All right, now we're going to talk about the content of the computer course. What topics in computer programming gave you the most difficulty?

D ... I think that it is a science-based subject, which involves math in such a way that it's hierarchical, I think. You start with the basics, and you build and build. So I think the last thing we learned, arrays, come to mind. Because that just happens to be the last thing, the last chapter, that we studied. I'm sure that the next thing we get into will be the most difficult thing I've ever encountered.

R ... I want you to talk about particular topics that I'm going to name in terms of difficulty. When we teach computer science, the first topic that is difficult for some students is the concept of a variable. Did you find that difficult?

D No, I've been dealing with variables for 13,I think um, almost a decade of that. It's really set the idea in my head. There was nothing new. It was nice to be able to compound doing it in a new field. No, not difficult.

R ... you were already familiar with variables, in what way?

D ... I learned about them in Algebra, Pre-algebra. And then I went on, and you use them all through out. I'm in Calculus now, but also I used them to figure out my budget. If I need so many gallons of milk, a gallon of milk will last in my refrigerator about 8 days. So, you know, usually until just the day before it says it's going to go bad, it's gone by that day. So, if I need 3 gallons of milk per month, you know it's 3 times X. X is so much, you know, I use that mostly in budgeting. Um, also after having taken a nutrition course, I use it in trying to figure out my daily intake so that I can get an idea of how I'm doing. I've been doing real good on that lately. So it helps you just, um, I use them every day; it's like second nature.

R Okay. What about IF/THEN statements? Was that difficult for you to learn?

D No, that's just logic. If, you know, you tell your mom you're gonna be home by 6 and you're not, then you'd better expect to be asked for an explanation. ... It's a necessary evil. If/Then.

R So, you were already familiar with it because you used it in everyday life then?

D I guess so. It's something that we've made up and it helps. It creates some conditions. I've learned that, I've known I for a lot longer than I've realized it. Because I've used it in everyday life.

R Um hum. What about looping structures? When you got to looping structures, were you able to relate that to anything in your life?

D Ritually. Though, I also just learned that sociologists say that this is some kind of a non-conformist attitude, but I've just seen it some where you just do it over and over again. But sometimes that's necessary, if you're doing a long multiplication with many numbers on paper, it's the same process and, um, I find it very, very convenient and efficient and wonderful at times that I can just explain it one time to the piece of machinery and this wonderful little microchip processor. And it will do that...that's something that... again, I actually found no difficulty. I was pleased, quite pleased actually to become acquainted with the actual, what's that word? Um, when you get an error, and it's a grammar error...

R Syntax?

D Syntax, yes. To learn that, it really pulled things together. In the beginning you're thinking, I have to do so many things and look at what these people have done. How have they done it? How am I ever going to do it? And then each time you learn something new, it can be hard, but once you get it, you are really glad you know it.

R ... So you didn't have any difficulty with learning looping structures then?

D No. Am I being too elaborate?

R No, this is exactly what I want. And this is a feeling thing. There are no right or wrong answers. I want to know how you feel about all of this. The next one is subprograms. Did you have any difficulty learning subprograms? Subroutines?

D No, I'm sorry...

R Like functions and stuff like that.

D No, um, I'm sorry. ... Well, I think it's subprograms like, um, cleaning house it's like, you know, I know I have to do all these things. I have to clean the kitchen, but that involves doing the dishes, wiping the counters, sweeping the floor, and, you know, dusting and mopping and, you know, maybe even once in awhile cleaning out that gunk that came of the bottom of the chicken package on the bottom of the refrigerator. Um, you know, and that's just one part of the house. I have to do the whole thing, so, you know, subprograms are just like, um just like that. ... I thought they were terribly easy just to be able to say, service, um, call the, call the dishwasher. ... I enjoyed subprograms

R What about the argument passing? Did you have difficulty with that part of it?

D I didn't really give it too much, too much thought, um, I had no difficulty, you know, reasoning it. Um, in fact ... I did understand what I was exposed to the actual, um, the, the use of it. The program we did on stats and I believe another one or two programs before that... it's like, um, like passing a, a note .... I knew that where you have to give the note to this person and they would do that but if you don't put exactly what you need on the note and then you give it to this person and they have to... rewrite the note... revise it somehow... to make it more... understandable to the next person who's going to get it. You have to include the pertinent information ... in this... stats program ... I saw that I only needed to pass what was important, ...when I got ready to think back I had to make sure that these, these are the important things no more or no less keep it simple and we'll all do fine.

R ... the last thing is the, the last subject that we studied, um was the array. Did you have trouble learning arrays?

D ... I don't really know what I think about them except that they help. You know I wouldn't want to list. I don't like writing recipes down, I... I wouldn't want to sit there and write every name and every grade and every time, you know, all in one go at it, you know. If it was just a planner with all the students names and then the grade when they take the test but when you convert that to digital information it becomes hideous to have to redo it, and you begin to wonder why, why we are doing this, how efficient is this that I have to do what I have exactly done times two. ... So it, it helps ...

R So when you were learning the new concepts did you try to relate them to something you already knew?

D I guess so... Yeah I did that. The only thing I know is what I have lived. I know about the path that I have walked down. Um, some I know as much as the back of my hand. ... it's really difficult to know too many things. You have to relate those things to other things in your life, and ... and it's important to have someone to talk to. Of course, like a study buddy.

R Yes.

D I feel like I need an interpreter at times but I... forgot the question again.

R We were just talking about relating what you were learning to the things that you are already familiar with to make them easier to learn.

D At this point in my life, I am learning so many new things, and I'm learning, I'm relearning old things in a way that I never imagined I would ever... and... it's become absolutely necessary ... that I'd take every aspect of my life and integrate them in a way that I can handle them. That I'm not, not a different person when I'm sitting at home doing the dishes than I am in the next room doing some work or in a totally different environment, so that it doesn't become a totally different environment when I step out of the door of where I have found myself recently.

R ... now we're going to talk about some persistent questions, maybe some satisfaction questions with the computer science classes. As the programming course progressed, did you think that the content became more difficult?

D ... I didn't find it really more difficult... but I didn't really find it more difficult, but I just found it to be the next logical step, and that was pretty much the extent of what I saw.

R ... as the course progressed and the topics changed, did this affect your self-confidence in any way?

D Now, that's... quite a question. ... I find that I will not do something that will affect my self-confidence negatively. Positively, now that's just something that I strive for daily. I can't really say...I did though, I got the satisfaction of knowing I completed something that I had put time and effort into that, that I had learned something new, and something that I knew would bring me to something else that would bring even more satisfaction. Confidence, uh, it was confidence, right?

R Uh huh.

D I don't know much about that, except that it's very necessary.

R I think that the question that I'm trying to ask is, as the content became more difficult, and you accomplished each step as you climbed the programming ladder, did your self-confidence increase along with it?

D Oh, yeah, in that case, I guess it did.

R Um, how did the content difficulty make you determined to succeed?

D Hmm, I don't think I understand the question.

R Some people when they get to difficult topics, one of two things will happen. They'll either quit or they'll be determined that they will succeed.

D Oh, ... I'm determined. I spent, ... long hours, long nights, not sleeping. ... I did probably lose... ten to fifteen minutes, winks, each night ... that I had something I just wasn't getting. Um, that loop, that um, printing loop, ... that was the last thing that we had to do and it was... not even required, but it was something that once brought to my attention could be done... I was determined to do that, and ...I did, you know, and I felt good about that. But I am not the type to quit or stop. Um, I've been posed that option too many times – that's become bland.

R ... as the programming course became more ... difficult, what steps did you take to understand the concepts and ideas that were presented?

D What steps did I take? In order to understand the ideas that were presented?

R Yeah. As it became more complex, or did you take any steps?

D ... I think, I continued in a way that, um, I knew would benefit me.

R Um hum.

D Of course... no, it had to benefit me if I am going to benefit anyone else that would help in all other areas. ... I just did what, um, was necessary and maybe a little bit more. Um, taking steps, um, I would sit back and look at what it was that ... I had to do for what has to be done. What ... was it that I needed to know in order to complete the objective, both in application and, and, examination. Then there was always that when there was going to be more after this, that I'd better go ahead and prepare a little while I'm at it. Um, so I did what I could and in that case, but it was not really much, except, in that area right now, except show up to class and if you have the time and the resources to do any outside research, but that involves a lot of time, so buying and borrowing books and reading them and going on-line, and then sorting through all that preparation, with a full schedule you just have all that kind of time. So the steps I took were the necessary ones, and if I just wanted to know a little bit about the topic, I just asked a few questions.

R Do you have a computer at home?

D No.

R All right. Now we're going to talk about some motivation questions. ... we're trying to find out why is it that some females will stick it out and some don't. So, here's the first question on motivation. Why did you originally choose computer science for your major?

D I, uh, had a plan, a few goals that I set in seventh grade in Miss Thornhill's class; she invited, um, an engineer, first a ham-radio operator, and then an engineer. All the guys were interested in ham radios and CB's, or truck driving and sitting there and talking to each other or whatever it is they talk about. ...but, um, when this engineer came, he was a guy, and he said that there were hardly any females. That was really, well, shoot, that floored me. I knew that I had a great interest in math and that I knew that I was good in the scientific method. ...but ... this guy had... said that, and that's when I, it really sparked, ignited, and ... ever since then, I've wanted to be a chemical engineer. And, I've gone through exactly what I've heard everyone goes through sometimes, where you change your major here, and you change it there, but every time, it's still there, my original decision. And I have gotten back to that with a finalized decision with this, um, just being this is what I am going to do. Um, so with the prospect of engineering in the future, I know that its going to be a very, very, very important to know about computers...

R All right. So your major is actually chemical engineering?

D Right.

R ... Tell me about your future goals.

L ... I want to have a family, and I want to be able to provide for myself till that point and for them ... and ... I want to meet and marry someone ... I've had the ambition to get to where I see myself in the future, at a comfortable salary ... I want that, you know. I, maybe I'm selfish, but I want that ability to be able to... buy the things I want to wear... I want to be able to say, well I have options, to get on my bike, I can roller blade, I could save up and every couple of years go skiing, that would be an awful good time.

R What do you like most about computer programming?

D I think it's, um I like it because it helps my mother do her job. ...every day that she goes home she knows that she's helped someone, it's good enough for her. ... That's why I like computer programming. I think it's a way to help other people. I don't see, you know, why I necessarily have to sit in an office, you know, and interact with you all the time when I can do two things at once. I could be there at my computer and doing something on a grand scale, something that can be distributed. And then on a personal level, ... ask someone how their day is going...and not have to worry about filing something.

R … What do you like least about computer programming?

D … I've not really found anything I don't like. … I don't think I have done it long enough not to have.

R … if you could change one thing about the computer programming courses, what would it be?

D … I really wouldn't change anything; I don't feel that I have the kind of authority to even really think about it yet. There's nothing I would, I would do.

R All right. Then, what would you suggest to encourage more females to enroll in computer programming courses and possibly consider it for a major?

D I do think that it would be nice to see more women in it, but, what would I suggest to have them enroll?

R Uh huh.

D Is uh...I would suggest is that mothers ingrain it into their daughters, and fathers show their little girls, hey it can work, really teach them. … if given the opportunity, I would say start young. I wouldn't necessarily … go around rallying "Join computer science! You know, it's a wonderful thing! You know, you would love it!" If I got into a conversation, I would bring it up, and, you know, inquire about someone's mathematical skills. Um, see if they agree with me on how relevant it is in life, you know, uh, where did it come from? Why do we have it? Why would it be taught in every school if it wasn't important? And, um, do you see the way that computers have developed and how they help, and uh, wouldn't you want to be a part of that? But, for a long-term solution, I would say, we have to start at home.

R ... In Tulane, in your Fortran class, did you receive any, ah, denigrating comments from the males...because you were female?

D ... no, I didn't. I'd just go in a pair of dirty jeans and a t-shirt and felt like one of them. Just like one of everybody, and uh, except my t-shirts are cuter. I won't get into some of the things I've seen on the back of men's t-shirts.... I didn't see any hot shots, anybody that, uh, could do everything, you know. I met one fellow over at that console that could do a whole bunch. He wasn't degrading, he was just funny. He said things like, you know, `Every time you break a computer, you learn a little bit more about it. You're not really an expert until you've shut down a computer about a hundred thousand times.' That to me, was very funny, um, but that's what I got out of it. A class full of, you know, regular humor and good folks that were, you know, there for a common goal. Nobody that had an attitude, or scary origin, female or male, you know, you run into them both in other aspects of life...

R Then, if you, if you were asked your opinion on why there are so few females in computer science, what would you say?

D ... first, I would wonder if it was a demographical question, you know, is this in a certain area of the world, of the country? Are there, is this... in general, I don't think that, well, girls don't like other girls. ...You know, I know there's few women left comparatively than men in computer science, what else is there? ... I never really gave that much thought, but I do know there are salary caps, things like that. I, I don't know, it just goes back to the way that we are treated and the way that we treat each other. If you don't find, if you don't find it in yourself, and you don't have someone telling you the importance of learning, and praising you for every small accomplishment, no matter how little the step is, it's still a step. Then you won't go on and computers are a very new thing and can be intimidating. And, uh, so I think that with the male attitude of `Hey, go gettum buddy! You slap them on the butt.' … It's like stop; go. Guys do that, they're just like the "stop and go" group of people. Girls are more hesitant. It seems females aren't as "go gettum" as guys are. You know, unless you're taking your son to a t-ball game. Um, I don't know, maybe I'm just being totally very typical, but, if that's how I view things...

R Could you elaborate about what you said a while ago about women don't like women?

D It just seems like, um, when you're young you find that, um, you really want to be around guys. … I find it's like 50/50. … I've met a lot of girls who really get along real well. They're good girls, but there are some cut-throat people out there, and you find that out early, and it gets worse as you get older. Your roommate is doing terrible things, you know, violating your trust, telling secrets, all of those things. I said girls don't like girls because it happens at a young age. You know, you tell your new best friend a secret and the whole school knows it. When it happens to you a couple of times, from a couple of different people, then you develop a sense of "girls suck." … but then when you get it older, it's also the same thing. You run into guys who make degrading comments. They, um, you know, comment on how… poor you look, physically terrible you look. Then there's the one guy who says, "You know, well, you look sick, or you look better today, or how are you feeling?" ... So it's the same thing, girls find boys icky to a certain point, but, you finally have to just circumvent that whole, the idea that girls don't like girls. You just have to get around that, and then make your way. I don't really know why I got into that...

R Well, I think that's it. ...Thank you.

[End of interview]

APPENDIX I

Interview with Evelyn (Excerpts)

R – Researcher E – Evelyn

R ... Of all your schooling, elementary through college, who has been your favorite teacher or instructor, and why?

E My favorite teacher or instructor was my Latin teacher in high school, Mrs. Horner.

R Okay. Talk about that.

E Um, she was just, she was very encouraging, very demanding. Along with the Latin, we had to learn Greek prefixes and suffixes, and that's helped with all my other classes, cause if I didn't know a word, I could usually tear it apart. So she gave me a good basis for that.

R Great.

E That's why I really liked her.

R Okay. Of all your high school or college classes, which type of learning activities do you prefer?

E Learning activities... You mean like, homework or? [looks puzzled]

R Um, hands-on, reading, writing ...

E Hands on.

R Okay, what type of hands-on activities do like the best?

E Um, like when you're in the classroom and you do the group, like discussion, and things. You hear a lot of different viewpoints, and sometimes it makes you think, so I like that better.

R Okay. What about the projects that you have done with your business courses?

E [grins] I'm not real fond of team projects, because you invariably have one person that doesn't want to do any work, and they want to take advantage of the grade. But, I can see where it's helped in a lot of ways as far as learning how to get along with different types, and get the best out of everybody.

R Okay. From which learning activities, in all your schooling, do you think you've learned the most?

E Although I hated it back then, it had to be the drills that Mrs. Horner would put us through with the Latin and the Greek. That gave me, I think, more to fall back on than anything else.

R What kind of drills were they?

E Um, well, like when we would go into class, we would have not only our little test in Latin and all, but we would have to have a test in the different prefixes and suffixes and root words. Every day.

R Was this like written, or oral?

E It was written.

R Written. Okay. All right. Um, when you were in computer science, did you notice that there were mostly males in the classes?

E Definitely.

R Okay. Did that make you feel intimidated?

E In a way, because you couldn't, you couldn't phrase a question the way that, that they would answer you the way you wanted it. You would have to go at it a couple of times before they finally understood what you were confused about.

R ... Did you feel hesitant to ask questions in class?

E Yes.

R Okay, talk about that.

E Um, the computer science instructors that I had that were male, one of them seemed to read more from the book than to actually know what was going on with the actual hands-on experience. One of them was so bright that when you would ask him a question, you would end up feeling like you were more in the dark than ever before. And the other one, if it was something simple that you weren't quite grasping at that moment, I don't think he meant to, but he would make you feel like you were incredibly slow, or this is something you should have caught on to. So, yeah, I think it did. [nods her head] I was more comfortable asking the females questions.

R Okay. How much previous computer use did you have before you took your first college computer course?

E None.

R Okay. What was your first college computer course?

E Basic, [laughs] which was not basic. [laughs some more] It was the language Basic, but it was so new that it was just billed as "Basic," a computer class for beginners. So, I took it.

R Tell me what computer classes you took.

E I've had Basic, I've had Fortran, I've had the introduction to computers, I've had, um, a little bit of, uh, I've had Pascal, I've had a little bit of C. So, and then playing with it every day, but those are the actual courses.

R All right. If you remember the basic course, how would you rate it in terms of difficulty on a scale from 1 to 5, with 5 being the most difficult and 1 being the easiest?

E It was challenging, um, difficult for some, but I'd say probably a 3 for me, 'cause it was a new concept. But once I got into it, then it was different. It was fun. It was, it wasn't easy, no. There was a lot of work involved in it.

R So after that you took ...

E Fortran.

R Fortran. How would you rate the Fortran course in terms of difficulty, same scale.

E That's the one where I had the instructor that read straight from the book and we basically taught ourselves. So, I would say that was more along a 5 because it was learn as you do.

R Okay, where did you go for help when you needed it?

E Usually to Mr. Meir, who taught Pascal. Also I had Cobol; I think I forgot about that one.

R All right. How would you describe yourself socially? Are you a loner, or a social butterfly, or somewhere in between?

E Somewhere in between. Depends on how comfortable I am with the situation.

R Okay, talk about that.

E Um, I can be a loner, I can get things done on my own, but I do prefer to be with other people, not necessarily a big group, but like one or two, three, something like that.

R Okay. When you were in computer science, and even now with your assignments in school, where do you primarily work on them?

E At home mostly. Either at home or in the library on computer. [laughs] That's where most of them end up going, yeah.

R Does anyone work with you on your assignments?

E Only the team projects that we have to or forced to. But on various papers and all, I do it by myself.

R Okay. When you were in computer programming, did you have like a study partner or a group?

E Yeah, survival in numbers was better. If you had a partner, you did a lot better.

R Okay. So you did have …

E Yeah, because a lot of times the concept you didn't understand they did, and you could kind of bounce off of each other.

R Okay. Do you ask questions in your business classes or your college classes when you do not understand, or do you just try to figure it out on your own?

E Most of the time I'll ask a question.

R Okay. What about in the computer classes with mostly males in the class?

E Until I get to know them really well, like the first two thirds of the semester, I'd probably wait and ask the instructor after class, rather than do it in the class.

R Okay. Why is that?

E 'Cause you're either treated like you're a know-it-all or you're treated like you're a total idiot by the other class members until you get to know them. Once they get to know you as a person, not a female, then you're fine. But otherwise it's a little intimidating, cause they look at you like, [disdain on her face] "uh, can't you shut up. You know, we've only got thirty minutes left. We can get out of here if you'll just shut up." So, at least that is what I've found in the ones I've taken.

R That has come out a lot. It's typical. Okay. What topics in computer science gave you the most difficulty?

E Had to be ... functions in C. I think. That would be what gave me the most difficulty. Passing back and forth.

R The arguments?

E Um-hmm [nods her head yes]

R What did you do to overcome the difficulty?

E Grabbed my instructor and said, "Hey! Look! How does this work again?" [laughs] But the instructor I had for C was really good about showing us how to do it. But I have this, I don't know, mental block about passing something out and back in, and the way to do it. I prefer Fortran to C. Most people prefer C because it's more flexible, but I like Fortran. So, ... [Shrugs her shoulders]

R Okay. One of the things we're investigating is the difficulty of the topics in computer science. And, if you remember, there were more females at the beginning of your classes than there were at the end of the classes, that it was usually the females that dropped out.

E [nods her head] Yeah.

R I want you to just talk to me about some different topics in computer science in terms of how difficult was it for you to learn them, and what did you relate that to that make it easy for you to learn? Okay, so the first one is the concept of a variable.

E That wasn't really difficult because I'd had some biology classes and all, and associating things with other things was not hard for me. That part females seem to get. The males had more trouble with that, I think, than the females did, for some reason. At least I know in the Basic class they did.

R Now that I think about it, that's a good point. I wonder. Do you have any speculations on why that might be true?

E No, other than the fact that I could just visualize it easier. I think. [shrugged her shoulders]

R And you just related it to something that you already knew.

E Well, women are always, like, when you're talking about a car or car parts, the thingamabob. You know what that is when you're talking about it. You already have another name for it. That's a variable name for it. But, see, guys are very … A cup is a cup. A plate is a plate. They wouldn't know the difference between a china dinner plate and just a, a snack thing. They are very literal in some ways. But women are usually more, like you know, I can't quite put the name on it, but you know that thingamabob that does the such and such. And most women will catch on with it. So that one, I didn't have any problems with.

R Okay. Good. What about if-then statements? Did you have any difficulty learning that?

E I didn't. [shaking her head no]

R All right. What did you relate that to?

E There's always a consequence to an action, one way or the other. So for me, that was easy.

R Right. So it wasn't necessarily another discipline, like Biology or math, it was ...

E That came from my mother's days of if you don't do this, then. So, I had my first if-then at a very early age. [laughs] So that didn't... really phase me. Um, I know what did phase me was the NOTs. And it took a friend that had been through, um, electrician's schooling to explain about NOT gates and how that worked. So then when I had a visual to follow through, then I understood it.

R ... What about looping structures?

E ... That one was easy, I guess. I don't know why.

R Did you consciously relate it to anything that you were familiar with?

E Not really.

R Okay. It just wasn't difficult for you to learn.

E The only part was ... Certain loops are post-checked and some are pre-checked. Some of the post-checked ones, if I wasn't thinking my way straight through it, I might get a little confused. But then when I start stepping through it, then I would be all right.

R But it just made sense to you.

E [nods her head yes]

R All right. How about arrays?

E That one screwed me up until the instructor started explaining it as like a … a letter box in the post office. Now once they did that, then I understood it. But I had to actually kind of see it in my head to understand it. And that was one of the harder parts, I think.

R … Now, what about subprograms? Not the argument passing. Just the idea of a subprogram.

E That I thought was cool. The only thing I would get confused on would be the passing parts, you know. What had to be, that you have to have the same variable, that you use a different variable and how it passed back, and that's where I would get a little confused with it.

R Right. But the actual subprogram itself?

E That didn't bother me. I thought it was really neat. It was a way that you could, um, take like maybe one subprogram that would work in one program and move it over to a different program and still continue to use it and save you a lot of time in coding.

R Right. Was that a new concept to you, the idea of a subprogram?

E Yeah. Yeah, it was.

R But it wasn't difficult to learn.

E Not once you figured out how you had to do, how you had to, you know, send things down and bring them back.

R Okay. Now tell me, now talk to me about the passing of the argument.

E Now that's where I get, I still even to this day I get really confused about how to send it. You know, and how to call it.

R All right. Are we talking a particular language like C?

E Yes, C in particular, yeah.

R Okay, with the little ampersand, and?

E Yeah. [nods her head] That's why, like I said, I like Fortran because I learned it back when, um, there was seventy two characters per line and it was, you know, this and this were real values, this is integer value. You had to float, you had to fix, those things. I prefer Fortran because it's a little more structured than Pascal and it's a little more structured than C. C I feel I might be all right in it if I started really working with it, but as long as I had something else to focus on, that is the last thing in the world I wanted to work on. [laughs]

R I see. So it was really the syntax that was difficult.

E Um-hmm. [nods her head] Because I'm more comfortable in a more structured environment.

R 'Cause you could do the parameter passing in another language.

E Um-hmm [nods her head]

R It was just in C with that syntax.

E Right.

R Okay. When you were learning in the other languages about the passing of the parameters, was that a totally unfamiliar concept to you when you learned it? Do you remember that?

E Yeah, it was an, well, you understand when I took computers the first time the only thing I'd had near, anything near a computer was a typewriter, an electric typewriter. So, yeah, it all was different. But I think that was the most difficult part of it.

R Um-hmm. Did you try to relate that to something in your own life or your own experience to help you learn it?

E Not that I remember.

R Okay. All right. As the programming courses that you took progressed along, do you think the content became more difficult?

E Yeah. Once you start getting to the higher levels, it is more difficult. Um, one thing I noticed got a lot of females out of it though was before you could even take a certain class, they told you that you had to have like this higher math and that higher math. And although that part is kind of necessary for the programming, I can see where someone who maybe hasn't had the higher math can still make it through. And I think the emphasis on the higher math was too much, maybe. Because I mean if you can program the whole program and all you need is a two-line piece of code for the math, you can always go find someone that's had the higher math or is a math-oriented person, and they'll talk them through that. So.

R Now you thought that all the math courses that are required was a stumbling block?

E I thought it was.

R Was it one for you?

E Um, well I'd had enough higher math by the time I got to C where it really wasn't, but to be honest, the higher math I had was from here, it wasn't from high school. I only went as high as Trig in high school.

R Um-hmm. And how far did you go here?

E Um, I forget, all the way up to Cal I, and Stats. But, I don't think that should be [pause]. The criteria is being able to think really, in a logical manner, not necessarily to write a formula. 'Cause you can, like I said, you can go get books, you can go get help from other people. I think that is a big stumbling block for a lot of people. I know like with my husband, he's great with it, he's great with programming. But he hasn't had the higher math and he's terrified of getting the higher math, so he'll probably never go any higher, 'cause he doesn't think he can make it through it. But he really can, because he thinks more like a computer than anybody I've ever met, you know, so. [laughs] He can figure it out even if he doesn't know the math.

R All right. Um, what about, let's talk about how you felt, your self-confidence, when you were in computer science. Did the content difficulty affect your confidence levels?

E In a positive way, yes, and in a negative way, yes.

R Okay, tell me.

E Um, when it was a difficult concept and I grasped it before, say, the people in the class, or, or whatever, then I felt, you know, hey, ok, I'm doing all right, you know. But when I didn't, then you know there's always that why can't I get this?

They do, why don't I? But, um, when I was doing poorly, then, you know, but [shrugs shoulders].

R	And when you did well, it affected it.

E	Yeah, I felt like I could handle things better, especially when you were, say, with a counterpart that was really having difficulty and you weren't for a change, then you feel like, all right! [giggles] Yeah.

R	What about the general male attitude in the class. Did that affect your self-confidence?

E	In Pascal it did. Um, I was taking that class with some other friends. One was male, one was, um, a guy I was dating. And as far as self-confidence, there was like a, a contest going on. And it, I felt pretty good about it because I kept, well, I did pretty good. But [laughs] there was a one-upmanship going on there, with the more colors, who can do this, who can do that. And in a way it made it fun. But it was a friendly rivalry. It wasn't a [long pause].

R	Cut-throat kind of competition?

E	Yeah.

R	Okay. How about your determination to succeed? When you got to the rough places? What I'm trying to figure out is when females, particularly, get to the rough places some of them persist through it and some of them quit. So when you reached the rough places, you finished out the courses?

E	In all but the C.

R And so, how did, or did the content difficulty affect your willingness to pursue this? Did it make you just grit your teeth and want to do it regardless? Or did you want to give up? Or how did you feel about it?

E Sometimes I'd feel like I'd want to give up, but then I would get mad and be determined but, at least try. I mean if I failed, then I gave it a good try. That was one thing. If I failed because I didn't try it, then, you know, what's the purpose? So, it usually made me more determined to go ahead and finish it. That's like one day, I don't care what my degree is, I am going to go ahead and get C under my belt. But, [clears throat] I just, I'm protesting it [laughs]

R What steps did you take that you can remember to understand the concepts and ideas as the programming course became more complex and difficult? Did you do anything in particular to help yourself understand and make it through?

E Like when I was writing my notes, if I could put an example in something, or, or like a little visual kind of flowchart to step me back through whatever it was I was doing, that helped.

R Something visual then.

E Yeah, like uh, a diagram, or, you know, if it was a really difficult concept, that was easiest for me.

R Okay. Why did you originally choose computer science for your major?

E When I came back to school after dropping out to get, to have kids and get married, I had already been to school for nursing and truthfully I wasn't. That was when the AIDS thing all came out, so I really didn't want to go back for that. I had to come back to school, and I had to come back on a grant; I had to come back for something I hadn't already three-fourths completed. So I decided I'd come back for business, accounting to be specific. And I took the accounting course the same time I was taking the computer courses. And the accounting is dry and boring, and the computer courses weren't. They were always changing, and interesting, and I just liked it better. I had a ball with it.

R All right. Now, what are your reasons for changing from computer science as your major?

E Because they still have that higher math requirement. That's my main one, more than anything in the world. And, once you get out of the junior college level and you go to the local senior college, which I've been to, both William Carey and USM. I have had one computer instructor at USM, um, the quality of instruction isn't there. Uh, the instructor is not usually accessible, especially if you are adjunct. You have to, you know, e-mail them, and then wait for them to email you back. But, like at William Carey, they're still running on an older version of Cobol, and, and uh, the computer class that they offered that I would have had to have taken for my four degree was uh, a Cobol class. Which I've already had Cobol, and it was a little more intensive, but they're a business college. They are not, you know, there was not the instruction there that you needed for that. And at USM Gulf Park, because I don't really want to go all the way up to Hattiesburg, the instructors may know what they're doing and how they're doing it, but the way it comes across to the student, there's not the incentive, the, um, not the attention. It's the way they do it. It's like, you know, hey, alright this is your assignment; I'll give you thirty minutes of dry lecture; you're kind of on your own. And particularly if you're a female at Gulf Park, you get the uh, you know, why can't you grasp this? Are you an idiot or something? And that really torques me off. So I decided to maybe go at it the back way. Get my four-year degree, and if I want to go do a Master's in computer science, I could do that, which would be a lot easier, and I could bypass some of the other garbage.

R I see. What are your future goals? You just kind of touched on that, but talk about it a little bit more.

E Right now it is to survive the summer. [laughs] Um, to get my four year degree in business, and then go back to school and concentrate on MIS and get a Master's in it. And it would help to find a job, if there's any out there.

R What do you like most about computer programming?

E If it's handled right it can be very imaginative; it can be exciting; it can be different. It's not the usual. That's what I couldn't stand about accounting. It was the same old, same old. They put in new rules, but the rules last for a hundred and fifty years, so the new rules are still seventy-five years down the road, and you're still dealing with somebody else's money, and you're not really, you're making figures line up in a book, but you're not really creating anything. And truthfully, if I could go back and start all over again, I would find a way I think to combine the computer and the graphics, somehow.

R What do you like least about computer programming?

E Gotta be the math. [laughs] I mean, if I understand the concept of what I'm doing, then it doesn't bug me, but when it starts getting way out there, then it's like, okay, wait a minute, let's do this step by step. It's gotta be the math.

R Okay. If you could change one thing about the programming courses, what would it be?

E The projects that they do I would try to find a way to make it where they could use their imagination and, and uh, put a lot of themselves into it instead of just following by the book. Make it come to life for them so they can understand it. I mean, um, I know in Basic, the instructor, one of the last programs she ever threw at her students was, um, a cable program, to figure out cable rates and packages, and what you got for this, and it involved the math, and the looping and all that stuff. And you could put as much into it as you wanted to. The more you put into it, of course, the better you did. And you could use your imagination. And, a lot of the books, the textbooks, don't do that. So I would change, I guess, I don't know, maybe the way it's approached in the books, even. But that's a better route for other people. But just for me, I would make it more fun, 'cause it can be fun.

R Yes, it can. What would you suggest to encourage more females to enroll in computer programming, and possibly consider it for a major?

E First thing would be to take the higher math out. Give them some sort of logical test to see if they could figure their way around it. Because, like I said, the math, you can always go get an expert for help. But if they can figure their way through that maze that you've created or whatever, they can do it. It may not have the finer points in the math, but they can do it. And, that higher math thing, that scares a lot of them away.

R Do you think, now this is just speculation, but do you think if computer science had not had all that upper math requirement, would you have stayed in it?

E Yeah, I would.

R Without a doubt?

E Um-hmm. [nods head yes] Yeah.

R Okay. Is there anything, any other comment you want to share?

E I would've probably stayed in it, too, if the original instructors that I had, as they started going into the higher math, not only understood it, used it as well, 'cause they had the book knowledge. But when it came to it, sometimes in the Fortran at least, sitting down and walking through it, they didn't know how to do it. They were a teacher, they weren't an everyday user.

R Of the programming language.

E Um-hmm. [nods] It was just a job to them. So, I think if I'd had them as more in computer science, then I probably never would have got out of it. So, that's it.

R Okay. Thank you very much.

[End of Interview]

APPENDIX J

Focus Group Interview

R – Researcher A – Abby B – Betty

C – Carol D – Diane

R ... The first question I would like to ask you is just to tell me your name and one thing that you enjoy doing.

B My name is Betty and um, I enjoy typing. [giggles]

A My name is Betty and I enjoy watching the programs work. Oh! My name is Abby. [laughs]

C My name is Carol and I like to watch TV and movies.

D My name is Diane and I enjoy roller skating and bicycling.

R Great. How much computer experience did you have before enrolling in the community college?

B Excel, and Word Perfect, and um, Lotus.

A Before enrolling in the community college, I had about six months worth of playing with programming many years ago.

C Just Excel, and Access, and Word and some accounting programs.

D Before enrolling here at Jefferson Davis, I had one semester of high school typing skills, one semester Basic, I mean programming in the Fortran language, and then half a semester of Office 97, which included Microsoft Word, and Power Point, and I taught myself Excel.

R All right. Great. In computer programming, what are some of the difficulties you've encountered in learning computer programming, at whatever level you're in? Abby, I'm going to start with you.

A In the Basic programming, the only difficulty I found was working with arrays.

C Looping and functions.

D I would have to say the difficulties I found were involving finding the time to do this. Plenty of my ideas came to later in the evening when I just didn't have notebooks sitting right next to me. That's it.

B Um, I have problems with pointers and string arrays.

R All right. How do you rate the programming courses in terms of difficulty? And if you've had more than one, tell me which course and how difficult you think that it was.

C QBasic, that's a three and Programming I looks like a four.

D Fortran, uh, on a scale from one to five? [Researcher nods] Four to four and a half, being my first language to learn. And then QBasic, um, two.

A Uh, QBasic a two, and Programming I, a three.

B Um, QBasic one, Fortran one, Programming I like a two, and Programming II a three.

R What is your idea of a good instructor?

A Someone who can make the class fun and explain it to where you understand it.

C Has to have a lot of patience because I don't get everything right off.

D Someone who is willing to spend quality extra time outside of what's written in the schedule book, um, as well as someone who is not only going to explain what's needed for the objectives, but give the perspective as to why it is that we should know these things or try to know these things.

B Somebody friendly and female, because they tend to understand female students and they, you know, ask female students questions. Whereas male instructors tend to ask males, you know, or they'll look more toward the male students. That's why.

R Do you all agree with that? [All but Diane nod]

D I really can't say either way. All three full semester classes have been taught by females. The Power Point, that class, the Office 97, our teacher was male, but he was generally um, he was just unbiased toward everyone. He really didn't say much of anything that anybody. You had to approach him, and I never really had to. I can't say either way. But I really have gotten along with my female teachers and my male teachers. I've never really noticed too much of a difference.

R Carol, you want to comment? [shakes her head no] [Researcher looks at Abby] [Abby shakes her head no] Okay. Describe your ideal classroom or learning environment. And this is not necessarily computers, just any classroom or learning environment. What would you include in it?

B Friendly, and friendly people, people who would. Both actually, students and instructor. They should be friendly and able to talk to each other without being sarcastic or, you know, be intimidating. [Abby nods her head in agreement]

A Uh, friendly, bright, warm, [laughs] and that's about it.

C Um, I guess I would have to say friendly, not too quiet, able to talk to people.

D Plenty of charismatic individuals who have skills or are open to learning techniques of debate, which I found is a great way to expose opinion and new ideas that you can't always get on your own time quickly.

R All right, now we're going to talk about having mostly male students in the programming classes. How does it make you feel? That's what I want to know. Having mostly males in the class, what kind of interactions have you noticed between the students, or between the students and the instructor, when there's mostly males, versus your other classes that may not be composed of mostly males? Do you all understand the question? [They nod] Okay.

B Females tend to participate less when, well, I participate less [laughs] when the rest of my class are male, and I don't talk as much. I listen a lot more, which is not really good because sometimes I don't concentrate as much when I have to listen. You know, I kind of wander off and [trails off]. It don't make me feel negative, it just makes me feel kind of smaller than myself, maybe. Just like I'm behind them; I feel I'm behind them, I'm not with them and that's it.

A Actually, I interact more with the class when it's mostly male because I grew up with a bunch of brothers. Um, but it does sometimes kind of single you out because you're the only one, or one of very few in the room, um, like when they get really rowdy, [laughs] which boys tend to do.

C No, that doesn't really matter to me that there's mostly males. It just doesn't. I don't know why.

D Does this apply to any class? [Researcher nods] Basic was all female in our Basic class. Actually, I have to agree. I have two brothers, older and younger. My older brother was a boy. My younger brother is very sweet and kind and, um, young, but there were always so many guys over, and when girls came over, they were always with the guys, so I had to be, to be in there, you know, involved in what's going on, I had to be in with the guys and so that's why. And when I get into class, I feel comfortable getting along with things, but I also note that you have to, in order to just speak up and talk, you can't just be speaking about, I don't know what the idiom is, just spitting out air. You have to be knowledgeable of your subject matter. So in the case of a class that I'm not totally interested in, maybe not taking because I necessarily want to, it helps; it's helped me a lot because I feel like, well, I want to be involved and I don't want to be left out, so I need to go home and really study this so that I can make this grade and that I can be a part of what I signed up for.

R Do I have any other comments? Okay. Do you ask questions in the programming classes? Any of them, or all of them.

B I did in QBasic and Fortran. Not so much in C, and not at all in Programming II.

R What about in your other classes?

B In Calculus I did, and in World Lit I didn't. It depends on the teachers. If he's very clear on all the stuff he talks about, I don't. But it also depends on how the instructor reacts to my questions. If he looked at me like, "Why did you ask me that," you know, I don't. But when the instructors are really friendly and tend to, you know, look at me for more questions, I will ask more.

A In most of my classes, in both of my programming classes I asked a lot of questions because I want to know what's coming up next and I get kind of impatient. But as far as my other classes, I tend to be really quiet. If they don't, if they act like, you know, you're stupid if you ask a question, you're not going to ask the questions as often. But if they're, if the teacher acts like they are interested in the questions and they want to answer and make sure you get it, then, you know, it's a different story. And then I usually ask questions in classes like that.

C In my programming classes, I asked a lot of questions, but in most of my other classes I didn't ask many questions.

R Why is that?

C Because in my other classes it's easier for me to follow along than in my programming classes.

D If I feel like there are not, is not enough participation outside of my own asking questions, then even when I'm not really unsure about a subject, but if you sense the mood of the class or of certain other people in it and it seems like someone is just not getting it and that particular person has given up on the whole idea, you know, I just don't get it so I'm not even going to bother. You get to know people like that don't, they just don't get involved with speaking their mind any more because they've, for whatever reason, so, I'll ask, um universal questions. Also, in the way the instructor, if I really am interested in something, and I feel like I'm being shunned in a way, if he's having a busy day, or she, whatever, if they just need to get on with it and you know, that I'm wasting time, then I'll ask more. [laughs]

R All right, how often do you require help with the new concepts or new programs, and this is in computer science. Think about it. When a new concept is presented, or a program, how often do you need help? Give me an idea.

B About two out of three. In the earlier programming courses, I didn't have any problems, it just, you know made sense, all made sense. But in Programming II, it just, I started having problems in Programming II with the introduction of pointers. [laughs] And after that, after pointers and the classes and you know, I started having a little difficulty with all that shock.

A I had problems with about two out of three where I needed help with about two out of three of the programs or new ideas. But most of it was just that I was questioning what comes next, because I wasn't, I was unsure. I thought I knew what was coming, but I wanted to make sure. It was generally the program itself, rather than the concept because the ideas are pretty simple to pick up on. So far. [laughs]

C I would have to have help every so often, because sometimes I would like forget the concept while I was doing the program, and so I would have to be, like, reminded.

D On those concepts that were not covered in available literature, I tend to ask for help. On a side note, I found that a lot of problems that are, seems now are inherently human, being in a capitalistic and a world economy, that they arrive financially, and there is an entire fifty, seventy foot aisle of computer literature, programming, at the bookstores. I just can't get them. I can't afford them, and that's one of the things that brings me into the future business, and so, if it's written, I will read it.

R Do you have a study partner or a study group in computer science and in your other subjects? Talk to me about those things.

B I don't have a study partner in any classes. I think I like, I prefer studying alone, so if I try for about two weeks and can't understand something, then [laughing] I go to somebody.

A In my computer science classes I don't have a study partner as a general basis. Now if I don't understand something, I will go to someone else and say, you know, "What's up with it?" And for most of my other classes, I don't either. For Psychology, I tend to study with someone to help them, and it also helps me. And that's about it.

C I don't have a study partner for my computer classes. If, you know like Liz, if I need help I might go to somebody in the class, but in my other classes, I don't have a study partner.

D My group of friends and I are such unique, we're all so different, background, and I have, you know, in the initial interview I mentioned that I could count all my good friends on two hands, but I have friends who are mothers, friends who are working mothers, you know, a few very young girls, but, in a way Renata. She is eighteen, wanting to get on her own, having had problems with her mother, living with her aunt. She comes over, and we've developed a really unique friendship that was based on her wanting to know a little bit more about what is it like to be twenty-two and living alone? And how long have you been doing this? And is it tough getting adjusting. And so we study in a different way, but we don't have, it's all about we don't have the same course material, so we really can't. She's not taking this class and I only know one person in my Basic, knew in my Basic class that just ended, and from high school, but she is, you know, married, lives with her mother, works and didn't seem to need any help, so we didn't develop a study buddy bond. But that's why. It's because I have a friendship base and I'm more than willing to be open to that. But I just, I guess I currently prefer to study alone.

R All right. Do you like to work in a group computer lab setting or do you prefer to go home and work by yourself?

B Lab setting because you get distractions at home and I don't like working in a group.

A I like to work in the computer lab setting because there are usually instructors around or someone who knows what they're doing if you have a problem. And, like she said, you do get a lot of, especially when you have kids, you have a lot of distractions at home, and it's just easier and more quiet in the computer lab.

C I like to work in the computer lab because if I need help I can go get it, and when I'm at home and I get stuck, I just stop doing it and do something else.

D Because of the quiet environment that I have at home, and as few distractions, that I don't have children and relatives living with me, I see the computer lab as more of an extension of home, because, only because I don't have a computer. But if, so I do all the paperwork and all the paper, what you call that?, the paper programming, I do all that at home. And if I had a computer, I would do all of it at home and come here to ask questions, write it down, go back and do it, because I look forward to the distractions.

R Are you satisfied with your level of understanding right now with computer science, or do you feel that you have missed something along the way?

B Hmm, I guess I know enough for this level. But, um, if I can choose, if I had the choice, I would prefer to know more and have more time on it.

A As far as the programming goes, um, I'm satisfied with my level of knowledge. But I feel like I missed something in the operating systems before the programming [laughs] that I didn't get.

C Um, I feel like I could have learned more because it's just like you're in class and the teacher shows you, like, tells you how it's supposed to work, but when you get in the lab, it's kind of like Greek, and you have to go back and figure it out.

D On one hand I feel like, I always feel like I've missed something in the way of public schooling. We did get computers eventually in English class, fourth grade. The day that we were supposed to do that, I happened to be in my gifted studies class, and I was never able to learn those. But, um, that's nothing. On the other hand, I'm really glad that now the children in public schools are, um, have access to that, and I know that I don't take for granted the options that are available to me, and I just take them as I see the opportunity arises to do so. And I'm confident of my abilities that whatever I end up with in the long run I can build on and I can look for work.

R All right, now I want to talk to you about, or actually I want you to talk to me, about how much math have you taken, and do you think the math classes have helped in your computer programming, and if so, in what way?

B I finished Cal II. I don't feel like it did very much to help, although in the earlier introductory courses the equations are very similar and that's what, sort of what I used to help me understand.

A I just finished Trig and like she said, the math classes at the beginning, the Algebra, College Algebra, all of that, it does help. A lot. You get past that, I don't see where it's helping any at all.

C Um, College Algebra, uh, not really.

D I've gone the series from elementary to pre-algebra to algebra, geometry, trig, I've done pre-calculus, and then I've just finished the Calculus II. Um, I actually think they've all helped, mostly in the Algebra, and of course, Calculus to me seemed ninety percent, you know ten percent, five percent Calculus and eighty percent Algebra and then another ten Trig. And so, um, Algebra is very, ever present. But I've found that as the scenery grows in math, as I learn more, it just tends to broaden my horizons, my perspectives, and it opens my mind to a new way of learning which permeates to all other courses, and in the case of computers, that's not an exception.

R All right, that was interesting. Okay, now here's really what I'm trying to get at. When you learn computer science, it is so different than most other things you have ever learned. Did you all notice that? [They nod] Programming is very different because it uses the left side of the brain, which is your Algebra, your logic, but it also uses the right side of the brain, which is the language, because you have syntax and language rules that you have to follow. It's like learning a foreign language. When I talked to you in your individual interviews, I got some very good information from you about how you integrated the new ideas into what you already knew inside, and we call those inside structures frameworks. And so what I'm going to ask you is how did you build, because you had to do this, how did you build the necessary mental framework inside your head for understanding computer programming? Because you have been successful in sticking it out, and you have the necessary framework inside to understand it. I want to know, how did you do that?

D My mother has told me for years, "You are so good at math. You have to go into math, science field." I've always wanted to work at NASA if I could ever get to that level. But the one thing she did always tell me is that math is like another language, and she would elaborate on that a bit. And that's when she, when I got into high school, she pushed me, or, you know, coerced me to take French, which I was already interested in. To me it seems that the computer science is different, in a way it's an applied science. It's as conceptual as all the other fields that we study. We don't just study the English language, but we study it in order to use it. We do the same here, except it's so much more involved because it takes in what we've learned in learning language skills, what we've learned in math, it's a science, a working component and a philosophy, you know, a thinking machine. So I've noticed that it's an applied thing, and um, I just think that, uh what was the actual question? [Researcher repeated the question] I think that my mother built that framework by talking with me about from the beginning, by telling me that this field, this math and science, the unconventional field for women is my way to go, and by bringing up the initial idea. It's like another language and you're very good at it, and as I got, as I excelled in French, she still used that, and would tell me how proud she was of me, and I think that she's the one that built the framework, and I've just been laying the rest of the foundation, in whichever order that goes.

C Um, what was the question? [Researcher repeated the question] I guess I just treat it like when I was learning history and English, just by adding on to different things. By adding on to what was already in there.

A There wasn't really much building to it. It was just memory work and connecting what you already know. Because if you're good at math and you're good at grammar and English, then it's just piecing it together and putting the two together to make computer programming. It is just like learning the English language when you're growing up. And so you just add it to, or connect it with, something you've learned that it's like in just everyday life.

B I relate the different concepts in programming to what I already know, like, um, except for pointers. Because it doesn't exist, I mean you can't, you can't, well actually I guess you could. Because a sign on the street points to something else, so I guess, I got it! [Excitedly laughs as everyone cheers] Okay, it took me a little long. I guess I just remembered pointers. [laughs again]

R Now Betty, I need you to explain to me about the reason that you didn't understand pointers real well was because, I'm just trying to get what you just told me, was because you couldn't relate it to something in your experience. Is that right?

B Yes. I had initial problems with pointers because it didn't make sense. It's not like arrays. Arrays you can, you know, it's like a counter you add to it, and it's a space that is allocated in memory. But pointers, it um, it holds the memory of something else. It doesn't hold a value, and I guess I just didn't understand you know, it, pointing at something else. And having an array of pointers that holds the addresses of stuff, instead of what, you know, the value and the function, or whatever. Well, the word "pointer," the word "point," you know it points at something. I just saw a finger, like point. [Points her finger] Yeah, I understood pointers; it just took me a while, and I can't believe how easy it would have been if I had thought about that.

R All right. This is great. I want to talk to you about, I want you to talk to me rather, about the difficulty. We already know, um, did we already say the things that you had difficulties with? [All nod] I thought we did. Did you encounter any difficulties in computer science that made you want to quit? If so, what were they and why didn't you quit? Who wants to go first on that one?

A Two-dimensional arrays drove me nuts. Um, but I haven't had to work at computer science all that, at programming all that hard. And the first thing that I really came up with that gave me trouble, and I thought, "If this is all I got to deal with, or this is the worse it gets I'm gonna keep going."

D Oh, gosh. Um, gee, you know what made me want to quit was the week of finals. I was done. I was so numb at this; I started a new job this last week, and I was, like, I'm so glad this is over, because whatever I didn't learn here I'll pick it up some other time. But, it was just the whole idea that okay, I'm at the last minute of this, we're not going to elaborate any more and so, um, though I don't quite understand two dimensional arrays, we are not, we might have a question or two about some test, but we didn't really study it, we never applied it, so I'll just get an idea of what they're about, but when I started to give some time to it, I thought, no, I just don't want to. That's when I quit that, but I would never do that halfway through a semester. I didn't even have the idea in my head any time until the very end when, uh, it just didn't matter.

C Um, the most difficult I had part about it was the amount of time for programs to get done, and, that's about it. And then, it um, I thought about quitting, like, it was more than halfway through, but I just thought I might as well stay in until the end.

B Um, the first pointer assignment made, almost made me want to quit. But Calculus made me want to quit. [giggled] It's just a lot harder. You can't understand math like you can with computers. You know, programming? That is nothing like math. You can't do a hands-on. You can't like build something and then just like to slice it into pieces. Hands on stuff helps you understand a lot more, and mostly the math that made me want to jump off the building.

R Okay. All right, what are the biggest obstacles you have had to overcome to remain in computer science? There may not be any, but if there are, what are they? Now keep in mind that we are talking about this because most of the females don't stay in computer science, and we're trying to identify places that might be obstacles. And, if you ran into any, then I want to know about it.

D Okay, this is my own idea and it's personal to me, and it's the truth. Um, these days, uh, many girls either have been given or have bought a car, um, have boyfriends who take them places. I had all that, you know, until I went to college, and, but I lost my car due to a flood. Now, the only obstacle I have is the fact that there are a lot of things that I have to um, I have to think about and worry about every day, like how am I going to next semester take my physics and calculus at another county when I still haven't bought a new car yet. Um, to walk to school in the rain, or wait for the bus, you know, in downpours, and then getting here and, you know, your hair has fallen, and you have no makeup on; you're just wearing a t-shirt and whatever, and then there are these cute girls walking around, you know, just looking like they spent, someone did their hair for them this morning and does it every day, and creases in their pants or hot pants, and you know, and the thing is you just don't get attention and, which actually I think helps my mind because you don't have to worry about it. But, the thing is I have, I have friends, I have guy friends, girl friends, um, who know me personally and my situation, so those would be obstacles have actually just become a part of life. Oh yeah, I forgot. I don't have a computer. I forgot about it; I'm in a room full of them. That helps, but um, yeah. That's something else, but I have to get a car first.

B Finding time to work on it is one of it because of Zach, my son. And um, and [pauses] I don't have any obstacles. No having somebody to relate to, to talk to in class. Females like to have somebody there for support, and when you're all by yourself, you know, the first time you have trouble, you run.

A Um, one obstacle is not having a, while I have a computer at home, I don't have the compiler it takes to run the programs at home. So, that's a big obstacle, but the school takes care of that and can fix it. Um, there's also a time obstacle involved. Finding time, whether you've got a computer at home or whether you have to do it at school, finding time around children and work and other things to, uh, to get the work done. And, uh, and a financial obstacle, which can be helped in most cases, um, but it still makes it rough. When you have to take a lot of your time spending on the school and an education, you don't have a lot of time to do other things and make money to support the rest of the other things [laughs]. That makes it hard. The people around them can be a big obstacle, because a lot of times I've had, especially with males, in general, even though they're not, they may not be serious, they may be serious, they tease, women get teased a lot about you know, you have the dumb blonde jokes, you know, and they actually, a lot of women take that seriously. And they don't think they have what it takes to make it through the programming classes and get the degree.

C Well, my biggest, one of my biggest, obstacles in computer science, you have to have a lot of math, and I've never really been good in math. And if I'm not interested in something, I tend to let my mind wander, so it's hard for me to keep my mind just on one thing.

R All right. Do you two, can you think of any obstacles that other females that you know might have run up against and caused them to drop out of computer science?

D I would say a big obstacle is reassurance. We need reassurance, um, I think in every area of our life and life, love, friendship, relationships, family, and then in what we do, in careers. And um, the most successful women, you know, are those who have people saying, you know, that report or that article you wrote, that was amazing, you know; I would like you to do this again, and again, and again. And, um, that kind of reassurance really lets you know what you're doing is respectable and it's, um, making a difference in somebody's life, whether it's selling more magazines or, you know, more software.

C Well, the ones, the women that I know that was computer science majors, um, they dropped out because, um, they didn't like the math. They didn't understand why so much math was needed. And when they was working on their programs, they got to a place where they didn't understand, and they were like, if I can't get this, then I won't ever get it. So they just dropped out.

R Okay, um, how about let's talk about your own confidence and your own abilities. Did you ever get to a place in your computer science programming courses where you questioned your own ability or your own confidence to do this? And if so, talk to me about that a little bit.

D Uh, I have to say that questions of confidence are no longer much of an issue as they were two, three years ago, even six, seven months ago. Because of those issues that did test my self-esteem, um, because they have um, kind of gone away, I've really had a chance to take some time and think them through, and they were gone before I enrolled in this class. I really, I really didn't think that I had any issues with confidence on being able to do this because I've had to rely solely on that for so long, on my ability to walk standing straight up, and you know, looking forward and um, you know, just trying not to be stepped on.

C Mine would have to be is when like a concept was introduced and I didn't, I didn't understand you know, why it was used for, and I just couldn't seem to get it. That was, I ask questions about it and tend to read about it.

A There were a couple of times my confidence was tested. Mostly, or the biggest one, was in the beginning when I started my Basic class and I hadn't had any kind of, hardly any kind of programming in fifteen years, and what little bit I did have was not much. Um, and I went, I went to class and I sat down, and there were all these guys talking about video games, and how to take them apart, and do this and that and everything else. Wait a minute, like, and they were using all these letters and I'm like, what are these people talking about? You know, uh, Lord, I must be the, I must be the biggest idiot in this class, because I have no idea what they are talking about. [laughs] Right. But then I realized, you know, that's just a step I haven't gotten to yet in the degree and that it will come, it just, I need to get through these courses first.

B She's right. Guys talking about all the stuff you don't understand about, you don't, you have never heard of, makes you go, "Okay, maybe I don't know quite as much" [giggles], and other times it's the programs, in the process of doing them, I go, "I cannot do this. I just can't do it." But um, my situation doesn't allow me to quit, you know, I'm on a limit, a time limit. I have to get out with like a Bachelor's degree so Zach can have something. And my husband is also very persistent.

R All right, would you say that there are people in your lives that have encouraged you to stay with computer science, and if so, who are they?

B My husband. He says, "You have to do it because I have to go back to school!" So, I can't quit. And my dad has been on my case since a long time ago. So I can't quit. And Zach. I do, do it for my son. And those are the people.

A Um, my mom has pushed a lot. It was my idea, and she knows why I'm doing it, so she'll, if I get down like I don't care, I give up, you know, she'll sit there and say, "No you can't give up. You started this; you're going to finish it." And my brother, my brother you know, gives me reassurance and says, "You're in one of the best fields that you could ever find." And I have my, and I push myself because I started this because I have three children and I'm single. So, and each time I get to where I'm going to quit, if they don't say something, I just, I just look at the kids and say, "I've got to, I've got to finish."

C I guess my mom because she's, she doesn't tell me, "Well, you started it, you finish it." She always tells me, "Well, you don't want to look, you don't want to quit and then fifteen years from now want to do something. So you might as well finish."

D I have a third party that does, that helps keep me, um, going. I, my mother and I parted years ago, which is a blessing to me because she's been quite ill and works two jobs. She has a disabled husband and a thirteen-year-old to chauffer, and um, and she has, does not have the time to find the words, much less to say them, so I have to just know that she would if she could. And though she, she got me involved with vocational rehabilitation, government agency, Mississippi state government agency, that um, by explaining to them before I knew what she was doing, my financial situation that I just, um, I have to be on my own of twenty-two and um, for what reasons, I can't live with, um, either parent. But in Tijus over in Texas, the step mom won't stand for another kid around, um, and especially not one who's opinionated. [giggles] So, I have a third party that I have to send my grades. I have to um, follow a strict regimen of um, applying for government aid, first of all. Whatever is not paid for through Pell grants is covered, plus books and, um, they're not supposed to do supplies, but they did buy a calculator for me. So it's fairly more me. I have to keep myself going, and um, remember the things I was told by my mother when I was young and impressionable. And then I call Miss Karla, and she says, "That's really good. I'm glad to know that you're doing well." And she gives me little, you know, tidbits of admiration and a little bit aspiration. And then, um, soon as my grades come in next week, I'll mail them to her, and then my incentive is two free summer school courses.

R And who is Miss Karla?

D Miss Karla is my voc rehab counselor. Karla Drummond. She's very kind; she's disabled and is sick, out a lot, but she works. I mean she works for the people. She has a job that influences, that affects other people's lives, and she knows it, so she is very personable.

R Okay. So I'm really interested to know that all of you, did you hear this, that all of you have support people to encourage you. I'm wondering if that's one of the things that maybe the other females don't have, is some encouragement.

B It can staple them to the ground.

R That's a good analogy to staple them to the ground. All right, so here are some ending questions. First, what is your motivation for continuing with computer science as your major?

B For Zach to have a bright future and make big bucks! [everyone laughs]

A Computer science, I went into computer science because it, it's one of the best money making fields in the world and everything is going to computers. So if you don't know how to work one, you're just going to be SOL. [laughs]

C Um, I don't really know. I guess, um, I don't know.

D Well, I think, I like what she said. Like my uncle always says, "Up shit creek without a paddle." But that's true. Um, I know now, I knew this just a minute ago, but I've just forgotten. [Everyone laughs] Um, oh yeah, I had something witty, but you know, the same reason. I want things. Um, I want to have, I want to be able to, Oh! Yeah! I know. There are lots of guys in it. [Everyone laughs] But I want to make, I want to make a respectable income, and find a respectable husband, and have children that can, um, look forward to, you know, the kind of education that will lead to scholarships and grants. And then no worries about how am I going to, you know, buy my laundry detergent while I'm in college and, uh, you know, I really don't want to sign up for that meal plan because it's way overpriced so can you just, Mom, I want this much money a month so I can buy my own meals or groceries." I'd like to be able to say, "Okay, you know, you've proven it, and ..." I want the amenities.

C Um, I guess because like they say a lot of jobs you have to know computers, and I want to be, have a job where I don't have to worry about, you know, money from week to week.

R Well, you all know, maybe, that last year there were 130,000 jobs that went without filling them. They were empty because there were no people to fill those jobs in computer science. 130,000 jobs. All right, um, let's talk about what would you suggest to encourage more female students to enroll in computer science?

B Having female classes so they will be more open to speaking up and um, sharing ideas, and what was it I said yesterday? [giggles] Um, anyway, I guess that's it.

A That's about the only thing I can think of as well is letting them know, having more female-oriented classes, which may or may not help.

C Um, I would have to say that we just need more females, and just let them know when they get finished they'll have a better, probably have a better opportunity to get a job.

D I would remind my friends that, and my new acquaintance, that there still is a salary, a glass ceiling, and I would give the new statistic that I've just learned to that person, or those people, and let them know if you can do something that some other man that comes in and he can't do it, well, then, you're going to get paid more than he will. It doesn't matter what your gender is, because you have worked for it, you know, and they've shown, trends have shown that a man and woman can do the same thing, sad but true, the man will still get paid a certain percentage higher. But if the woman can do it and the man can't, well then he won't get paid diddly.

B Having a friendly instructor helps a lot. One who will probe; what you do. [laughs] You know, you helped a lot. The initial course is what got me started and let me decide that I want to do this, and an instructor who makes it fun for the females and let them relate, like, you know, the grocery thing, like maybe that's something I did, I can't remember. But one of the things, one of the programs I did made me go, "Oh, I can do this at home," you know, or something I can apply to. And an instructor who is, who doesn't make fun of the student, and makes sure that the other students doesn't make fun of the student, helps a lot too.

R Okay. Last question. I want you to tell me real quick what you like best about computer programming, what you like least, and if you could change one thing about it, what would it be? Got that? Best, least, and if you could change one thing, what would it be?

D I like best that it is organized, structured, easy to understand for a person who is willing to invest the time and effort into it. I like least that, um, it is not so readily um, available, the information that a person needs in order to learn the things to invest the time and effort into this, that it's um, but that thing taken care of through exposing children these days in public schooling in and around the country and in other parts of the world. What I would change, um, well, I would fast-forward about two decades and, you know, get the technology and bring it back. But since that is totally, um, unreasonable at this time um, I would, um, I would take people from the field, and I would convince them to take nine weeks of every year and teach, so that we can offer um, not just the objectives, but the perspectives that is needed to, in order to motivate today's students to, um, to do what's actually going to be required of them in order to keep this advancement at a steady and hopefully, a non-environmentally damaged pace. We are killing ourselves, and that computers are going to be able to help us slow down a lot of that.

C Um, I like most, um, getting the programs done. That's what I like most. I like least, um, sometimes the concepts, they just seem like, uh, not useful to me. Um, I'd change, um, I don't think there's anything I would change but some of the concepts, like functions.

A I like most watching the programs run and see them come up on the screen. That is kind of interesting. Um, least; I like two-dimensional arrays the least because they are hard to get, hard to get down. Um, if I could change one thing, I wouldn't necessarily have the people in, um, in industry teach, but I think it should be a requirement that each one go to the high schools or, you know, the colleges, and let the people that are going into these, that are starting these fields, know what they can do with them. You know, because there are so many different, uh, areas and people don't know. Well, it's just a, it's a general overall degree. They don't realize they can specialize in many different areas in whatever they want to do with it.

B I like knowing that I can do something very few other people can do. And it makes me feel powerful. And, um, I don't like, there's really nothing I don't like about it, except for the math. And, um, if there is something I would change it would be the amount of lab time we get in class. Like I said before, hands-on is more important, well, it's not more important, but it's very, very important for somebody to understand it. And that would be what I would change.

R All right, does anybody have any final comment about your feelings about being one of the few females in computer science?

B I sort of like it because I know I'm one of the few people who can do it.

R All right, anybody else? [They shake their heads] Well, thank you ladies for doing this.

[end of our interview]

APPENDIX K

THE UNIVERSITY OF SOUTHERN MISSISSIPPI

AUTHORIZATION TO PARTICIPATE IN RESEARCH PROJECT

Subject's Name ____

Consent is hereby given to participate in the research project entitled "Gender Underrepresentation in Beginning Computer Programming Courses." All procedures and/or investigations to be followed and their purpose, including any experimental procedures, were explained by Mrs. Peggy West. Information was given about all benefits, risks, inconveniences, or discomforts that might be expected.

The opportunity to ask questions regarding the research and procedures was given. Participation in the project is completely voluntary, and subjects may withdraw at any time without penalty, prejudice, or loss of benefits. All personal information is strictly confidential, and no names will be disclosed. Any new information that develops during the project will be provided if that information may affect the willingness to continue participation in the project.

Questions concerning the research, at any time during or after the project, should be directed to Mrs. Peggy West at (228) 897-3819. This project and this consent form have been reviewed by the Human Subjects Protection Review Committee, which ensures that research projects involving human subjects follow federal regulations. Any questions or concerns about rights as a research subject should be directed to the Director of Research and Sponsored Programs at (601) 266-4119.

A copy of this form will be given to the participant.

______________________________	____________________
Subject's Signature	Date
______________________________	____________________
Signature of Auditor – Witness	Date

REFERENCES

Arenz, B.W., & Lee, M.J. (1990, April). Gender differences in the attitude, interest and participation of secondary students in computer use. Paper presented at the Annual Meeting of the American Educational Research Association, Boston, Massachusetts. (ERIC Document Reproduction Service No. ED 327 389)

Association for Computing Machinery. (2001). Computing Curricula 2001. Report from the ACM/ICEEE-CS joint task force on computing curricula 2001. New York, Author. (www.acm.org/sigcse/cc2001/)

Bogden, R. C. & Biklen, S. K. (1998). Qualitative research for education: An introduction to theory and methods (3rd ed.). Boston: Allyn & Bacon.

Bunderson, E.D., & Christensen, M.E. (1995). An analysis of retention problems for female students in university computer science programs. Journal of Research on Computing in Education, 28, 1-18. (Academic Search Elite Reproduction Service No. AN 9601090566 ISSN 0888-6504)

Butler, D. (2000). Gender, girls, and computer technology: What's the status now? Clearing House, 73(4), 225-230. (Academic Search Elite Reproduction Service No. AN 2873300 ISSN 0009-8655)

Camp, T. (1997). The incredible shrinking pipeline. Communications of the ACM, 40, pp. 103-110. (http://www.mines.edu/fs_home/tcamp/cacm/paper.html)

Camp, T., Miller, K., & Davies, V. (in press). The incredible shrinking pipeline unlikely to reverse. Communications of the ACM. (http://www.mines.edu/fs_home/tcamp/new-study/new-studt.html)

Charlton, J.P., & Birkett, P.E. (1999). An integrative model of factors related to computing course performance. Journal of Educational Computing Research, 20(3), 237-257.

Charlton, J.P., & Birkett, P.E. (1998). Psychological characteristics of students taking programming-oriented and applications-oriented computing courses. Journal of Educational Computing Research, 18(2), 163-182.

Clegg, S. & Trayhurn, D. (2000). Gender and computing: Not the same old problem. British Educational Research Journal, 26(1), 75-90. (Academic Search Elite Reproduction Service No. AN 2980632 ISSN 0141-1926)

Computer Gender Gap Prompts Major Study by ACM. (1998, November). Association for Computing Machinery Press Release. (http://www.acm.org/announcements/acmwnsf.html)

Crombie, G. & Armstrong, P. (1999). Effects of classroom gender composition on adolescents' computer-related attitudes and future intentions. Journal of Educational Computing Research, 20(4), 317-327.

Damarin, S. K. (2000). The mathematically able as a marked category. Gender and Education, 12, 69-86. (Academic Search Elite Reproduction Service No. AN 2969133 ISSN 0954-0253)

DeBello, T.C. (1989, March). Comparison of eleven major learning styles models: Variables; appropriate population; validity of instrumentation; and the research behind them. Paper presented at the National Conference of the Association for Supervision and Curriculum Development, Orlando, FL.

Dryburgh, H. (2000). Underrepresentation of girls and women in computer science: Classification of 1990s research. Journal of Educational Computing Research, 23(2), 181-202.

Dunn, R., Denig, S., & Lovelace, M.K. (2001). Two sides of the same coin or different strokes for different folks? Teacher Librarian, 28(3), 9-16. (Academic Search Elite Reproduction Service No. AN 4256926 ISSN 1481-1782)

Dunn, R., Dunn, K., & Price, G.E. (1977). Diagnosing learning styles: A prescription for avoiding malpractice suits. Phi Delta Kappan, 58(5), 418-420.

Dunn, R., Dunn, K., & Price, G.E. (1981). Learning styles: Research vs. opinion. Phi Delta Kappan, 62(9), 645-646.

Dunn, R., Dunn, K., & Price, G.E. (1996). Productivity environmental preference survey. Lawrence, Kansas: Price Systems.

Durndell, A. & Thomson, K. (1997). Gender and computing: A decade of change? Computers and Education, 28(1), 1-9.

Fan, T., Li, Y., and Niess, M. (1998). Predicting academic achievement of college computer science majors. Journal of Research on Computing in Education, 31(2), 155-172.

Gardner, H. (1991). The unschooled mind. New York: BasicBooks.

Gipson, J. (1997). Girls and computer technology: Barrier or key? Educational Technology, 37(2), 41-43.

Henwood, F. (2000). From the woman question in technology to the technology question in feminism: Rethinking gender equality in IT education. The European Journal of Women's Studies, 7(2), 209-227. (EBSCO Online Citation ID ETUCDC78E3J94U107P14)

Joyce, B. A. (2000). Young girls in science: Academic ability, perceptions and future participation in science. Roeper Review, 22(4), 261-262. (Academic Search Elite Reproduction Service No. AN 3379031 ISSN 0278-3193)

Kadijevich, D. (2000). Gender differences in computer attitude among ninth grade students. Journal of Educational Computing Research, 22(2), 145-154.

Katz, L., Maitland, M., Hannah, R., Burggraf, K., & King, S. (1999). The effects of gender and academic program on learning styles and attitudes of undergraduate students using multimedia, web-based anatomy labs. In J. A. Chambers (ed.), Selected Papers from the 10th International Conference on College Teaching and Learning (pp.81-87). Jacksonville, Florida: Florida Community College. (ERIC Document Reproduction Service No. ED 431 343)

Kirkpatrick, H. & Cuban, L. (1998) Should we be worried? What the research says about gender differences in access, use, attitudes, and achievement with computers. Educational Technology, 38(4), 56-61.

Koch, M. (1994). No girls allowed! TECHNOS, 3(3), 14-19.

Krueger, R. A. (1994). Focus groups: A practical guide for applied research (2nd ed.). Thousand Oaks: Sage.

Learning styles (1986, January). (Guides – Non-Classroom Use; Tests/Evaluation Instruments) Columbia: Missouri University, College of Education. (ERIC Document Reproduction Service No. ED 323 249)

Magolda, M.B. (1989). Gender difference sin cognitive development: An analysis of cognitive complexity and learning styles. Journal of College Student Development, 30, 213-220.

Mahony, K. & Van Toen, B. (1990). Mathematical formalism as a means of occupational closure in computing – Why "hard" computing tends to exclude women. Gender and Education, 2(3), 319-333. (Academic Search Elite Reproduction Service No. AN 9707102721 ISSN 0954-0253)

Margolis, J., Fisher, A., & Miller, F. (1999). Caring about connections: Gender and computing. IEEE Technology and Society Magazine, 18(4), 13-20.

Margolis, J., Fisher, A., & Miller, F. (2000). The anatomy of interest: Women in undergraduate computer science. Women's Studies Quarterly, 28(1&2), 104-127.

McCullough, M.K. (December, 1996). Student participation in the science classroom: A gender issue. Paper presented at the Annual Research Forum, Winston-Salem, NC. (ERIC Document Reproduction Service No. ED 418 941)

Miller, S.H. (2000, September 11). Indianapolis group encourages women to enter technology fields. The Indianapolis Star, p. 1. (Newspaper Source No. 2W63733509208)

Mitra, A., Lenzmeier, S., Steffensmeier, T., Avon, R., Qu, N., & Hazen, M. (2000). Gender and computer sue in an academic institution: Report from a longitudinal study. Journal of Educational Computing Research, 23(1), 67-84.

NCES (1996). Digest of Educational Statistics. National Center for Education Statistics, U.S. Department of Education, Washington, D.C.

Patton, M. Q. (1990). Qualitative evaluation and research methods (2nd ed.). Newbury Park, CA: SAGE.

Philbin, M., Meier, E., Huffman, S. & Boverie, P. (1995). A survey of gender and learning styles. Sex Roles, 32(7/8), 485-494.

Price, G. E. (1996). Productivity environmental preference survey manual. Lawrence, KS: Price Systems.

Ray-McCutcheon, M. (1996, December). Gender-stereotypical behaviors in high school classrooms. Paper presented at the Annual Research Forum, Winston-Salem, NC. (ERIC Document Reproduction Service No. ED 418 941)

Revilla, A. T. (1998, March). "Sugar and spice and everything nice..." Gender inequities in mathematics. IDRA Newsletter, 11-12.

Rothschild, B. (2000). Computing gender bias. Humanist, 60(2), 36-37. (Academic Search Elite Reproduction Service No. AN 2831144 ISSN 0018-7399)

Sadker, Myra and Sadker, David (1994). Failing at Fairness: How America's Schools Cheat Girls. New York: Macmillan.

Severiens, S. & Dam, G.T. (1997). Gender and gender identity differences in learning styles. Educational Psychology, 17(1/2), 79-93.

Shashaani, L. (1994). Gender differences in computer experience and its influence on computer attitudes. Journal of Educational Computing Research, 11(4), 347-367.

Shashaani, L. (1997). Gender differences in computer attitudes and use among college students. Journal of Educational Computing Research, 16(1), 37-51.

Skemp, Richard (1987). The Psychology of Learning Mathematics. Mahwah, New Jersey: Lawrence Erlbaum Associates.

Snyder, R.F. (2000). The relationship between learning styles/multiple intelligences and academic achievement of high school students. High School Journal, 83(2), 11-20. (Academic Search Elite Reproduction Service No. AN 2742795 ISSN 0018-1498)

Spertus, E. (1991). Why are there so few female computer scientists? (Technical Report 1315). Cambridge: Massachusetts Institute of Technology, Artificial Intelligence Laboratory. (http://www.mills.edu/ACAD_INFO/MCS/SPERTUS/Gender/pap)

Taylor, H. G. & Mounfield, L. C. (1994). Exploration of the relationship between prior computing experience and gender on success in college computer science. Journal of Educational Computing Research, 11(4), 291-306.

Tech-savvy: Educating girls in the new computer age. (2000). American Association of University Women Educational Foundation Commission on Technology, Gender, and Teacher Education. Washington, D.C.: Author. (http://www.aauw.org/2000/techsavvybd.html)

Weinman, J. and Cain, L. (1999). Technology – The new gender gap. Technos, 8(1), 9-12.