INFORMATION TO USERS

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

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

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

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

ProQuest Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

Project Demonstrating Excellence

A CURRICULUM IN COMPUTER SCIENCE:
A PROBLEM SOLVING APPROACH

By

George Daneluk

Submitted to U G S Committee
In Partial Fulfillment of Requirements for the
Degree of Doctor of Philosophy

Union Graduate School
Cinncinati, Ohio
June, 1980

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

ACKNOWLEDGEMENTS

I wish to express my deep appreciation to Dr. Edward V. Bonnemere, whose encouragement and example led to my taking on the project, Dr. Hilda Lee Dail, whose continuing support helped me move from chaos to creativity, Valerie D'Antonio, without whose efforts I would not have had a project, Suzanne Hain, who will recognize many of the ideas incorporated into it, George Foley, who once again provided my update in the field as he has on a perennial basis, Ellen Koeppen, whose conscientious work at the end helped me get it all down on paper, Tony Montanaro, whose advice enabled me to maintain a philosophic perspective with respect to the technical aspects of content and structure, Dr. John Reckzeh, who continues to serve as mentor in my long term mathematical development, Dr. Halloway C. Sells, whose steering brought the whole ship across at high water time. I also wish to express my appreciation to Joan Palazzi and the rest of the staff, too numerous to mention, of the Systems Technical Education Center of AT&T Long Lines for the courtesy and cooperation extended during my internship.

ABSTRACT

The Project Demonstrating Excellence is an adaptation of a systematic, "phased" approach to job training applied in an academic environment to the development of a total curriculum in computer science. The process of job training development is based on an application of general principles of systems analysis and design to course development. The PDE specifically documents the "front end" problem analysis, the objectives, and the preliminary design of the overall curriculum. The problem/opportunity identified for resolution is the need for critical thinking and rational decision making viewed in a technical sense in systems development. Skills and knowledge sufficient to enable graduates to perform in functional roles in a development team, in operations, and in technical support are identified based on a case study at AT&T Long Lines. End-of-curriculum objectives and enabling objectives are specified in terms of the skills and knowledge requirements. A model curriculum designed to enable the objectives is given with components in Business Information Systems Development, Hardware Organization, Software Engineering, and Mathematical Foundations. A case is made for a problem solving approach in each of the four components. The developmental activities are documented according to documentation standards for program approval of the New Jersey Board of Higher Education, the Jersey City State College Senate Curriculum and Instruction Committee, and the Training Development Standards used at the Systems Technical Education Center of AT&T Long Lines. The curriculum documented in the PDE has been recommended to the Computer Science Curriculum Development Committee, of which the author is a member, and the college administration as the basis for a new major in computer science to be implemented at JCSC during the 1980-81 academic year.

TABLE OF CONTENTS

PRELIMINARY PROGRAM ANNOUNCEMENT

1. Jersey City State College
September 1979

2. Proposed program:

BS in Computer Science

To train computer professionals who will be able to work in business and industry as programmers or analysts, or who will be able to pursue graduate work in computer science.

Numerous studies show that the growth in computer applications will create a large number of career opportunities in the computer field for at least the coming decade. Job opportunities in Hudson County and New York City alone will be more than adequate to employ JCSC graduates. No public four year college in Hudson or Bergen Counties offers a comprehensive program in this field.

3. Projected Freshman Enrollments:

a) First year 30 FTES

b) Fifth year 50 FTES

Date to be offered: February, 1980

4. Faculty Resources:

The college can offer most of the program with current faculty. About 2 or 3 full time positions or the equivalent in visiting or adjunct positions will eventually need to be added.

5. Facilities Resources:

The college has adequate computer resources and laboratories to start the program. Additional facilities will be added as the program grows.

6. Other Resources:

The area is rich in technical expertise. The college will be able to draw advice and capable adjunct faculty from nearby business and industry.

7. The institutional master plan calls for programs with

demonstrated career opportunities for students. The same holds for the state master plan.

8. Two nearby schools offer computer science programs, namely, Stevens Institute of Technology and St. Peter's College. The tuition at both of these schools is appreciably higher than at JCSC and thus they are too expensive for many students. Further, the curriculum at Stevens is more engineering oriented than the proposed curriculum and the program at St. Peter's does not have the same scope.

9. JCSC has made preliminary contact with nearby community colleges and will work to insure the maximum amount of transfer credit for students who have suitable courses.

SECTION I

RATIONALE FOR THE CURRICULUM AT JCSC

A. Introduction

The work leading to the material presented in this document was carried out by the Computer Science Curriculum Development Committee. Valuable input to the committee was provided by way of consultation by Mary J. Clark (JCSC Dean of Arts and Sciences), Magdalena Efros (JCSC Center for Computing Services), Carl Holloman (JCSC Center for Computing Services), Valerie D'Antonio (AT&T Long Lines), George Foley (Bell Telephone Laboratories), Suzanne Hain (AT&T Long Lines), and Anthony Montanaro (Rutgers University). Subject matter expertise required for several course outlines was provided by Bruce Kittinger (AT&T Long Lines), and Cynthia Serrano (AT&T Long Lines). The Computer Science Curriculum Development Committee assumes, of course, full responsibility for the content and recommendations contained in the report.

B. Overall Objectives of the Program

The proposed major in Computer Science is designed to (1) prepare students to go into the field as entry level computer professionals in areas of analysis, design, implementation, validation, and maintenance of computer systems, (2) prepare students for graduate study in computer oriented sciences, and (3) prepare students to teach computer science and computer related mathematics on the secondary school level. For non-majors, the program contains courses designed to provide skills and knowledge sufficient to use the computer as a tool for problem solving in his or her area of specialization. For the general student the program includes introductory and survey type courses covering concepts and applications of computer science in our technological society.

C. Rationale for the Development of the Program at JCSC

Jersey City State College, because of its commitment to serve the urban communities within which it is located, has specifically delineated as one of its objectives the development of professional programs for its students.[2] Students at the college designate adequate career and professional training as their most important reason for electing to come to this college.[3] Deriving as they do from families with multiple socioeconomic disadvantages, our students are anxious to provide themselves with economic security

as a base for improving other facets of their lives. Statistics show that a large percentage of the students at this college come from families with an average income far below the national and state averages. For example 60% of their families have incomes below $15,000 as compared with 34% nationally. More than a third of our students come from ethnic groups which have suffered economic discrimination in the past or from groups which have recently migrated from "third world" countries.[3] Thus, upward economic and social mobility must reasonably be regarded as a prime goal for a majority of our students. In the last five years, the college has added degree programs in Criminal Justice, Fire Safety and Security, Nursing and Business Administration to address student needs in new areas of professional preparation. It now seeks approval for the Bachelor of Science Degree Program to make an additional area of professional preparation available to incoming students. A Bachelor of Science in Computer Science is a readily recognized and respected degree which will enable students from Jersey City State College to compete for jobs as computer professionals and in related fields. Jobs in the computer field are available throughout industry, government, commerce and education.

There are numerous cogent arguments which further support the establishment of a computer science major at this college at this time. Any college which expects to serve the needs of a highly technological society must provide adequate programs in computer science for its students. To deny the college the chance to develope its computer program is to truncate the possibilities of maintaining modern programs in education, the biological and physical sciences, the social sciences, business administration, and criminal justice administration, all of which are becoming increasingly dependent on computer technology. If the major is not established now it will need to be established in the future, since there exists a clear student and curricular demand for it. Indeed, most of the courses in the projected program are already offered with good enrollments. The cost to the college will not be excessive, since only a modest expansion of present faculty and facilities will be necessary to provide for both the computer science program and subsidiary computer oriented minors for other majors. In fact, the point needs to be emphasized that the college will need to expand its computer services, in any case, just to take care of existing demands from other major programs. Thus, no logical reason exists for denying the students at this college the opportunity to graduate with a degree in a field which feeds into a rapidly expanding job market, particularly since technical fields have always been ladders to economic success for disadvantaged youth.

Because of the fact that the college accommodates students from the most urban and industrialized area of the state, it recognizes that the average proparation of a number of these students has been slightly below that of students in sister state colleges or private institutions. Nevertheless, the college also enrolls many students from the local counties, who have exceptional capability, who come here for varying reasons among which are ready access, low tuition, and family loyalty to the institution. Furthermore, a significant percentage of our students are immigrants from foreign countries with excellent mathematical and scientific skills.

D. Demand and Projected Enrollments for the Program

The first computer courses were initiated at Jersey City State College in 1959. By the school year of 1976-77, twenty courses were offered with an enrollment of 470 students. Two years later in Academic Year 1979-80 computer course listings had more than doubled to 45 with a total student enrollment of 1,040. These statistics are similar to those at other state colleges which have already had computer science degree programs approved. If the computer science major is established, a doubling of course enrollments to 2,000 students can be expected within two years.

E. Employment Prospects for Computer Science Graduates

The employment picture for present computer science graduates is very promising. Without the occurrence of a major decline of the United States economy, and specifically the computer industry, which does not always parallel general business and industry expansion and contraction cycles, this employment picture is projected to continue into the decade of of the Nineties.

The employment projections just recently prepared by the State of New Jersey, shows that the occupation of computer specialist, which is comprised of programmers, analysts, and other computer specialists, is expected to grow from 13,300 (a 1979 count) to 19,300 in 1985 in the State of New Jersey; an increase of 45.11 percent. National projections show an even greater increase of 55 percent when total United States numbers have increased to 475,000 in 1985 from 308,000 in 1974.[4]

In a recent (1979) nationwide study done by Dr. John W. Hamblen, of the Rolla School of Mines, he notes that the United States is only producing one-sixth the computer science bachelor's, one-eleventh the computer science master's, and one-fourth the computer science doctorate's

that the nation needs annually. Also, his projections show a need for approximately 116,000 additional computer people each year during the next five to ten years. This figure excludes operators, key entry, and other clerical types.

One very important conclusion that Dr. Hamblen derives in his study, is that the industry is over-utilizing the under-educated computer specialist; i.e., that there are too many one- and two-year vocational and community college graduates entering the work force and not nearly enough people with an undergraduate degree and beyond.

Another barometer of the demand for computer specialists is the classified ad section of each Sunday's edition of of any major metropolitan newspaper. Although this measure of demand is inflated by multiple ads (by personnel agencies) for the same positions, it shows a strong and growing demand, whose end is not in sight.

As stated earlier, barring any catastrophic downturn of the economy, computer science graduates during the next five to ten years will be able to choose from many companies and many positions, and will command a starting salary that is much higher than that of comparable disciplines.

In particular, students graduating with a Computer Science degree from Jersey City State College will have no difficulty finding relevant employment. Direct evidence of this is the fact that in recent samples approximately 80% of those who took the Computer Science Mathematics Specialization offered by the Mathematics Department, obtained positions in the computer field.[5] During the last four years Jersey City State College graduates have obtained positions in nationally known firms including:

AT&T Ling Lines, Bedminster, NJ
Bell Laboratories, Murray Hill, NJ
IBM, Sterling Forest, Suffern, NY
Auxton Computer Enterprises, Edison, NJ
American Telephone and Telegraph, Piscataway, NJ
Manufactures Hanover Trust, New York, NY
New York Life Insurance Co., New York, NY
Insco Corp., Neptune, NJ
Compuscan, Teterboro, NJ
Prudential Insurance Co., Holmdel, NJ
Systems Management, Englewood Cliffs, NJ
Union Carbide Corp, New York, NY
Kidder, Peabody & Co., New York, NY

F. Chronology of Computer Science Curriculum Development at JCSC

Jersey City State College was the first state college in New Jersey to make use of a general purpose digital computer in its curriculum. In 1958 the computer began to be used in connection with the course Numerical Analysis offered by the Mathematics Department utilizing an IBM 1620 located at what was then called the Newark College of Engineering. In 1963 a similar arrangement was made to use the computer facilities at Rutgers University in New Brunswick with students traveling from Jersey City during the evenings and on week-ends. It was during this period that the first discussions occurred between Thomas Mott (Rutgers) and George Daneluk (JCSC) about the possibility of Rutgers providing time sharing services to all the state colleges by means of an IBM 360, for which financial support was being sought from the National Science Foundation. Pursuant to this, Jersey City State College hosted the first meeting of representatives of the State Colleges, Rutgers, the N.J. Board of Higher Education, and IBM to explore the possibility of establishing a computer network for the public institutions of higher learning in the state. See Appendix 6 for a copy of the minutes of that meeting. This initiative, which eventually led to the creation of the N.J. Educational Computer Network, obviously had a profound effect on the subsequent development of academic computing in New Jersey.

Jersey City State College was also the first state college to install its own general purpose digital computer. In 1966 the college acquired a second hand IBM 1620 and other unit record equipment through the efforts of Bob Pecka, Director of the JCSC Center for Occupational Education, in a cooperative project with the Newark Manpower Training Center located on the site of the old Newark State College campus. A member of the Mathematics Department at JCSC put the computer in operation and immediately introduced some new courses making use of it, e.g., Introduction to Computers, a computer workshop for Mathematics Teachers at Woodbridge High School, etc.

At the same time JCSC was an active participant in the ground work going on at the state level leading to the purchase of the IBM 1130's, which were finally installed in 1968. The proximate availability of a computer on campus opened up the possibility for the development of a full curriculum. The Mathematics Department under the chairmanship of Dr. John Reckzech took the lead in introducing new courses in computer science and computer related mathematics, which were patterned after the 1964 recommendations on computational mathematics of the Panel on Computing of the Committee on the Undergraduate Programs in Mathematics (CUPM) and the 1968 recommendations for academic programs

in computer science of the Association for Computing Machinery.[28] In 1973 a Committee on Computational Mathematics was formed in the Mathematics Department under the chairmanship of George Daneluk to develope an undergraduate major program in Computer Science Mathematics based on the 1971 recommendations on computational mathematics of CUPM.[25] See Appendix 2 for a copy of the program description as approved by the Senate Curriculum and Instruction Committee in 1975. This program has been successfully offered in the Mathematics Department until the present time as the Computer Science Mathematics Specialization for mathematics majors. Almost all of the courses making up the Specialization have been incorporated into the curriculum being proposed in this document.

Shortly after the IBM 1130 was put in operation the Physics Department at JCSC under the leadership of Dr. Philip Goldstein also became actively involved in the development of curricula in computer science. In 1972 the Physics Department implemented a program in computer studies designed to provide a subject matter concentration as an option for the general student. The department also developed a set of courses dealing with the hardware aspects of both digital and analog computing devices and courses in scientific computing. To support these and other courses the Physics Department has built up an extensive collection of laboratory equipment including logic circuit boards, micro-processor components, analog computers, hybrid systems, etc. See Appendix 3 for a copy of the program description of the Computer Studies Option. Almost all of the courses making up the Computer Studies Option have been incorporated into the curriculum being proposed in this document.

The courses and programs presently being offered at JCSC have been successful in the important sense that enrollment has steadily increased each year after their inception. For example, the number of sections of the course Introduction to Computers has increased from two sections per year starting in 1967 (one each semester) to an average of 14 sections per year by 1979. The number of sections of the Cobol course has increased from 1 section per year starting in 1976 to an average of 12 sections per year by 1979. An average of one new advanced course per year in computer science has been implemented into the curriculum since 1968, most of which are being currently offered at least once a year. A total of at least 30 graduate and undergraduate courses in computer science have been introduced at JCSC, nearly all of which are being offered on a continuing basis. Computer oriented subject matter and methodology has also been incorporated into the curriculum of nearly all the departments of the college to some extent. The point is that academic computing has exhibited the same potential for growth at

JCSC as it has at every other academic institution in the state. The rate of growth of academic computing in the public colleges and universities of New Jersey generally has been largely determined by the degree of financial support for staff and hardware provided by the state itself.

The proposed curriculum constitutes an effort to consolidate and extend the heretofore separate and uncoordinated development of programs at JCSC into a single curriculum having a degree of consistency, functional independence, and completeness and which conforms reasonably closely to standards contained in recently published recommendations of ACM (1978) and IEEE (1977).[27,30]

G. Project Goals and Methodology

The intent of the computer science curriculum development project is to replace a somewhat haphazard approach to program development and implementation at the college with a more systematic process, which will hopefully result in educational products that are more timely, consistent, complete, functionally independent, and more relevant to the needs of students on the job after they graduate. The process of course and program development will be carried out in the following phases: (1) preproject study and skills and knowledge analysis planning, (2) skills and knowledge analysis with implication for curriculum change, (3) development of instructional objectives, criterion test development, and curriculum design, (4) development of course materials and planning for testing the materials, (5) testing of the materials and planning for the introduction of the courses, (6) course introduction and planning for follow-up evaluation, (7) follow-up evaluation and recommendations for remedial action, (8) remedial action and planning for new skills and knowledge analysis, i.e. go back to phase (1).[1] The products of the activities of each phase will be documented and presented to appropriate committees for review and evaluation. The present proposal provides documentation for phases (1), (2), and (3) using the documentation standards for program approval of the N. J. Board of Higher Education, the JCSC Senate Curriculum and Instruction Committee, and Training Development Standards of AT&T Long Lines.[1]

The overall aim of the Computer Science Curriculum to be developed is to insure the availability of relevant skills and knowledge training in the areas of information systems development, hardware organization of computers, software engineering principles and practice, and mathematical foundations. The responsibility of the Computer Science Curriculum Development Committee include: (1) identifying

state-of-the-art developments needing instructional attention, (2) preparing documentation on state-of-the-art developments for instructional consideration, (3) prioritizing needs for course development and/or maintenance, (4) recommending the development of courses based on skills and knowledge needs assessments, (5) providing expertise in subject matter and methodology. The principle function of the adminstrative structure established to implement the proposed curriculum will be to develope courses based on the needs determined by curriculum committees and offer the courses on as cost effective, timely, and convenient a basis as possible to students.

H. Summary and Conclusions

1. Jersey City State College is located in an urban setting many of whose residents are in the lower economic strata. A primary mission of the college is to provide viable career opportunities for persons who live in Hudson and other nearby counties. During the 1980's thousands of new computer jobs will be created each year as a result of the continued and rapid expansion of this field. Many of these jobs will be available in Northern New Jersey and New York City, areas where most graduates of the college live or work. It is quite logical that the college plan to implement a computer science degree program.

2. The need for a computer science program is highlighted by the numerous requests the college receives regularly from students interested in pursuing a computer science career. Unfortunately, these students go elsewhere or go without because of the lack of a suitable program at the college. In addition, some students are forced to transfer from the college after one or two years because they want a computer science degree. Students who remain at the college persistently ask why we don't have a computer science program. Thus by not offering a computer science degree program, the college is not doing its share in providing adequate opportunities for career development for its students in an important field.

3. Two other nearby schools offer computer science programs, Stevens Institute of Technology and St. Peters College. The tuition at both of these schools is appreciably higher than at JCSC and too expensive for many students. Further, the curriculum at Stevens is more engineering oriented than our proposed curriculum, and the curriculum at St. Peters is not as broad in scope as the proposed program. In any event, there is sufficient demand in the computer field so that a computer science program at JCSC will not adversely impact the programs at these

other schools.

4. In summary, the economics, the demographics, and job market proximity clearly warrant a computer science program at Jersey City State College.

5. "Honest poverty is a gem that even a King might feel proud to call his own, but I wish to sell out. I have sported that kind of jewelry long enough. I want some variety. I wish to become rich, so that I can instruct the people and glorify honest poverty a little, like those kind-hearted, fat, benevolent people do."

Mark Twain, Mark Twain's Travels with Mr. Brown, p. 236.

SECTION II

PREPROJECT ANALYSIS

A. Purpose

The purpose of this preproject analysis is to determine whether or not deficiencies worth solving exist, which can be addressed by a curriculum in computer science. A program should not be implemented merely because it would be "nice" to have. The analysis is intended to identify deficiencies that are caused specifically by a lack of skill and knowledge requiring training and education. The analysis is intended to provide information about the value of solving deficiencies and about the penalties incurred if they are not solved. The documentation is intended to enable review committees to decide whether it will be worthwhile to authorize the next steps in the curriculum development process.[17]

B. Motivation for Approval Request

The motivation for the present computer science curriculum development project is based on the following assumptions: (1) Students at JCSC need to be trained as computer professionals with skills in critical thinking and abstract reasoning. See Appendix 6 for a copy of a survey giving a profile of the student body at the college. (2) A major program in computer science could improve skills and knowledge in rational decision making and problem solving. (3) The harmful effects of poor decisions and the existence of large numbers of unsolved problems is evidence of the need for training and education in these processes. (4) Assuming that alternative ways of doing things exist, and that some give better results than others, it is obvious that a process of rational decision making in systems development resulting in optimal choices would have salutory effects. (5) Failure to make optimal choices is due to a lack of knowledge about how to construct models, how to define criteria with respect to the variables of a model, how to determine constraints, and/or how to conduct a search among alternatives satisfying the criteria and falling within the bounds of the constraints. (6) Although individuals of high intelligence and extensive experience are able to solve problems and make rational decisions on a more or less intuitive level and do it well, there is a need for improved selection of such personnel. The discipline required in the study of computer science is an effective method of selection.[15]

C. Efficiency and/or Deficiency Statement

Conditions: The circumstances under which deficiencies and/or efficiencies in decision making and problem solving occur are universal. The problem of finding the "best" possible alternative is the most fundamental problem of both individuals and societal organizations. Who: Because of the nature of the optimization problem, the individuals and/or societal organizations for whom improved outputs could be anticipated include everyone. Standards: The optimization phase of the decision making process means finding the best alternative relative to some norm. Although norms can be arbitrarily defined, that norm once selected determines the standard. The standard is by definition the "best" relative to the given norm. Deviation/Change: In practical terms the benefits of the rational decision making process may not be reflected in output per se. The benefit of the process derives from the fact that it provides for testing system models against reality, and the data gathered used to alter the model so that solutions can be arrived at by a process of recursive refinement. The process hopefully results in continuous system change and improvement. Methods of Measuring Change: If the relevant variables of a system model are quantitative, the model is a mathematical model amenable to analysis using the full store of mathematical and scientific methods. Cause of the deficiency: The principle reason for the absence of a systematic application of rational decision making and problem solving techniques is the lack of adequate training in mathematically oriented sciences. Penalties/Efficiences: The extreme penalty for failure to solve problems is in the last analysis inability to survive as individuals or as societal organizations (business, academic, governmental, etc.). Efficiencies which might be achieved are obvious. Value of a Solution: The cost to society of poorly functioning systems has been extensively documented. A precise cost/benefit analysis of educational and research programs is difficult to make. However, the price business and industry is willing to pay to secure personnel adequately trained in information systems development is a rough measure of the value of a solution. The Solution: The solution is to develope curricula in mathematical and computational sciences to provide students with skills and knowledge required to enable them to assume decision making and problem solving roles in systems development projects.

D. Developmental Environment

Computer Science curriculum development can be carried out in a variety of different developmental environments. For example, since deficiencies in the specific output of computer professionals on the job can be demonstrated,

specific training should be developed to improve performance in the specific jobs. Because of the obvious difficulty of training for all existing jobs, however, courses should emphasize general principles which are applicable to different job functions.

Job training is also needed in an environment in which new or changed procedures and methods are being implemented. At the same time a computer science curriculum should include subject matter and methodology that may have no immediate application, since practice at times may lag behind theoretical research and may change capriciously in response to economic or political vagaries. A curriculum should strive toward a proper balance between theory and practice.

A new technology clearly requires education and training and, in fact, is an extremely important feature of the environment of curriculum development in computer science. The exploitation of new technologies is at the core of business and industry. However, academic institutions generally have not played a strong role in providing education in an environment where systems are initially being put in operation for the first time.

Undergraduate education in computing has traditionally played a major role in an environment where initial training is being provided for existing job functions or where training is being provided for changed job populations. The appropriateness of the design of a curriculum in this environment requires, of course, a free and open exchange of ideas between academic institutions and business organizations. Professional societies provide a reasonably effective means for this type of interaction and exchange. The professional societies in the field of computing have, in fact, provided curriculum recommendations for academic programs on a periodic basis. Unfortunately or fortunately as the case may be, one effect of these curriculum recommendations is that course development in certain subject areas has become mandatory, a consequence being that curricula end up looking alike.

Finally, the computer itself has opened up an environment for new instructional technologies. Computer science as an academic discipline can be viewed as simply traditional education in science and mathematics using new instructional strategies made possible by the computer.

Although a long range goal of a curriculum in computer science should be to address needs in all the above environments, the assumption of the present curriculum development project is that the principle developmental environment is initial training and education for existing functional roles. The target population consists of students at JCSC interested

in becoming computer professionals. There are, of course, many specialties in the computer field and certification standards have not been universally adopted. The assumption is that a minimum qualification for employment is an undergraduate degree in computer science or mathematics. Members of the student population may be performing on-the-job, especially part time evening students, but most of the full time day students will not be on-the-job.

E. Data Collection Plans

In academia the professor serves in the role of course developer, instructor, course administrator, and frequently as curriculum manager. In any case, the faculty will be required to maintain an up-to-date subject matter expertise in computer science from the literature in the field. The main source of subject matter will naturally be the books and journals of the library. Faculty will also be expected to maintain an awareness of state-of-the-art developments by attendance at colloquia, conferences, etc.

Deficient outputs are by definition the result of decisions which are non-optimal or based on invalid (unrealistic) system models. A data collection plan should investigate penalties associated with low quality decision making in systems design in terms of increased costs, poor service, lost sales, need for overtime, project delays, reduced efficiency, and low quality of product. Poor decisions obviously have effects on personnel involving injuries, lost time, and reduced or impaired morale. Poor decisions in systems design result in unexpected or unwarranted costs, loss of revenue, and inability to compete. There is general agreement that the above problems exist. It is doubtful, however, whether business, academic, or governmental management recognize that the deficiencies are due to inadequate training in problem solving skill and rational decision making techniques.

F. Summary and Conclusions

(1) Although most people are capable of making rational decisions on a more or less intuitive level, few are able to adequately describe a systematic decision making or problem solving process. Deficiencies due to a lack of in-depth knowledge of subject matter and methodology pertaining to good systems development principles and to a lack of skill in applying the principles in "real world" situations exist.

(2) Many business, academic, and governmental organizations take a systems approach to problem solving using personnel in user areas as analysts and designers. The potential benefits of training and education in techniques of systems

development is highest for these groups.

(3) Every societial organization suffers the penalties of poor decision making in systems design. There can be no question as to whether the deficiencies are worth correcting.

(4) An important factor in the cause of the deficiencies is the lack of sufficient skills and knowledges. The reason for the lack of skills and knowledges is the failure to recognize the value of rational, systematic approaches to problem solving and/or a failure to realize the need for education in appropriate subject matter and methodology.

(5) "Thought! You should not try to think. One cannot think without the proper machinery."

Mark Twain, The American Claimant, Etc., "Playing Courier," p. 488.

SECTION III

SKILLS AND KNOWLEDGE ANALYSIS

A. Purpose

The purpose of this phase of the curriculum development project is to determine the skills and knowledges required to perform problem solving functions in computer subsystems analysis and design, and the skills and knowledges, in particular, to be enabled through the curriculum. The analysis is intended to provide the basis for deciding what the post-curriculum, end-of-curriculum, and enabling objectives will be and, in turn, the underlying concept of the curriculum with respect to structure and content. Although academic institutions give lip service to the concept of providing education enabling students to perform on-the-job, the task is extremely difficult for job functions requiring decision making and problem solving skills. The problem derives from the variability of the job inputs, the unpredictability of the job inputs, and the difficulty of describing the job process.

This analysis is not intended to be comprehensive or detailed in nature because of the constraints imposed by the project scope. Data collection and analysis must necessarily be an ongoing and iterative process in the rapidly changing environment of computer technology. Emphasis will be placed on higher levels of activity on the job such as duty, responsibility, function, etc, rather than lower levels, as task details, step-by-step procedures, etc. The analysis is based in large part on data collected in a case study at AT&T Long Lines.[18]

B. Functional Roles

Functional roles define areas of expertise which commonly are required in information systems development, operation, and support. The roles do not equate to jobs. A functional role may be filled by an expert who specializes in just one role, but more commonly many roles will be filled by information systems generalists. Project management must determine how the roles will be filled, considering the existing organization, available resources, the nature and size of the project, and any special requirements.

Functional roles may be grouped into three categories:[31]

1. Development Team
2. Operations and Maintenance
3. Technical Support

1. Development Team

a. Project Management

Responsible for management and administration of development project. Determines overall project strategy. Determines necessary phase products and activities, including support requirements. Obtains project resources such as personnel, facilities, funding, etc. Controls project quality, schedule, and expenses. Responsible for overall direction and ultimate result of development effort.

Application Expertise

Knows existing methods and procedures in application area, including organizational objectives and measurements. Represents user organization in specification of user needs and the information and operational objectives to be met by the system. Assists in data collection efforts. Provides ongoing validation of system requirements and specifications.

System Analysis

Analyzes existing environment to identify opportunities for improvement or problems resulting from inefficient processes or ineffective products. Identifies user needs to be addressed by system and develops alternative system models to satisfy those needs. Defines informational, functional and performance requirements of new system. Defines system boundaries and interfaces.

System Design

The System Design role requires a general knowledge of several disciplines. The role is performed by an interdisciplinary team consisting of PSS Design, CSS Design and other specialists as required. Develops the architecture (overall design for a physical system) needed to support the functional requirements of the system. Allocates system functions to the appropriate resource (people, equipment, software, hardware) in a manner that optimizes total system performance. Analyzes system functions to determine how best to incorporate system control, reliability, and recovery processes.

Data Base Administration

Assists in the planning and design of specific application data requirements. Analyzes those

requirements to determine availability and location of such data. Analyzes the data relationships and usage requirements defined by the developmental team in order to develop relational and usage views of the data for the overall application. Establishes and maintains directories or catalogues of the data at various organizational levels.

Personnel Subsystem Design (PSS)

Designs the task, position, and job structure for the personnel subsystem. Develops the procedural specifications for manual processes. Determines training, workstation, and physical facility requirements for manual positions.

Position Development

Develops procedural instructions and/or supporting information for application and support positions, including performance aids, references, etc. Performs component verification testing of procedures and documents to ensure the position specifications have been met.

Training Development

Develops training specifications, training modules, texts, instructional aids, etc. Develops means for evaluating student performance. Determines training resource requirements, and training sequences and prerequisites. Tests training modules to ensure that they meet training specifications.

Test

Determines overall testing requirements and specifications for the proposed system. Develops test plans and schedule. Sets up test structure and procedures. Designs individual test cases and test data. Maintains test library and documentation. Performs system verification and validation testing, and coordinates the efforts of user and computer operations personnel in performing system certification testing.

Data Communications Design

Determines data communication requirements for application, including interfaces to other applications. Specifies traffic volumes and characteristics, and network performance requirements. Specifies number, location and types of terminals. Evaluates

application requirements with regard to available data communications products and capabilities. Participates in design of operator/terminal interface.

b. Data Center Functions on Development Team

Project Leadership

Delegated overall CSS development responsibility for an application by a data center manager. Plans CSS development, and provides CSS schedules and costs. Coordinates and controls the effort of the CSS team, and serves liaison with data services staff technical support groups. Works closely with project management to ensure that the computer subsystem meets specifications, schedules, and cost projections.

System Design

The System Design role also requires a general knowledge of several disciplines. The role is performed by an interdisciplinary team consisting of PSS design, CSS design and other specialists as required. Develops the architecture (overall design for a physical system) needed to support the functional requirements of systems. Allocates system functions to the appropriate resource (people, equipment, software, hardware) in a manner that optimizes total system performance. Analyzes system functions to determine how best to incorporate system control, reliability and recovery processes.

Computer Subsystem Design (CSS)

Designs the module, program, and job structure for the computer subsystem. Develops the procedural specifications for application programs and modules. Determines hardware and software requirements for the application.

Data Base Management

Assisted by data base adminstration, develops the structural view of the application's data. Designs and structures the physical data base (physical view). Designs, codes, tests and implements the application's logical and physical data base (s). Develops and maintains the data dictionary.

Programming

Develops code for computer execution. Performs component verification testing of code and outputs to ensure that program specifications have been met.

c. Audit Functions on Development Team

Audit Support

Serves in an advisory capacity as required. Provides project support in the identification of system control requirements, procedures, etc. Recommends various means by which system controls might be accomplished.

2. Operations and Maintenance

a. System Management

Manages overall use of the system. Coordinates activities of all organizations involved in system operation. Establishes system priorities and schedules. Monitors total system performance and effectiveness. Coordinates trouble resolution and system recovery procedures. Evaluates and prioritizes system maintenance requests. Coordinates test and installation of new system releases.

Application Expertise

Has detailed knowledge of the application and its operational characteristics. Serves as consultant on application questions. Acts as point of contact for application problems. Diagnoses or participates in diagnosis of problem cause(s). Implements fixes for routine trouble situation or initiates alternative processing procedures. Invokes other functions, as required, for problem resolution.

Training Administration

Schedules and provides application training. Maintains instructional staff and facilities. Evaluates course and student effectiveness. Identifies need for new or modified training materials.

Course Instruction

Conducts application training required by organizations operating or using the system.

PSS Maintenance

Tests and installs personnel subsystem portion of system release. Monitors PSS performance. Performs procedures analysis and identifies need for improvement of methods, documentation, and/or training. Analyzes system change requests to determine development and cost requirements. Coordinates changes with CSS maintenance.

System Use

Utilizes system outputs or system capabilities. May also provide input to system. Responsible for proper use of the system. Identifies system change and improvement requirements.

Position Supervision

Supervises manual work functions. Negotiates media schedules for work unit. Monitors position performance, including quality, proper use of procedures, schedules, productivity, and control measures. Identifies operational problems via trouble report or system change request.

Position Operation

Performs position procedures for application and support position functions, both clerical and management.

b. Data Center Functions for Operations and Maintenance

Data Center Management

Responsible for computer subsystem portion of the system. Coordinates with other data systems groups to obtain/maintain the computer and technical resources needed to satisfy system performance requirements. Coordinates resolution of computer subsystem problems. Coordinates the test and installation of new system releases.

Computer Center Supervision

Supervises Computer Center functions, negotiates Computer Center media schedules. Monitors production work, including quality, reruns, schedules, and control procedures. Evaluates computer subsystem performance and identifies operational problems via trouble report or system change request.

Computer Center Operation

Performs Computer Center Functions:

Input Receipt
Scheduling
Set-up
Computer Operation
Output Control
Distribution
Tape/Disc Library

Data Base Management

Responsible for the integrity and security of physical data, data base conversions, reorganization and recovery of the operational data base. Monitors and optimizes data base performance and resource utilization. Maintains the data dictionary.

Computer Center Technical Support

Monitors hardware/software operation. Assists in problem diagnosis and system recovery. Coordinates with vendor personnel on day-to-day problems. Provides technical support to computer center personnel.

Network Control

Coordinates installation and test of new components and applications. Monitors network performance; identifies failures or degradation. Isolates problem and dispatches appropriate agent for repair. Performs physical network control, e.g., patching, rerouting, etc. Maintains performance data log.

CSS Maintenance

Tests and installs computer subsystem portion of system release. Monitors CSS performance. Performs preventive and repair maintenance functions. Analyzes system change requests to determine development and cost requirements. Coordinates changes with PSS maintenance.

3. Technical Support

a. Departmental Functions for Technical Support

Departmental Data Administration

Implements, monitors and evaluates data policy within the department. Analyzes department's

general data requirements, and directs efforts and policies toward attaining them. Maintains departmental data catalogue.

Application (Data Systems) Planning

Develops long-range application plans. Determines areas for possible mechanization. Evaluates existing and planned centrally developed systems to determine potential use.

User Approval

Evaluates each system under development as to its ability to satisfy user objectives and achieve stated benefits. Evaluates projected operational characteristics of proposed systems. Evaluates overall performance of operational systems and identifies areas for improvement and enhancement of system functions.

b. Data Services Staff Functions in Technical Support

Authorizations
Evaluates the costs, benefits and impact of individual systems. Recommends approval of resource expenditures for system development and major maintenance. Monitors overall developmental and technical direction of project, and initiates project reviews at designated checkpoints during development.

Develops methods, tools, and provides project support for system development, project management, documentation and review procedures.

Plans and directs the development of new or improved methodology to support the data services organization. Administers existing and new data systems which support the administrative management functions of the data services staff, Data Center managers and departmental users.

Standards

Develops methods and procedures to be utilized by the data systems organization. Establishes basic operating and control procedures. Develops new procedures necessitated by changes in the operational environment. Develops methodologies to aid in the performance of data systems functions, e.g., development, maintenance, documentation, operations, etc.

Provides technical advice and consultation to project members on the development of the computer subsystem:

Data Administration

Centrally administers data as a resource. Participates in data requirements planning for the organization's applications. Responsible for policy in areas of data security, performance and usage requirements, recovery and testing of data base systems.

Serves as Data Base Administration for selected interdepartmental data bases.

c. Timeshare Center-Technical Support

Timeshare Support

Administers in-house timeshare systems, including systems access, availability, utilization, trouble resolution, and billing. Identifies needs for improved software and utilities. Assists design/user in effective use of timeshare facilities, via technical support, publications, and training.

Approves and administers use of out-of-house timeshare; processes and charges back out-of-house costs.

d. Communications Function

Internal Data Network Support

Responsible for the design of internal data network configuration. Coordinates with new applications which plan to use the network. Specifies data switcher configuration and identifies application interface requirements. Designs application network. Assists in installation and tests of communication hardware and software. Monitors network performance and **forecasts** growth requirements.

e. Systems Technical Education Center Function

Data Systems Training

Provides training in data systems procedures. Identifies new training requirements, establishes training objectives, and develops or obtains course materials. Coordinates training provided by outside sources. Evaluates course and student effectiveness.

system design, program design, coding and test. Identifies the need for improved procedures and/or training in the CSS area.

Analyzes application requirements to determine need for new software versions or packages. Evaluates software options and selects specific software products. Generates, customizes, tests and maintains system software. Responsible for development and coordination of application conversion and testing procedures.

Evaluates computer center methods and procedures, including performance of work functions, security, disaster and recovery policies, performance indices, etc. Evaluates new tools and procedures that can improve computer center effectiveness. Assists in the implementation of new methods.

Planning and Telecommunications

Develops long and short range EDP plans for mechanization requirements. Plans the distribution of applications, the data center, hardware, software, network, and personnel requirements required for system development, operation, and maintenance.

Determines total hardware requirement across applications and selects the specific hardware to meet those needs, both shared and dedicated. Evaluates new hardware offerings. Analyzes contractual options and selects most attractive means of acquisition. Responsible for hardware test and acceptance. Minitors equipment performance, reliability, and utilization.

Monitors and evaluates the overall utilization of computer resources. Analyzes resource usage data to diagnos hardware configuration problems, design and coding inefficiencies, ineffective or inefficient operating procedures, and utilization trends. Identifies and/or develops performance improvement measures. Assists in implementation of new procedures.

Plans, reviews designs and provides technical support for data communications and terminal hardware. Evaluates and recommends new terminal equipment. Negotiates and administers all general agreements and contracts with outside vendors for teleprocessing devices and timeshare systems.

Provides total hardware and software support for microfilm and key entry.

C. The Systems Analyst Function (Amplification)

The position is responsible for studying and analyzing systems for the purpose of identifying problem/opportunity areas and recommending improvements. In the context of this job description, a system is intended to include the organization, its personnel, and the supporting methods and information systems.

The incumbent in this position is responsible for identifying the functional areas to be studied, gathering information about the system, systematically classifying and displaying that information, stating concisely the real problems and opportunities for improvement that are uncovered, suggesting alternative approaches to change, evaluating the feasibility of each alternative, and recommending the appropriate solutions to management.

The responsibility with the most important impact is that of performing problem/opportunity analysis. This requires an appreciation and understanding of each function within an organization. The systems analyst must be able to visualize and define the present and potential relationships between those functions and recognize when functions do not interact, but should. There are several means the analyst can use to resolve problems and improve systems including organizational change, additional training, new management techniques, new methods, and new information systems.

The purpose of this position is to provide management with the expertise and manpower needed to perform "front-end" analysis services for all disciplines within the organization. This requires a knowledge of the principles of good management and enough knowledge of available techniques and related disciplines to recognize when they can be of use in improving an existing system.

The duties and responsibilities of a systems analyst are to:

1. Study and document the existing system.
    a. Define the scope of analysis. This involves interpreting the facts surrounding the decision to analyze a particular functional area and setting initial boundaries of the system to be studied.
    b. Review the results of previous analysis and any major studies, task forces, or committees that dealt with the areas under investigation.
    c. Collect data regarding all aspects of the system under study using the appropriate information gathering techniques. Techniques include interviews, meetings, observation, questionnaires, etc.

d. Document user objectives that guide the system being studied. This often involves originating a solid definition of the operational objectives, if they have never been documented, or clarifying and quantifying existing ones.

e. Analyze and document functions, methods, and major decision points in the area under investigation. This requires the systematic identification and description of all activities in terms of inputs and their sources, processes, outputs and their destination, timing, initiating condition, and means of conducting.

f. Categorize the functions according to type-strategy, planning, control, contracting, information, etc.- to aid further in-depth problem and deficiency analysis.

g. Evaluate the characteristics of the system as a whole and document problem symptoms and areas where detailed analysis is required.

2. Perform detailed problem/opportunity analysis.

a. Evaluate the effectiveness, validity, and criticality of each function with respect to organizational objectives, structure, personnel, technology, environment, and method of implementation.

b. Determine if there are any functions that are not being performed but should be performed, and document how these new functions relate to existing functions.

c. Analyze the existing information flow and evaluate documents and informal data sources in terms of their effectiveness in supporting the functions of the personnel in the organization.

d. Trace any deficiencies noted to possible causes, compounded causes, or additional problems. Identify relationships between problems and identify those problems that have related causes.

e. Evaluate the effect of any deficiencies in terms of force, control, service, economics, and impact, and determine the value of pursuing their resolution.

f. Clearly state the problems and opportunities uncovered, document findings and recommend the resolution of those problem/opportunities that would most benefit the organization.

3. Develop and describe alternative approaches to improving the existing system.

   a. Define the user objectives that would have to be met in order to resolve the problem or take advantage of the opportunities.

   b. Examine and evaluate existing methodologies or systems that could possibly be used to meet new user objectives.

   c. Outline the alternative approaches and for each determine the appropriate discipline or disciplines (training, organizational development, management, job engineering, systems) to be charged with the problem resolution.

   d. Define the objectives of each alternative and describe how they relate to the user objectives.

4. Evaluate the feasibility of each course of action.

   a. Determine the operational impact of alternatives on the existing system. Particular care should be taken to predict the effect each alternative will have on the functions currently being performed. Ensure that any new jobs envisioned are meaningful.

   b. Evaluate and document the technical feasibility of each alternative. If appropriate, identify opportunities and constraints existing in hardware and software technology which may affect the choice of an alternative.

   c. Estimate the gross costs that would be incurred by each alternative - cost of developing, cost of implementing, cost of operation.

   d. Evaluate and document the direct and indirect benefits to be derived from each alternative and compare with the predicted costs.

   e. Consider long term requirements of the existing system and determine the compatibility of each alternative with the future environment.

5. Select one or more feasible alternatives and document the recommendations.

   a. Document the requirements of each selected alternative. Include the objectives of each solution, the user objectives that each solution supports, a general outline of what is to be accomplished, the disciplines to be charged with the development, any technical considerations, any assumptions and

constraints, and the impact and payoff to the organization.

b. If more than one alternative is documented, outline the major differences and tradeoffs of each.

c. Prepare the final report, and present and promote the recommendations to the appropriate management personnel.

6. Coordinate the implementation of the chosen alternative.

a. Obtain management concurrence in and support of a recommended alternative.

b. Outline a developmental plan including estimates of resources required and possible schedules for implementation.

c. Work with specialists in the recipient discipline to insure that the requirements of the chosen alternative are understood and complete.

d. When the changes and improvements associated with the chosen alternative have been implemented, verify that the problems and deficiencies addressed by the solution have been eliminated and that any new user objectives can be met.

7. Scope and Nature of the Analyst Function.

The Systems Analyst may be the only member or one of several members of a project team organized to study a system. The analyst function is envisioned as a staff position that initiates on-going performance analysis within given functional areas, or that responds to management requests to analyze particular problems or problem symptoms. However the position is implemented, the incumbent must be familiar with the systems approach to problem solving and must be able to gain the confidence, assistance, and support of various management levels. Often, the analyst's investigation and study will cross functional lines and will require an interface with specialists in various disciplines including instructional technology, organizational development, human factors, engineering, EDP systems design, operations research, and management theory.

Performance evolution, problem/opportunity analysis, and the synthesis of alternative solutions require a high degree of imagination, creativity, and a tolerance of uncertainty.

D. The Systems Designer Function (Amplification)

This position is responsible for the detailed description and design of new or improved mechanized information systems. The end product of the system design process is a complete specification for the total system including the documentation of the system's inputs, outputs, data base, manual and mechanized functions, testing criteria, conversion requirements, and training requirements.

The incumbent is given a statement of the system requirements which includes the objectives which must be met by the proposed system, a general outline of what the system must do, and a description of the predicted impact of the system on its users. The Systems Designer must confer and work closely with the Systems Analyst who produced the requirements and the system's users in order to transform the conceptual design into a detailed one for implementation. The design specification should document the components of an information system that most effectively accomplishes the desired objectives.

The design specification is a detailed blueprint of the total information system which includes functions that will be programmed and functions that will be performed by personnel in the user's environment or the environment in which the system will operate. This position develops and documents the logical design specifications for the programmable functions and oversees their physical design and implementation. The documentation for the programmable functions must be detailed enough to allow the Data Services Organization to develop program specifications and related software. All manual functions are completely designed and implemented by the Systems Designer. In addition, the incumbent in this position develops or supervises the development of training needed to acquaint personnel with the system functions they will perform.

This position serves as an active interface between the system's users and the technical organizations involved in the actual construction of the system. The Systems Designer translates the needs of the users, as outlined by the Systems Analyst, into a detailed plan for implementation. This requires a thorough understanding of the users' operational environment as well as a working knowledge of computer technology and system development methodologies.

The duties and responsibilities of a systems designer are to:

1. Review and evaluate the requirements for a new mechanized information system or changes in the requirements of an existing one.

   a. Upon request, aid the Systems Analyst in establishing the technical feasibility of a proposed mechanized system.

   b. Suggest changes to the outline of the proposed system where appropriate and re-establish the system feasibility if changes are agreed upon.

   c. Maintain constant communication with the proposed system's users to insure that the eventual system will reflect the current needs of the users.

2. Define the total system configuration.

   a. Define the boundaries of the proposed system by identifying all the system inputs and their sources, and all the system outputs and their destinations.

   b. Refine and finalize the objectives of the proposed system by stating the detailed performance requirements of the system outputs.

   c. Determine and define the data content and the format of all the system outputs and inputs.

   d. Identify and define all the system functions and the system's internal data requirements. Indicate duplicate functions and those functions that have already been implemented in other mechanized systems.

   e. When required, divide the proposed system into manageable subsystems which have as few information exchange interfaces as possible. Within each subsystem, determine which functions will be performed manually and which will be mechanized.

   f. Re-evaluate the logical, technical, economic, and operational feasibility of the proposed system as its initial configuration becomes finalized. Document the expected impact of the proposed system on other mechanized systems.

   g. Document and review the definition of the total system with the users and verify that their requirements and needs will be met by the proposed system.

3. Originate and document the complete logical design specification for those functions in the total system which will be performed by a computer.

a. Organize the mechanized functions into a hierarchical relationship. Analyze the general information flows to determine the best logical structure for the computer subsystem.

b. Decide how each function is to be accomplished (procedures, algorithms, mathematical formulas, etc.) and document any dependencies that exist among functions.

c. Define and document the data interfaces between functions within the computer subsystem.

d. Identify and design additional computer functions as the need for them becomes apparent. Additional functions can include mechanized interfaces between the proposed system's functions and other EDP systems, as well as mechanized interfaces between the computer functions and those performed manually.

e. Design any mechanized functions required to convert the proposed system into actual operation.

f. Identify functions requiring data communication. Describe the data to be transmitted in terms of volume and performance requirements, and work with corporate communications to establish detailed communications network specifications.

g. Originate and prepare test criteria and develop a test plan to be used in verifying that the computer subsystem meets the design criteria.

4. Design logical data bases.

a. Create a dictionary of all data needed by the proposed system using standard definitions and descriptions if they exist.

b. Interface with any centralized data administration function to determine the extent to which the data needed by the proposed system is already available in other systems. In addition, determine if there is data that this system could provide any other proposed information system.

c. Logically organize data elements and groups into larger units of information (entities).

d. Define the relationships between data entities to clearly document how the user sees the data and how the system will use the data.

e. Document the logical data base specifications and differentiate between mechanized and non-mechanized data bases. The specifications describing data needed by the computer functions will be used to design physical data bases, files, and records and the software required to access them.

5. Insure that the detailed design and implementation of the computer subsystem will meet the logical design specifications.

   a. Work closely with the data services organization to clarify the design specifications if necessary.

   b. Review the hardware and software choices made by the data services organization.

   c. Coordinate with and support the data services organization during programming and testing.

6. Design the manual functions and provide for their successful implementation.

   a. Perform task analysis to determine in detail how each manual function within the proposed information system is to be accomplished.

   b. Perform physical data base design. Decide where and in what format the data needed by the manual functions is to be stored and accessed.

   c. Identify and design additional manual functions as the need for them becomes apparent. Additional functions can include manual interfaces between the proposed system's functions and other EDP systems, as well as manual interfaces between the personnel subsystem and the computer subsystem.

   d. Design any manual functions required to convert the proposed system into actual operation.

   e. Develop and design positions, jobs, and organizations that will meet the requirements of the manual functions.

   f. Determine skill/knowledge, training, performance aids, and human factors engineering requirements for each position and job.

   g. Develop documentation for use in performing the positions and jobs. Design and develop performance aids and tools, as required, for use in performing positions.

h. Develop or supervise the development of training required to acquaint personnel with methods and techniques needed by positions and jobs.

i. Test the effectiveness of the positions and jobs by creating test cases, tracing the flow of logic, and simulating the working environment.

7. Supervise the acceptance testing of the total system.

a. Originate and prepare test cases that will prove the logical correctness of the total system in its operating environment. The criteria developed should also test the ability of the system to meet its performance and quality specifications.

b. Work with the user to supervise the preparation of test data and to analyze and compare the test results against acceptance criteria.

c. Document and recommend any changes required for proper system operation.

d. Insure that any changes are properly integrated into the system and verify that the system has been completely tested before beginning installation.

8. Supervise the installation of the new system.

a. Plan the activities required to make the transition from the previous system to the new system.

b. Insure that the system's users are adequately prepared to accept the new system. Conduct seminars and/or training sessions to describe the impact the new system will have once it becomes operational.

c. Supervise the conversion of all mechanized data bases and create the data required by the manual functions.

d. Insure that all system personnel have been adequately trained in the execution of their positions and jobs.

9. Participate in a performance review of the new system in its operating environment.

a. Compare the actual operation of the system against the original system objectives and performance specifications.

b. Document any aspect of the system's performance that fails to meet the system objectives.

c. Initiate and make changes to the system if failures are a result of inadequate design.

10. Scope and nature of the designer function.

The Systems Designer may be the only member or one of several members of a project team organized to develop a mechanized information system. Depending on the size and nature of the system to be designed, this position may rely on others to perform the more specialized functions of logical data base design or personnel subsystem design. In addition, this position may also be assigned the role of project manager. This would require the inclusion of planning and control functions such as assigning work to be done, and tracking the phase-related activities.

The Systems Designer must interface with a variety of individuals in order to carry out the responsibilities of the position. The interfaces include users, system managers, project managers, programmers and project leaders in a data services organization, corporate communications, data base administrators, and possibly contractors or training specialists.

E. The Programmer Function (Amplification)

1. Project Design, Control, and Supervision

a. Responsible for the development/maintenance of a portion of a project or a small sub-project and the technical decisions associated with the project. Makes design decisions that commit an organization to courses of action with interdepartmental impact, involving substantial expenditures.

b. Responsible for control of a sub-project or of a discrete part of a project, and all design and programming functions including development of hierarchies and walk-thrus, coding, testing, maintaining and documenting of the associated programs as well as system design consultation and review with project leader, client (system manager) and possibly user. May directly supervise one or more subordinates and/or direct and assign work on a project basis.

c. Appraises the work of subordinates, participates in feedback sessions, and recommends upgrades, downgrades and dismissals. As to personnel directed on a project basis, makes recommendations relative to the employee's work performance evaluations.

Schedules subordinates, including the authorization of extra work time as required, and including training.

d. Responsible for preparing and meeting resource, time and budget estimates as well as tracking status for the assigned portion of the project as input for the project leader.

e. Estimates resource requirements, both processing time and data storage, for the sub-project or portion of a project.

2. Data Systems Skills and Responsibilities

a. Analyzes client/project leader specifications, resolves questionable areas, converts them to programs and modules utilizing design hierarchies.

b. Creates program/module design specifications according to specified standards in consultation with a project leader or client devising appropriate processing flows.

c. Develops and implements test plan, assuring proper program interface, system integration, and system integrity.

d. Assists in the decision of a file organization for efficiency, including specification of access methods, file types, and internal organization of data.

e. Generates and/or directs the generation of proper documentation.

3. Performs and/or directs coding, testing, and debugging required for developing/maintaining a sub-project or portion of a project in the programming environment using the following knowledge:

a. At least one programming language.

b. Structured programming techniques.

c. Detail logic diagrams, such as IPO and hierarchy charts.

d. General knowledge of the operating environment in which assigned programs or processes are to be run and the job control language.

e. Understanding of standard access methods.

f. Use of testing and production libraries.

g. Timesharing for development and testing.

h. Efficient coding techniques.

i. Experience with module and job documentation.

j. Understanding of basic utilities applicable to the operating environment.

k. Production conversion procedures.

l. Procedures for interfacing with other data centers, where applicable.

m. General knowledge of basic software packages available in the environment (i.e., performance aids, file management system).

n. Understanding of local testing procedures.

4. Demonstrates a command of the programming environment used in maintaining computer processes in a computer installation and, in relation to a particular project or application, has additional skills to:

a. Conduct and/or review coding walk-thrus.

b. Adapt existing programs and routines to perform the required processing or to operate in a different operating environment.

c. Investigate new technical developments and determine their effect on the processing environment of the installation as well as the application.

5. Assists in the technical development of personnel through task assignment, on-the-job training, and recommendations for formal training.

F. Summary and Conclusions

1. The basic inputs to personnel performing systems development functions on the job are problems to solve. The range includes problems with more than one solution, problems with unique solutions, and possibly problems having no solution.

2. The outputs required of personnel are solutions to the problems.

3. The general procedure followed by job incumbents is (a) to construct a model (b) to determine criteria

(objectives, requirements) with respect to the variables of the model, (c) to determine the constraints, and (d) to optimize among the alternatives satisfying criteria and constraints. The general process is the process of rational decision making.

4. Personnel require skills and knowledge in performing logical processes in the abstract and in constructing concrete realizations.

5. The standard for acceptable performance is proven ability to build optimally functioning systems.

6. Personnel need skills in critical thinking.

7. Personnel need skills in verbal communication.

8. Personnel need state-of-the-art technical knowledge.

9. Personnel need practical experience.

10. "Soap and education are not as sudden as a massacre, but they are more deadly in the long run." (Mark Twain)

SECTION IV

CURRICULUM OBJECTIVES AND DESIGN

A. Purpose

The skills and knowledge analysis documented in the preceeding phase constitutes a basis for formulating specifications of performance required of students on the job as computer professionals after they graduate. These performance specifications constitute the post-curriculum objectives of the proposed curriculum in computer science, which in turn, determine the end-of-curriculum objectives, and enabling objectives as well as the criterion tests. The purpose of this phase of the curriculum development project is to define the objectives and to design a curriculum to achieve them.[9]

The purpose of this phase of the curriculum development project is also to document some of the constraints on (1) criterion tests; i.e. tests which can be used to evaluate the attainment of the end-of-curriculum objectives, and (2) the curriculum design. The constraints fall into the following categories:

1. Degree of Simulation

   Is a particular degree of simulation mandated by college governing bodies? Does the degree of simulation have to match other curricula at the college or other colleges? Does the skills and knowledge analysis specify a given level of simulation? Is sufficient hardware available for the required level of simulation? To what degree can the work environment be simulated?

2. Course developer resources

   Are there enough course developers? Are they qualified?

3. Time

   Does completion date impose developmental constraints? How much time has been allocated to complete course development? Is there a fixed constraint on course length and curriculum length?

4. Target population

   Is there a wide range in student ability? Has one or several target populations been identified? Does the target population have any special characteristics other than lack of skills and knowledge? Is there a problem with the size of the target population? Is

there a problem with the distribution of students over time?

5. Course stability

   To what extent are jobs, job functions, skills and knowledge requirements stable or changing? Is the documentation stable or changing?

6. Budget

   Is the development budget sufficient to carry out all design and development activities?

7. Form of training

   Has a particular form of training been mandated (e.g. instructor led, self-paced, etc.)? Are certain media choices sufficient (e.g. audio-visual, library)? Are sufficient facilities available to accomodate the required hardware, students, etc.? Are there wide variations in job environments, skill and knowledge requirements, etc.?

8. Instructor and/or administrator resources

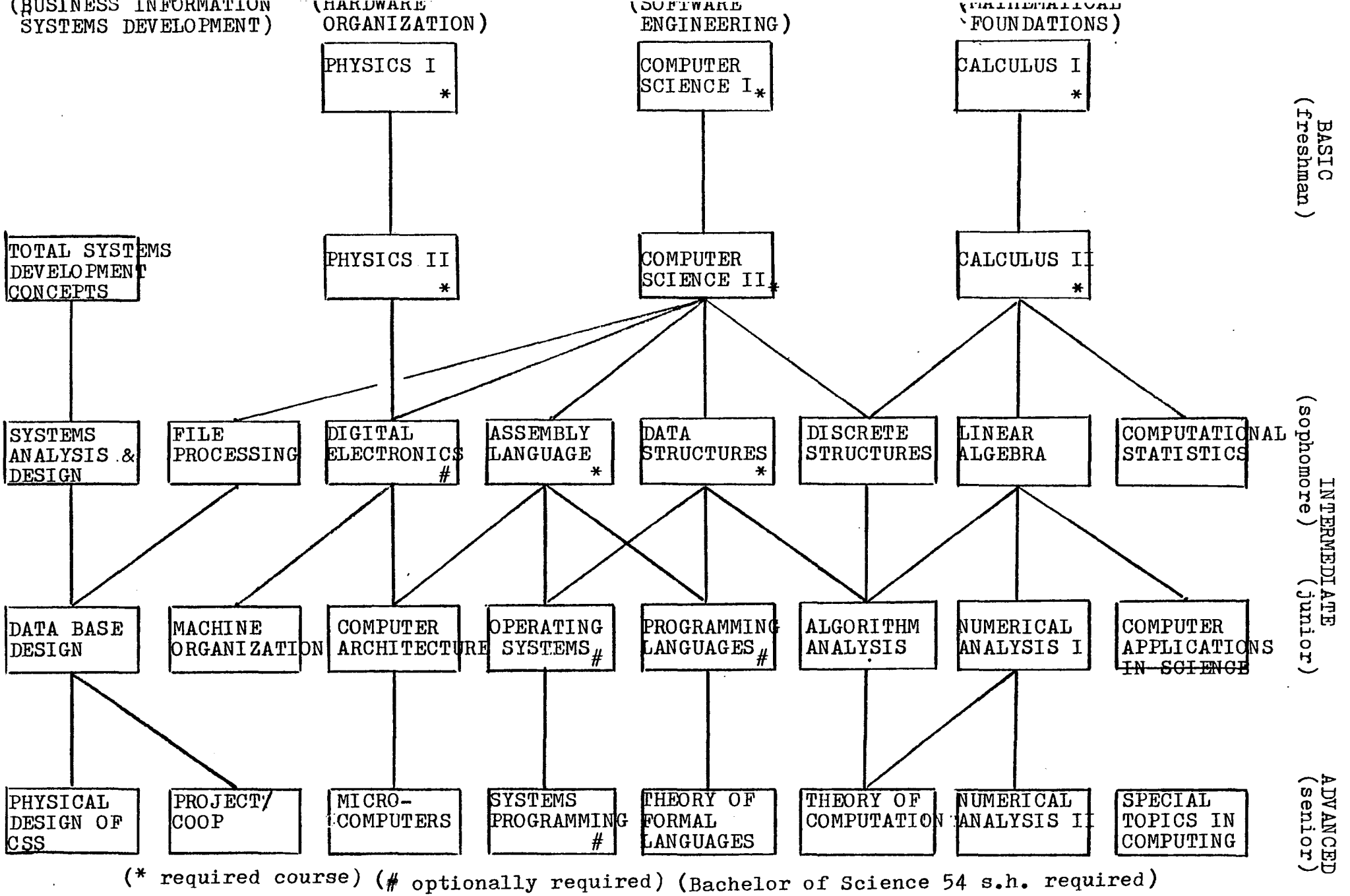
   Are the needed resources available? Do they have the necessary background or qualifications?

The documentation is not intended to be detailed in nature but rather sufficient to enable appropriate review committees to decide whether the curriculum project should begin the next phases of the development process.

B. Overall Curriculum Design[13]

A structure chart showing the courses recommended for the proposed Curriculum in Computer Science is given in Figure 1. The structure chart provides an overview of the proposed curriculum indicating course groupings, pre-requisite lines, and overall structure. The total program is designed to provide sufficient background for (1) a Computer Science major, (2) a Computer Science minor, (3) students majoring in other areas (Mathematics, Science, Business, etc.) who are interested in a working knowledge of the computer as a problem solving tool, and (4) students interested in an introduction to computers, computer applications, and the role of computer technology in society.

The courses are organized into four interrelated components, namely, courses in Business Information Systems Development, courses in Hardware Organization, courses in Software



(* required course) (# optionally required) (Bachelor of Science 54 s.h. required)

Figure 1

Engineering, and courses in Mathematical Foundations. Courses within each component are stacked to indicate the relative level. The lines are prerequisite lines which, however, should be viewed as tentative. If a student has not met specified prerequisites, he or she should consult the individual instructor before registering for a course. Course Guides giving brief descriptions of each course in the curriculum are shown in Appendix 1.

Although the four components are intended to reflect several distinct views of computer science as a science, the subject matter content of the courses in each component permeates the total curriculum; the components are not intended to be functionally independent modules. The distinction between the hardware and software features of computing systems is reflected in the Hardware Organization component and the Software Engineering component, respectively. The Business Information Systems Development component, although properly an integral part of Software Engineering, is intended to reflect the distinction between practical applications and the principles of computer science. The Mathematical Foundations component is intended to emphasize the theoretical aspects of computer science as opposed to the more applied aspects of the other components. The characteristics that distinguish the components should not obscure the obvious overlap between them and the interdependence of the subject matter and methodology. And while the separate components allow for a degree of specialization for a computer science major, they should not be viewed as separate tracks. Also while the total curriculum does not pretend to cover all facets of computer technology, it is intended to provide a balanced and solid foundation in the major conceptual views of computing. The attempt to structure the subject matter and methodology to reflect a balance between principle and practice, theory and application, the abstract and the real, software and hardware, the logical view and the physical view, is in fact, the unifying concept binding the courses in the curriculum. The assumption is that this is a good strategy for providing students with the generalized skills and knowledges required in critical thinking, logical reasoning, scientific research, and rational decision making.

While curriculum guidelines of professional societies contain strong recommendations with respect to supporting work in mathematics, many implementations of curricula have de-emphasized the mathematics component. A look at college catalogs shows that many programs in computer science overemphasize the strictly business data processing aspects. Research oriented schools usually emphasize the theoretical aspects. Engineering schools for obvious reasons emphasize the hardware aspects of computing systems. Many programs overemphasize the strictly programming aspects of computer science. Some emphasize the view of the computer as simply

a tool to be used in other disciplines. The proposed curriculum on the other hand, is an attempt again to provide a balance in four major conceptual areas with emphasis, if any, on the view of computer science as a science. It is unpretentious in scope, yet it has substantial depth in foundations. The assumption is that computer science is a unique discipline in its own right and provide the best kind of training for developing problem solving skills.

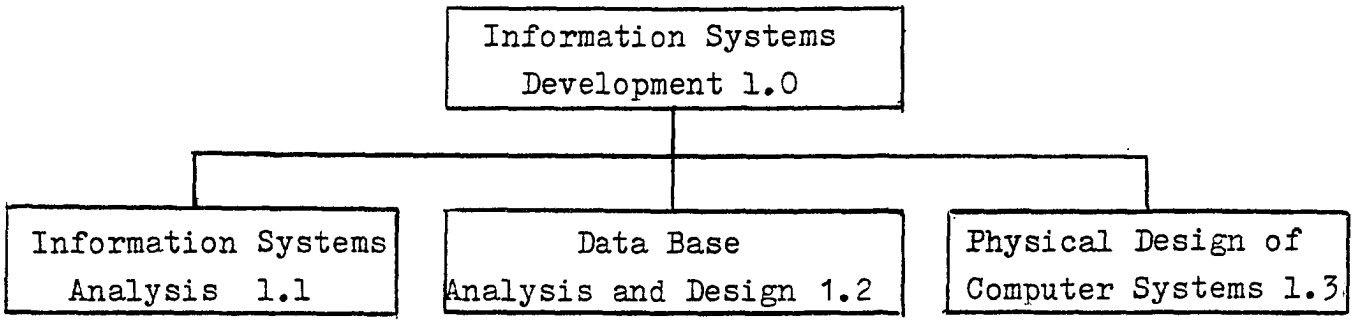
C. The Business Information Systems Development Component [12]

1. Objectives

The purpose of the Business Information Systems Development component is to provide the student with an understanding of the total life cycle of information systems development in a business environment. The sequence of courses would constitute a component of the Computer Science major and a component of a minor in Information Systems Development for the Business major.[30]

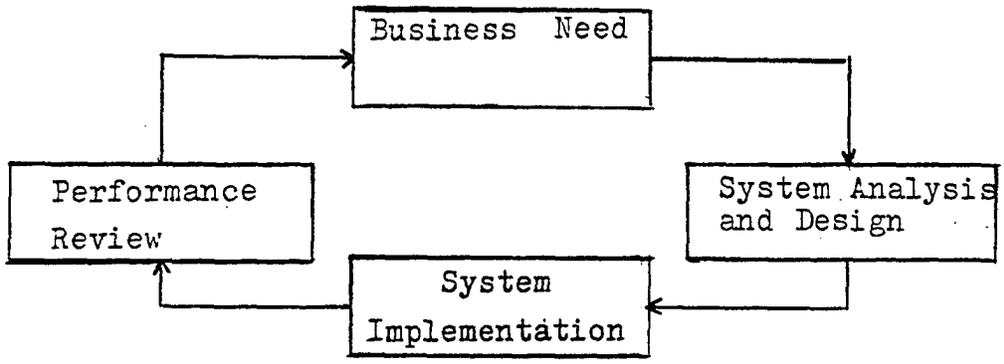
2. Content

The basic modules for the curriculum are shown in the structure chart below. Each module represents one semester length course covering concepts and methodology in the phases of business systems development.[6]



Module 1.0 will cover topics in total systems development concepts. Module 1.1 will cover topics in project management, structured systems analysis and data gathering. Module 1.2 will cover topics in data base concepts, logical data base design, data base management systems and IMS total data base. Module 1.3 will cover topics in IMS implementation, module design, structured design,

file design and computer subsystem design. Topics will be selected and sequenced to show the relationship of information systems development to the need to solve business problems and to take advantage of business opportunities. A schematic representation of the life cycle of the systems development process as related to business needs is shown below. The courses in the Business Information Systems Development are: Total Systems Development Concepts, Systems Analysis and Design, Data Base Design, Physical Design of Computer Systems, and File Design.



D. The Mathematical Foundations Component

1. Objectives

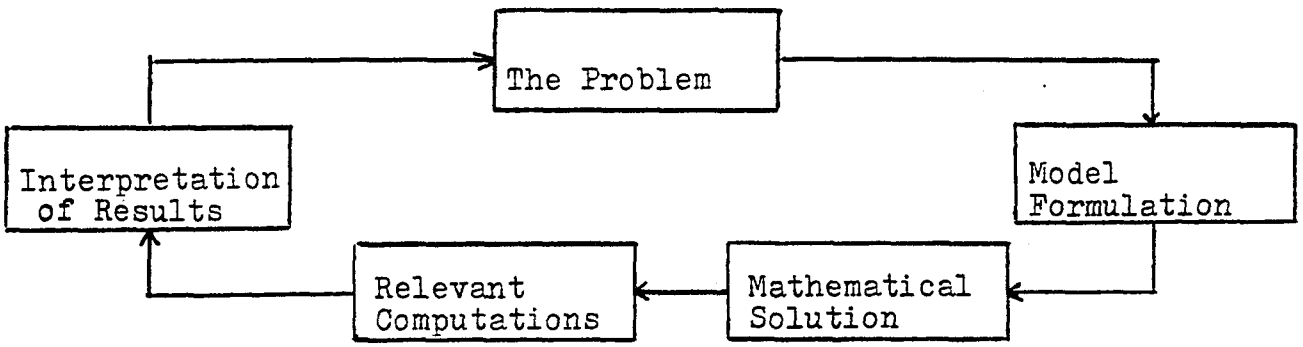
Classical mathematics in such areas as numerical analysis has become quite relevant today, because of the availability of high speed computers. Moreover, in recent years there has occurred major advances in the theory of computation, the theory of formal languages, automata theory, and algorithm analysis, which provide a theoretical framework for approaches to practical applications. The intent of the component in Mathematical Foundations is to provide this theoretical background. The purpose is to give the content of the total curriculum structure and to make it systematic. The courses in the component are intended to set the tone for the program and provide a framework and notation for understanding and expressing the theoretical principles of computer science. "Choose a notation and let it do your thinking for you." (Alfred North Whitehead. The component is also intended to provide the curriculum with a degree of stability, since computer technology is

a rapidly changing field.[23] While specific skills become obsolete, principles on which the skills are based continue to be relevant. The component is also intended to provide the basic problem solving mentality needed to perform in the functional roles identified in the skills and knowledge analysis.

The overall objective is to enable the student to perform the steps a working mathematician follows in studying a given situation:

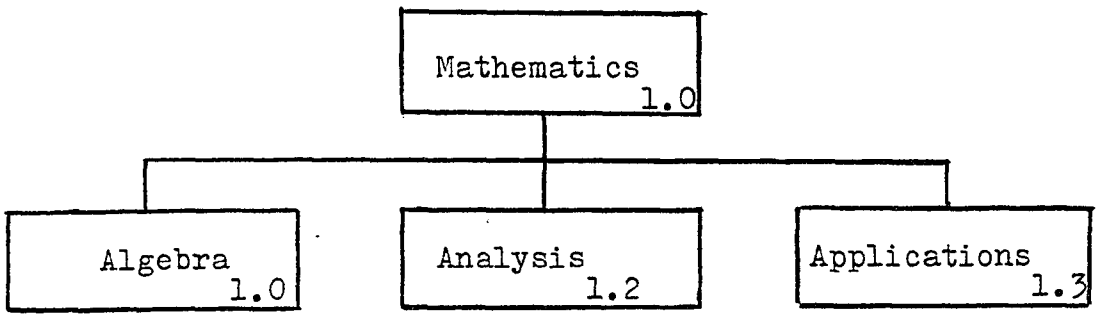
a. Recognition of the nonmathematical problem.
b. Formulation of a mathematical model.
c. Solution of the mathematical problem
d. Perform the relevant computations.
e. Interpretation of the results in the context of the original problem.[24]

A schematic representation of the problem solving life cycle is shown below:



2. Content

The basic modules for the curriculum are shown in the structure chart below. Each module represents two or more courses.



Module 1.1 will cover topics in discrete mathematics including algebraic systems, combinatorics, discrete probability, graph theory, formal language theory, etc. Module 1.2 will cover topics in the calculus, numerical analysis, algorithm analysis, theory of computability, etc. Module 1.3 will cover topics in the theory of optimization, game theory, statistics, etc. The courses in the Mathematical Foundations component are: Calculus I and II, Linear Algebra, Discrete Structures, Data Structures, Computational Statistics, Algorithm Analysis, Computer Applications in Science, Numerical Analysis I and II, Theory of Computability, and Theory of Formal Languages.

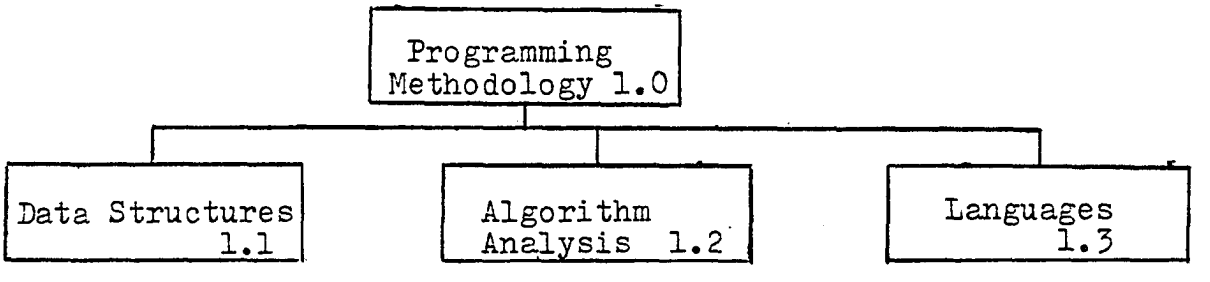
E. The Software Engineering Component [7,8,9,10,11,14]

1. Objective

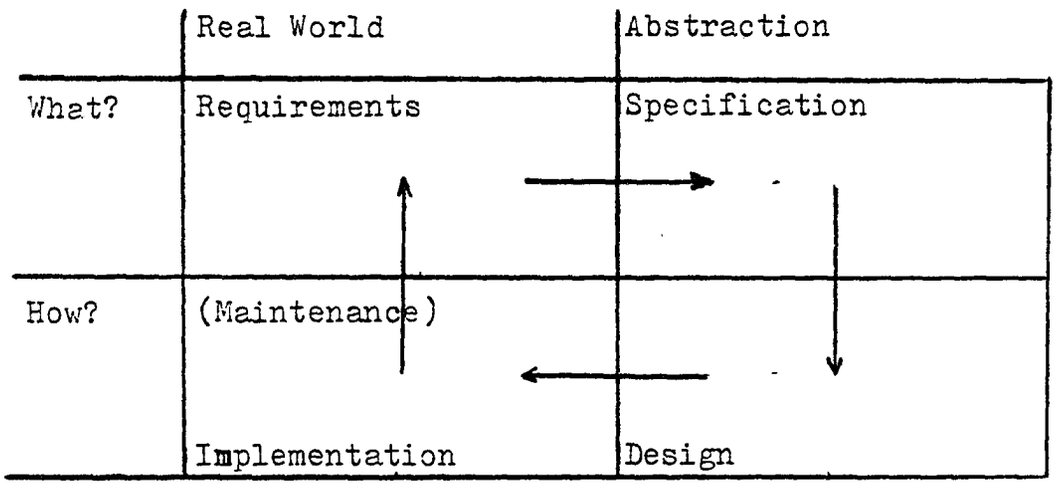
The Software Engineering component is intended to reflect an emerging discipline in the field for managing the development of computer systems.[20] The objective is to enable a student to (a) use techniques that manage software complexity, (b) increase software reliability and correctness, and (c) use techniques to estimate costs more accurately. The courses are intended to provide a student with an understanding of the phases of the software development life cycle, namely, (a) requirements, (b) specifications, (c) design, (d) coding, (e) testing, and (f) operation and maintenance. The objectives are similar to the objectives of Business Information Systems Development component, but with emphasis on elements and general principles.

2. Content

The basic modules for the Software Engineering component are shown in the structure chart below. Each module represents two or more courses.



Module 1.0 will cover topics in structured programming, design tools, design strategies, etc. Module 1.1 will cover topics in data types, data structures, etc. Module 1.2 will cover topics in algorithm analysis and design. Module 1.3 will cover topics in programming and design languages, operating systems, formal languages, etc. The diagram below gives a characterization of the software development life cycle. The courses in the Software Engineering component include: Computer Science I and II, Assembly Language, Data Structures, Programming Languages, Operating Systems, Systems Programming, Theory of Formal Languages, and Project-Coop.



F. The Hardware Organization Component[16,21,22]

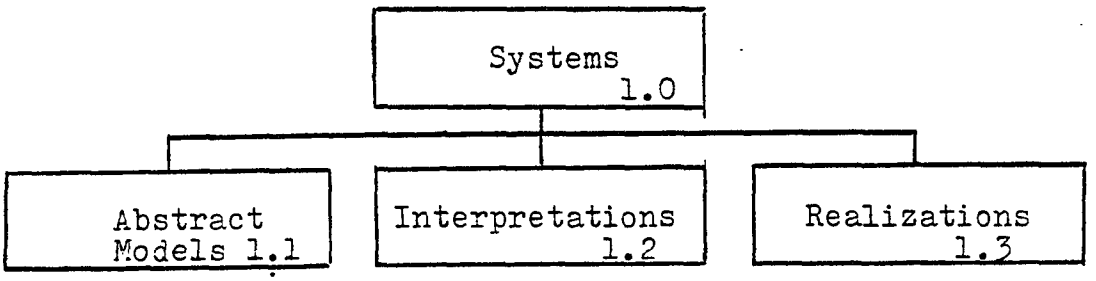
1. Objectives

The intent of the Hardware Organization component is to enable the computer science major to understand the the basic hardware features of digital devices in terms of elements, networks, systems, and computation. The primary task of an engineer is the construction of a system which accomplishes a required task and meets specifications and constraints. The steps in the overall task are to (a) select all the appropriate components, (b) connect the components in some "network" or system, and (c) test the system to verify that the objectives and specifications have been met. The purpose of the Hardware Organization component is to enable a student to perform such a task using the Abstract-Force method as opposed to a Physical-Brute-Force method of attack on the problem. Whenever two or more physical elements

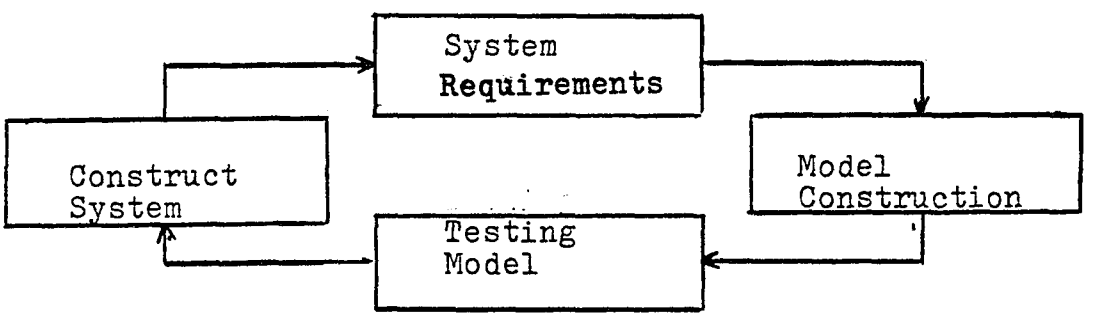
are interconnected, an overall system is obtained whose response is determined by (a) the characteristics of each element of the system, and (b) the manner in which the elements are interconnected (the topology of the network). The intent of the curriculum is to enable a student to design such systems by constructing mathematical models of the system and if the predicted mathematical response interpreted in physical terms does not meet posed requirements, to modify the elements and/or topology of the model until they are met. (Note the similarity of the process to that of the other components.)

2. Content

A structure chart of the Hardware Organization Component is given below.



Module 1.1 will cover topics in Boolean Algebra, the state model, finite state automata, the Turing machine, etc. Module 1.2 will cover topics in switching circuits, network analysis, etc. Module 1.3 will cover topics involving realizations having relevance to digital and/or analog computation. The courses in the Hardware Organization component include: Physics I and II, Assembly Language, Digital Electronics, Machine Organization, Computer Architecture, and Micro-computers. A schematic diagram of the Abstract-Force method of design is given below.



G. End-of-curriculum Objectives

The total curriculum in Computer Science is designed to (1) prepare students to go into the field as entry level computer professionals in areas of analysis, design, implementation, and operation of computer systems, (2) prepare students for graduate study in computer oriented sciences, (3) to prepare students to teach computer science and computer related mathematics on the secondary school level. For non-majors, the program contains courses designed to provide skills and knowledge sufficient to enable the student to use the computer as a tool for problem solving in his or her area of specialization. For the general student the program includes introductory and survey type courses covering concepts and applications of computer science in our technological society.

The models for the proposed program in Computer Science were (1) the 1978 Recommendations for the Undergraduate Program in Computer Science, a Report of the ACM Curriculum Committee on Computer Science, (2) the 1977 Recommendation for a Curriculum in Computer Science and Engineering, a Committee Report of IEEE, and (3) the curriculum for Systems Development Training and Advanced Programmer Training of the System Technical Education Center of AT&T Long Lines.[27,30]

The specific objectives of the major in Computer Science are:

1. to enable students to write readable, well documented programs that execute correctly,
2. to enable students to develop well organized and efficient procedures for solving problems,
3. to provide students with a variety of tools for solving the different classes of problems and the ability to recognize classes that are amenable to computer solution,
4. to provide an understanding of computer hardware organization and architecture,
5. to provide students with an understanding of the total life cycle of information systems development,
6. to enable the student to perform the steps in the process of information systems development in a systematic and disciplined fashion,
7. to provide a sound theoretical foundation in computer science as well as practical experience,
8. to provide a core of courses necessary for further specialized study in computer science.

The enabling objectives are given in the course descriptions in Appendix 1.

H. The Bachelor of Science Program in Computer Science

1. Major requirements

A student majoring in Computer Science must complete a minimum of 54 semester hour credits of course work in the curriculum. The required courses in the major are:

CS 101	Computer Science I
CS 102	Computer Science II
CS 201	Assembly Language
CS 202	Data Structures
MATH 105	Calculus I
MATH 106	Calculus II
PHYS 105	Physics I
PHYS 106	Physics II

And any two of the following:

CS 207	Digital Electronics
CS 302	Operating Systems
CS 305	Programming Languages
CS 403	Systems Programming

A student must also take at least three 400 level courses. Substitutions from the elective component or courses offered by other departments are permitted but only with the approval of the student's major advisors. The remaining courses of the curriculum are elective. Every student is strongly advised to register for the Project-Coop course in the senior year (see course description).

2. General requirements

A major in Computer Science must satisfy the general requirements for an undergraduate degree at JCSC as set forth by the School of Arts & Sciences. These include a total of 128 semester hour credits, 36 of which make up the General Studies requirement, 54 of which make up the maximum for a major, and the remainder of which are of free electives. A student is advised to choose General Studies courses and free electives in a manner which will supplement work in the major. Since computer professionals work in environments involving people from many different disciplines, the student should select courses in the humanities and arts in a manner which will give him or her a breadth of experience, as well as depth in his or her specialty. A student is especially advised to take courses which will help develop good reading, writing and speaking skills. Since

computer science is essentially a mathematical science, a student is strongly urged to take courses designed to strengthen his or her knowledge of mathematics.

I. Suggested Four Year Student Program

Freshman Year

Fall	Spring
1. Math 105 Calculus I	1. Math 106 Calculus II
2. CS 101 Computer Science I	2. CS 102 Computer Sci. II
3. Physics 105 Physics I	3. Phys. 106 Physics II
4. Eng. 101 Fund. of Communication	4. Eng.102 Fund.Communication
5. (General Studies Elective	5. (General Studies Elective)

Sophomore Year

Fall	Spring
1. CS 201 Assembly Language	1. CS 202 Machine Organizatio
2. CS 203 Discrete Structures	2. CS 202 Data Structures
3. CS 103 Total Syst. Dev. Concepts	3. CS 205 Computational Stat.
4. (General Studies Elective)	4. (General Studies Elective)
5. (General Studies Elective)	5. (General Studies Elective)

Junior Year

Fall	Spring
1. CS 307 Numer. Analysis I	1. CS 302 Operating Systems
2. CS 305 Programming Lang.	2. CS 303 Data Base Design
3. CS 304 Comp. Architecture	3. (General Studies Elective)
4. (General Studies Elective)	4. (Free Elective)
5. (General Studies Elective)	5. (Free Elective)
6. (Free Elective)	6. (Free Elective)

Senior Year

Fall	Spring
1. CS 401 Project-Coop	1. CS 407 Theory/Computability
2. CS 406 Theory Formal Lang.	2. CS 403 Systems Programming
3. (Free Elective)	3. (Free Elective)
4. (Free Elective)	4. (Free Elective)
5. (Free Elective)	5. (Free Elective)

J. Articulation

Students transferring from other accredited institutions must follow the procedures for evaluating their transcripts for equivalences set up by the Center for Academic Advisement. In order to receive credit for courses in computer science, the student must submit to the department chairperson a request for evaluation. The student is required to provide a transcript of the academic record, a catalog of the transfer institution, and a list of text books used in courses being transferred. A graduate of any two-year community college in New Jersey will generally be given credit towards the major for all credits earned.

Special programs will be offered to high school students in the Hudson County area. For example, to advertise the opportunities available through the college, a Saturday Course in Computer Related Mathematics for high school students has already successfully been offered and will be reinstituted. See Appendix 5 . Students already enrolled at the college who have insufficient background to immediately undertake the program will be carefully advised in a program designed to remedy deficiencies and to prepare them as quickly as possible to take on the rigors of the curriculum.

K. Computer Facilities

A. Hardware and Software

The Center for Computing Services of Jersey City State College provides both batch and interactive computing services through an IBM 1130 Computing System and nine terminals linked to an IBM 370/158-168 configuration. The hardware and software services are maintained by the New Jersey Educational Computer Network Inc. (ECN), the organization providing academic and administrative computing services for most of the public colleges and universities in New Jersey. Jersey City State College is an active participant in ECN and currently an extensive user of its hardware and software facilities. Gaining familiarity with the hardware organization and and software capabilities of the systems supported by ECN is one of the objectives of the proposed program. Jersey City State College is also presently in the process of acquiring both mini-computer and microcomputer capability and additional data entry devices. Equipment on hand includes:

1. One IBM 1130 computer currently being used as an RJE station to the New Jersey Educational Computer Network. A mini-computer replacement is expected sometime in the near future. In addition to supplying elementary time

sharing locally, the new computer will support two printers and two card readers. One printer will be reserved for student use.

2. Six time sharing terminals for student use. Three more are on order. Additional terminals will be acquired in the future as usage increases.

3. Three terminals for faculty use.

4. A number of special terminals are on order. These include graphics terminals and intelligent terminals.

5. Five key punches presently are available for students. Three additional key punches are on order.

6. Microprocessors and digital curcuit test assemblies are available to support the hardware aspects of the curriculum.

7. Personal computers will be acquired as soon as state regulations permit.

L. Library

The Jersey City State College Library already possesses significant holdings of periodicals and books in computer science and mathematics. But in order to expand the resources available, the library has been requested to implement the library list recommended by ACM and IEEE for institutions having undergraduate programs in computer science, computer engineering, and information systems development. Adequate manuals and documentation for programs are available through ECN and the college computer center.

M. Faculty and Staff

A list of members of the faculty and staff of Jersey City State College teaching computer oriented courses and their subject matter specialization is given below. Courses in the basic mathematics component are taught by members of the Mathematics Department at large. (See the College Catalogue.) Three additional full time faculty, whose primary training and experience is in computer science, will be hired over the next four years. Adjunct faculty from business and industry will be used to immediately teach some of the specialized courses and sections of service courses.

Harold F. Carney; Full Professor, Mathematics Department; B.S. (Saint Peter's College), M.A. (Seton Hall University), Ph.D. (New York University); combinatorial analysis and discrete probability theory, mathematical programming,

business data processing.

Philip W. Caverly; Assistant Professor, Mathematics Department; B.S. (Stevens Institute of Technology), M.S. (Seton Hall University); Ph.D. (New York University); mathematical analysis, differential equations, analog computing, modeling and system simulation, approximation theory.

George Daneluk; Associate Professor, Mathematics Department; B.A. (University of Wyoming), M.A. (Rutgers University); Ph.D. (Union Graduate School); numerical analysis, graph theory, Boolean Algebra, machine organization, statistical analysis, data structures, methods of optimization, theory of formal languages, theory of computability.

Gerald Derman; Assistant Professor, Mathematics Department; B.A. (Rutgers University); M.A. (Rutgers University); mathematical analysis, algebraic structures, statistical analysis.

Magda Efros; Computer Center Staff Member; B.A. (University of Romania); M.S. Engineering & Applied Science (Harvard & Radcliffe); M.S. (Pending at N.J. Institute of Technology); modeling and simulation, scientific programming, educational systems, operating systems.

Gloria Gardner; Computer Center Staff Member; B.A. (Jersey City State College); M.A. (N.J. Institute of Technology); hardware organization, programming languages, educational systems, mathematical programming.

Philip Goldstein; Full Professor, Physics Department; B.S. (City College of New York); M.S., Ph.D. (Carnegie Mellon University); computer applications in physics, modeling and simulation of physical systems, computer assisted instruction electronics and switching theory.

Carl Holloman; Director of Computer Services; operating systems and systems programming, educational systems, business information systems development.

Teresa C. Michnowicz; Associate Professor, Mathematics Department; A.B.,M.S. (Rutgers University); computational methods in linear and abstract algebra, numerical analysis, computer graphics, data processing and Cobol programming, APL.

John Raines; Assistant Professor, Mathematics Department; B.A. (Jersey City State College); M.A. (Columbia University); Ph.D. (Pending at Columbia University); mathematical analysis, discrete structures, computer aided instruction, algorithm analysis.

Richard F. Riggs; Associate Professor, Mathematics Department; A.B. (Knox College); M.A., Ed.D. (Rutgers University);

combinatorial analysis and probability theory, mathematical programming, computational methods in linear algebra, computer assisted instruction, mathematical modeling, a applications of statistics to the social sciences, mathematical statistics.

N. Evaluation Plan

The Computer Science Department will continually monitor and revise the program as deemed necessary to insure currency of course content, proper structural relationship of courses, and program quality. Outside consultants will be employed at regular intervals in order to provide input on such matters as:

1. Program structure--does it provide students with an adequate background for work in industry and graduate school?

2. Program quality--is the program sufficiently demanding of students?

3. Computer facilities--is the physical plant adequate? Are the computer services adequate? Is there sufficient hardware on hand?

In addition, a number of faculty members are acquainted with experts from industry and various universities who are willing to provide informal input on curriculum development. The advice of such experts will be extremely helpful in maintaining the quality and relevancy of the curriculum.

All programs at JCSC also periodically undergo evaluation for purposes of accredition. The Curriculum in Computer Science will be subjected to the same evaluation procedures. The computer science programs will, of course, be evaluated in terms of stated objectives, namely, job placement and admission to graduate school. The criteria for job placement shall be 75% of the graduating seniors being employed as entry level computer professionals. The criteria for admission to graduate school shall be all students in the top 20% of their class who apply being accepted. The placement office of JCSC each year conducts a survey of the job placement experience of the previous years' seniors. This will be the primary source of data for evaluation of the employability of the seniors. The primary source of data for admission to graduate school will be follow up by faculty writing letters of recommendation.

O. Diversion of Resources by this Program

Only a modest initial increase in the college budget is required to support the program. There are two reasons for

this. First, most of the courses in the program are already being given regularly, and most of the faculty needed to run the program are already with the college. Second, much of the increase in physical facilities is needed in any case to support the college wide increase in demand for computer services. However, once the program begins operating, increases in college enrollment will more than compensate for extra costs incurred by the program.

P. Administrative Structure

The present Mathematics Department shall be reconstituted as the Department of Mathematical Sciences comprised of three divisions, namely, the Division of Computer Science, the Division of Mathematics Education, and the Division of Statistics. The faculty of the Division of Computer Science shall be responsible for the development and maintenance of programs in computer and information science. The faculty of the Division of Mathematics Education shall be responsible for the development and maintenance of programs in basic skills, mathematics education (both graduate and undergraduate), and pure mathematics. The Division of Statistics shall be responsible for the development and maintenance of programs in statistics. All courses offered in the department will have a course number with either a CS, MATH, or STAT prefix.

Each division of the Department of Mathematical Sciences shall be headed by a Division Coordinator elected by the faculty of the respective division. The Division Coordinators shall have the responsibility within their respective divisions for scheduling courses, assignment of staff, and the administration of procedures for monitoring the division's programs and students. The Division Coordinators shall report directly to the Department Chairperson, who shall oversee the division programs, provide the divisions operational parameters, and provide the interface with the college administration. Division coordinators shall be given one hour of release time to perform the administrative functions.

A faculty member of the Department of Mathematical Sciences or any other department of the college shall be defined to be a voting member of a particular division if and only if at least one half of his or her teaching schedule is in the division. During the period of reformation the initial assignment of faculty to divisions will be made by the Dean of Arts and Sciences. Once the divisions are constituted the Division Coordinators will recommend all other transfers and/or new-hires with the approval of the Department Chairperson.

Policies and procedures adopted within the divisions of the Department of Mathematical Sciences shall be arrived by a democratic process of decision making.

Q. Equipment Budget

1. PRESENT EXPENDITURES - Supporting Academic Computing

a. Personnel

Presently one Programmer II allocates 75% of time in support of Academic Computing: $12,000/yr

b. Hardware/Software Services

Services are purchased from ECN. $28,000/yr

c. Equipment Rental & Maintenance

Terminals/Modems/ Keypunches $19,044/yr
(NOTE: Conversion to purchase or lease-purchase will greatly reduce this figure as a yearly expense).

d. Miscellaneous Items

Paper, cards, ribbons, etc. $ 4,000/yr

TOTAL EXPENDITURES in support of Academic Computing $63,044/yr

NOTE: The above expenditures (for FY 79-80) include equipment and services costs for Academic Computing support demands without any Computer Science Program implementation.

2. ADDED EXPENDITURES- (currently budgeted) to support increased demand for computer services

a. Personnel

Present Computer Programmer II allocated 100%. Adds $ 4,000/yr

Academic RJE/Term Support Section
Add Two (2) operators $16,000/yr

b. Hardware/Software (expanded services)
Services Supplied by ECN (some supplied local supplied local) $ 5,000/yr

c. Equipment

Add Three (3) Keypunches $ 3,060/yr

Add Three (3) Terminals $ 4,000/yr (purchase cost)

RJE Station (console/printer/ cardreader) Additional printer already on order	$12,000/yr
d. Miscellaneous Items	
Paper/cards/ribbons/etc.	$ 2,000/yr
TOTAL ADDITIONAL EXPENDITURES in support of Academic Computing Expense	$42,060/yr
Purchase	$ 4,000

3. Additional Budget to support Computer Science Program

a. Personnel	
One Computer Programmer III	$14,000/yr
b. Software services (from ECN grand total)	$ 5,000/yr
c. Equipment	
Terminals (purchase 3/yr for 3 yrs.)	$ 4,000/yr
Keypunches	$ 2,000/yr
Microcomputers (needed also for general use)	$10-15,000/3yr price
Lab equipment	$ 2,000/yr
d. Miscellaneous	$ 1,000/yr
Additional total Budget E pense	$24,000/yr
Purchase	$ 7,300-9,000/yr

R. Minor in Computer Science

The minor is intended for students who wish an in-depth knowledge of computing but whose major is in some other field. The minor consists of four required courses (12 cr) and four elective courses (12 cr). Courses in the elective component can be chosen so as to emphasize one of the following applications areas: business, science, mathematical

computing, social science education or computer science. Thus the minor can be structured to suit the needs and/or interests of a large number of students.

The required courses are:

CS 101 Computer Science I
CS 102 Computer Science II
CS 201 Assembly Language
CS 202 Data Structures

The remaining 9 credits may be chosen from the list of computer science major courses as well as from the additional list shown below.

Cobol I, II
Computer Applications in Science II
Computer Applications in the Social Sciences I, II
Computer Aided Instruction I, II
Man Made World I, II
Computers in Health Management
Computers in Art
Computers in Society

Some possible elective sequences for students interested in applications areas are:

Science

Computer Applications in Science I, II
Digital Electronics
Microprocessors and Microcomputers

Business

Cobol I, II
Total Systems Development
File Processing
System Analysis and Design
Data Base Design

Social Science

Computer Applications in Social Science I, II
Man Made World I, II
File Processing
Data Base Design

Education

Computer Aided Instruction I, II
Man Made World I, II
Digital Electronics

Service Courses

Many of the courses listed under the additional computer science course list do not have any prerequisites and hence may be taken by students who want to take just one or two courses in order to get some appreciation of computer methods in some selected field. Students wishing a general background in computing should consider such courses as:

Computer Science I, II
Man Made World I, II
Computers and Society

S. Summary and Conclusions

1. The degree of simulation of the curriculum is sufficient to enable students to enter the field as entry level computer programmers and within one to five years assume decision making roles in a development team, operations, and technical support.

2. The college has a sufficient number of qualified course developers and instructors in Mathematical Foundations, Hardware Organization, and Software Engineering, but will need full time or adjunct faculty to develope the courses in Business Information Systems Development.

3. Most of the program is already in place. The remaining parts of it could be developed within a year.

4. The curriculum has sufficient breadth to satisfy the needs of the several target populations.

5. The curriculum is designed to enable problem solving skills using proven mathematical, engineering, and business principles, and should remain stable at least through the early 1980's.

6. The equipment budget will need to be gradually increased.

7. The program is substantial, but not beyond the capabilities of the students at JCSC.

8. The program is ambitious, but not beyond the capabilities of the college, and deserves the support called for in this approval request.

9. "Training is everything. The peach was once a bitter almond; cauliflower is nothing but cabbage with a college education." Mark Twain, Pudd'nhead Wilson's Calendar, p. 37

Appendix 1

Course List

CS 101	Computer Science I
CS 102	Computer Science II
MATH 105	Calculus I
MATH 106	Calculus II
PHYS 105	Physics I
PHYS 106	Physics II
CS 103	Total Systems Development Concepts
CS 115	Man Made World
CS 201	Assembly Language
CS 202	Data Structures
CS 203	Discrete Structures
CS 204	Systems Analysis and Design
CS 205	Computational Statistics
MATH 206	Linear Algebra
CS 207	Digital Electronics
CS 208	File Processing
CS 301	Machine Organization
CS 302	Operating Systems
CS 303	Data Base Design
CS 304	Computer Architecture
CS 305	Programming Languages
CS 306	Algorithm Analysis
CS 307	Numerical Analysis I
CS 308	Numerical Analysis II
CS 309	Computer Application in Science
CS 401	Project/Coop
CS 402	Special Topics in Computing
CS 403	Systems Programming
CS 404	Physical Design of Computer Systems
CS 405	Micro-computers
CS 406	Theory of Formal Languages
CS 407	Theory of Computability

NOTE: Course Guides follow in numeric order.

SUBJECT:	Course Guide
COURSE NUMBER:	CS 101
COURSE NAME:	Computer Science I
DATE OF APPROVAL:	
CREDIT:	3 Semester Hours
PREREQUISITES:	Elementary Algebra
TERM:	Fall Semester
MAJOR:	Required
MINOR:	Required
GENERAL STUDIES:	Area A

COURSE DESCRIPTION:

The course introduces the Fortran programming language, concepts on algorithm development, and an overview of computer organization. It is a freshman level course providing basic concepts of computer science. Use is made of batch and inter-active computing systems.

I. Aims and Objectives:

The purpose of the course is to provide the computer science major with the fundamental concepts and knowledge required for work in the intermediate and advanced level courses. The course is intended to provide skills and knowledge in: (1) methods of problem solving and algorithm development; (2) a subset of a high level programming language such as BASIC, Fortran, PL/I; (3) techniques used in designing, coding, debugging, and documenting computer programs. The course is intended to lay the foundation for good habits in problem definition, analysis, design, implementation and documentation.

II. Content and Scope of the Course:

Module A (The Information Machine) 10%

1. Historical development
2. Essential components of an automatic digital computer
3. General description of systems of automatic computing

Module B (The Fortran Language) 45%

1. Representation of data and instructions
2. Data types
3. Arithmetic expressions
4. Input/output
5. Assignment statements
6. Logical and relational expressions
7. Sequencing, alternation, and iteration
8. Data structures, arrays
9. Operating procedures and systems
10. Programming projects

Module C (Algorithm Development) 45%

1. Problem solving techniques
2. Tools for algorithm analysis
3. Flowcharts, hierarchy charts
4. Tools for program design
5. Specification charts, psuedo-code
6. Simple numerical methods
7. Algorithms for searching, sorting, merging
8. Business applications

III. Procedures, Techniques, and Instructional Strategies:

1. The student will study and learn through "guided design"
2. The student is introduced to a subset of Fortran
3. Subject matter and methodology presented through lectures
4. Course includes a laboratory component
5. Emphasis is on techniques of algorithm development
6. Experience provided with open ended problems
7. Material covered as an integrated whole

IV. Instructional Materials:

1. Textbook, programming manuals
2. Batch and interactive computing system
3. Design tools such as coding form, specification charts, etc.

V. Course Requirements:

1. Approximately 12 programming problems assigned
2. Midterm and final examination
3. Classroom participation

VI. Textbook:

Walker, T. M., Introduction to Computer Science: An Interdisciplinary Approach, Allyn and Bacon, Inc, 1972.

VII. Bibliography

Thomas C. Bartee, Introduction to Computer Science, McGraw, 1975.

M.S. Carberry, H.M.Khalil, F.J. Leathrum, L.S. Levy, General Computer Science, Merrill, 1976.

Gordon B. Davis, Introduction to Computers, (3rd ed.) McGraw, 1977.

A.I. Forsythe, T.A. Kennan, E.I. Organick, W. Stenberg, Computer Science: A First Course, (2nd ed.) Wiley, 1975.

C.W. Gear, Introduction to Computer Science: Short Edition SRA, 1975.

B. Higman, Foundation Course in Computer Science, Elsevier, 1975.

Harry Katzan, Jr., Introduction to Computer Science, Petrocelli, 1975.

Gerald W. Kimble, Information and Computer Science, Holt, 1975.

Marray Laver, An Introduction to the Uses of Computers, Cambridge, 1976.

Herbert Maisel, Computers: Programming and Applications, SRA, 1976.

Peter Naur, Concise Survey of Computer Methods, Petrocelli, 1975.

C. Peter Olivieri, Michael W. Rubin, Computers and Programming: A Neoclassical Approach, McGraw, 1975.

Alan J. Perlis, Introduction to Computer Science, Harper, 1975.

Edwin R. Sage, Fun and Games with the Computer, Entelek, 1975.

Frederic Tonge, Julian Feldman, Computing: Introduction to Procedures and Procedure - Followers, McGraw, 1975.

Terry M. Walker, Fundamentals of Computer Science, Allyn, 1975.

W. Y. Arms, J.E. Baker, R.M. Pengelly, A Practical Approach to Computing, Wiley, 1976.

P.M. Banks, J. Doupnik, Introduction to Computer Science, Wiley, 1976.

SUBJECT:	Course Guide
COURSE NUMBER:	CS 102
COURSE NAME:	Computer Science II
DATE OF APPROVAL:	
CREDIT:	3 Semester Hours
PREREQUISITES:	Computer Science I
TERM:	Spring Semester
MAJOR:	Required
MINOR:	Required
GENERAL STUDIES:	Area A

COURSE DESCRIPTION:

This course is a continuation of Computer Science I, using the PL/1 programming system.

Techniques of disciplined programming including tools for program design involving large problems and algorithmic analysis. Programming topics include string processing, recursion, internal search methods, concepts in data structures and program structures.

I. Aims and Objectives:

The purpose of the course is to provide a grounding in disciplined program design, in programming style, and in debugging and testing of programs. Skills in modular design, stepwise refinement, and top down programming are provided. The aim is to demonstrate the principles of software engineering and design.

II. Course Content and Scope:

Module A (Style) 15%

1. Principles of disciplined programming
2. Documentation standards
3. Elements of PL/1

Module B (Structured Programming) 20%

1. More PL/1
2. Control flow
3. Unvariant relation of a loop
4. Stepwise refinement of statements and data structures

Module C (String Processing) 15%

1. More PL/1
2. String operations
3. Introduction to the algebra of strings

Module D (Data Structures) 20%

1. More PL/1
2. Linear allocation
3. Link allocation
4. Introduction to graph theory

Module E (Internal Searching and Sorting) 20%

1. More PL/1
2. Binary search
3. Radix search
4. Shell search
5. Quicksort
6. Merge sort

Module F (Recursion) 10%

1. More PL/1

III. Procedures, Techniques, and Instructional Strategies:

1. Some programming projects will be team projects
2. A case study involving a realistic application will be used
3. Emphasis will be on good programming style, expression, and documentation
4. Course includes a laboratory component
5. Material covered as an integrated whole

IV. Instructional Materials:

1. Textbook, programming manual
2. Batch and interactive computing system
3. Design tools

V. Course Requirements:

1. Approximately 12 programming problems assigned
2. Participation in team project
3. Classroom participation
4. Midterm and final examination

VI. Textbook:

M.V. Zelkowitz, A.C. Shaw, J.D. Gannon, Principles of Software Engineering and Design, Prentice Hall, 1979.

VII. Bibliography

F. L. Bauer, (ed.), Software Engineering: An Advanced Course, Springer, 1975.

Frederick P. Brooks, Jr., The Mythical Man-Month, Essays on Software Engineering, AW, 1974.

John Buxton, Peter Naur, Brian Randall, (eds.), Software Engineering Concepts and Techniques, Petrocelli, 1976.

Maurice Halstead, Elements of Software Science, Elsevier, 1977.

Ellis Horowitz, (ed.), Practical Strategies for Developing Large Software Systems, AW, 1975.

Glenford J. Meyers, Software Reliability: Principles and Practices, Wiley, 1976.

A.I. Wasserman, P. Freeman, (eds.), Software Engineering Education: Needs and Objectives, Springer, 1976.

Joel D. Aron, The Program Development Process, Part 1 - The Individual Programmer, AW, 1975

A.R. Brown, W.A. Sampson, Program Debugging, Elsevier, 1973.

Richard Conway, David Gries, Primer on Structured Programming Using PL/1 and PL/C and PL/CT, Winthrop, 1976.

R. Conway, D. Gries, D. Wortman, Introduction to Structured Programming, Using PL/1 and SP/K, Winthrop, 1977.

O.J. Dahl, E.W. Dijkstra, C.A.R. Hoare, Structured Programming, Academic, 1972.

Bernard Fischer, Herman Fischer, Structured Programming in PL/1 and PL/C, Dekker, 1976.

F.L. Friedman, E.B. Koffmaan, Problem Solving and Structured Programming in FORTRAN, AW, 1977.

Dennis P. Geller, Daniel P. Freedman, Structured Programming in APL, Winthrop, 1976.

F. Gruenberger, (ed.), Effective vs. Efficient Computing, PH, 1973.

William C. Hetzel, Program Test Methods, PH, 1973.

Joan K. Hughes, J. I. Michtom, A Structured Approach to Programming, PH, 1977.

Brian W. Kernighan, P.J. Plauger, The Elements of Programming Style, McGraw, 1974.

B.W. Kernighan, P.J. Plauger, Software Tools, AW, 1976.

Richard B. Kieburtz, Structured Programming and Problem Solving with ALGOL W, PH, 1975.

R. B. Kieburtz, Structured Programming and Problem Solving with PL/1, PH, 1977.

Henry F. Ledgard, Programming Proverbs, Hayden, 1975.

H.F. Ledgard, Programming Proverbs for FORTRAN Programming, Hayden, 1975.

Michael Marcothy, Structured Programming with PL/1, An Introduction, PH, 1977.

Clement L. McGowan, John R. Kelley, Top-Down Structured Programming Techniques, Petrocelli, 1975.

Glenford J. Myers, Reliable Software Through Composite Design, Petrocelli, 1975.

Dennie Van Tassel, Program Style, Design, Efficiency, Debugging and Testing, PH, 1974.

G. M. Weinberg, N.F. Yasukawa, R. Marcus, Structured Programming Using PL/C: An Abecedarian, Wiley, 1973.

D.M. Wheatley, A.W. Unwin, The Algorithm Writer's Guide, Longman, 1973.

Niklaus Wirth, Systematic Programming, An Introduction, PH, 1973.

N. Wirth, Algorithms + Data Structures = Programs, PH, 1976.

Bibliography Cont.

Raymond T. Yeh, (ed.), Current Trends in Programming Methodology, Vol. I, Software Specifications and Design, PH, 1977.

R. T. Yeh, (ed.), Current Trends in Programming Methodologies, Vol. II, Software Architecture Design, PH, 1977.

Edward Yourdon, Techniques of Program Structure and Design, PH, 1976.

E. Yourdon, How to Manage Structured Programming, Yourdon, 1976.

E. Yourdon, Larry L. Constantine, Structured Design, Yourdon, 1975.

SUBJECT:	Course Guide
COURSE NUMBER:	
COURSE NAME:	Total Systems Development Concepts
DATE OF APPROVAL:	
CREDIT:	3 Semester Hours
PREREQUISITE:	Computer Science I and II
TERM:	Fall
MAJOR:	Elective
MINOR:	Elective
GENERAL STUDIES:	No

COURSE DESCRIPTION:

Total Systems Development Concepts (TSDC) presents a conceptual overview of the eight phases of a typical life cycle for an information systems development project. The fundamental principles and practices of meeting business needs and opportunities with a fully operational, and often computerized, information system are presented in TSDC. The student is introduced to typical organizational constraints, approval authorities, documentation requirements administrative techniques and major activities associated with the systems development process.

I. General Aim:

The student will understand the application of a top-down phased approach to the development of systems that process information on computers to meet business needs. The student will be able to identify the tools and techniques that are used, define the terms unique to systems development and describe the eight phases in terms of their goals, activities and products.

II. Specific Objectives:

Upon completion of this course, the student will be able to:

1. Relate information systems to business goals and objectives.
2. Describe the eight phases of a typical information systems development life cycle. For each phase, the student will
   a) State its objective

b) Describe its major activities
c) Recognize its products

3. Describe the functional roles of a project required to do a systems development project. Describe how the complexion of the team changes from phase to phase.

4. Describe the tools and techniques to apply to complete the activities of a TSD project.

5. Identify and correlate the following subsystems that must be developed for a TSD project:

   - Computer Subsystem
   - Personnel Subsystem
   - Testing and Conversion Subsystem

6. Describe the purpose and application of project management to the TSD process.

III. Content and Scope of the Course:

A. Definitions

1. Business systems vs. information systems
2. Techniques: Systems analysis, systems design, programming, structured theories, data base, top-down, etc.
3. Data vs. Information
4. Project related terms: functional roles, work plans, life cycle, etc.

B. Relationship of information systems to business goals

1. Role of information systems
2. Benefits
3. Hazards

C. The Life Cycle

1. Overview of Phases
   - General architecture
   - Iterative nature
2. Phase I - Proposal
3. Phase II - Feasibility
4. Phase III - Definition
5. Phase IV - Preliminary Design
6. Phase V - Detail Design
7. Phase VI - Implementation
8. Phase VII - Conversion
9. Phase VIII - Performance Review

D. The Project Management of TSD

1. The Workplan
2. The Cost Estimates
3. The Resource Estimates
   - Personnel
   - Hardware Support
   - Software Support

E. Project Team

1. Functional Roles
2. Complexion of team by phase

F. Tools and Techniques

1. Analysis
2. Design
3. Programming
4. Project Management
5. Human Factors engineering

G. Summary and Exam

IV. Instructional Strategy:

The theory and concepts will be presented in lecture mode supplemented by reading assignments. Exercises and discussion will be employed to reinforce the objectives.

V. Instructional Materials to be used in the Course:

Basic Texts:

A. Managing the Systems Development Process by Biggs, Birks, Atkins, (Prentice Hall)

B. Information Systems by Burch and Strater, (Hamilton)

C. Business Systems by Willoughby and Seym (Associa- for System Management)

VI. Basic Requirements for Successful Completion of the Course:

A. Attend class and discuss material
B. Pass final examination

VII. Bibliography

Russell L. Ackoff, Fred E. Emery, On Purposeful Systems, Aldine, 1972.

Sherman S. Blumenthal, Management Information Systems: A Framework for Planning and Development, PH, 1969.

William A. Bocchino, Management Systems: Tools and Techniques, PH, 1972.

Gordon B. Davis, Management Information Systems: Conceptual Foundations, Structure, and Development, McGraw, 1974.

G. B. Davis, Gordon Everest, Reading in Management Information Systems, McGraw, 1976.

Alan Eliason, Kent D. Kitts, Business Computer Systems and Applications, SRA, 1974.

F. E. Emery, (ed.), Systems Thinking, Pengiun, 1972.

Bart E. Holm, You Can Manage Your Information, Van, 1968.

Edward O. Joslin, (ed.), Computer Readings for Making It Count, College, 1976.

F. G. Kirk, Total Systems Development for Information Systems, Wiley, 1973.

Borje Langefors, Theoretical Analysis of Information Systems, (4th ed.), Petrocelli, 1973.

Henry C. Lucas, Toward Creative Systems Design, Columbia 1974.

Henry C. Lucas, The Analysis, Design and Implementation of Information Systems, McGraw, 1976.

H. C. Lucas, Why Information Systems Fail, Columbia, 1975.

Chris Mader, R. Hagin, Information Systems: Technology, Economics and Applications, SRA, 1974.

James Martin, Design of Man-Computer Dialogues, PH, 1973.

Lawrence Orilla, Nancy B. Stern Robert Stern, Business Data Processing Systems, Wiley, 1972.

Thomas R. Prince, Information Systems for Management Planning and Control, (3rd Ed.), Irwin, 1975.

Theodore C. Willoughby, James A. Senn, Business Systems, Sys. Mang., 1975.

Frederick G. Withington, The Use of Computers in Business Organizations, AW, 1971.

SUBJECT:	Course Guide
COURSE NUMBER:	CS 115
COURSE NAME:	Man Made World
DATE OF APPROVAL:	
CREDIT:	3 Semester Hours
PREREQUISITE:	College Admission
MAJOR:	Elective
MINOR:	Elective
GENERAL STUDIES:	Yes
TERM:	Spring and Fall

COURSE DESCRIPTION:

A survey of concepts dealing with technological-societal interaction. Topics include techniques in decision making such as game theory and linear programming, modeling for population and pollution problems, the concept of systems with applications to biological, social systems, dynamics with applications to speech, communication, prosthetics, and simulation techniques using the analog and digital computer.

I. Aims and Objectives:

The purpose of the course, Man Made World, is to:

1. Introduce the student to those technological concepts which are applicable to problems having a social and economic as well as a technical aspect.

   Help the student gain insight into the processes of interaction of technology and society from a practical rather than a pure science point of view.

2. Introduce the student to concepts underlying systems which relate directly to the life of every individual in our society, e.g. pollution control, resource management, etc.

   Help the student gain insight into approaches to the solution of problems so that he can participate in a more intelligent manner in the processes of decision making necessary in today's technological society.

3. Encourage the development of rational attitudes toward technological change and avoid the extremes of undue fear or excessive optimism.

4. Familiarize teachers with a course being widely suggested, as a course for high school students to illustrate the cultural significance of science and technology.

5. To show how science and mathematics is related through technology to social conditions and how to apply the analytical methods of science in multidisciplinary approaches to studying and solving societal problems.

II. Specific Course Objectives:

The content of the course includes the following topics:

1. General considerations dealing with the interaction of man and machine including the problem of matching technology to man.

2. The concepts and processes involved in decision making such as criteria selection, algorithms for solutions, and dynamic programming.

3. The concept of optimization by the methods of game theory and linear programming.

4. The techniques of modeling illustrated by examples dealing with traffic flow, population, and resource management.

5. The concept of systems including input-output ideas, subsystems, and modeling of systems using the analog and digital computers.

6. Communication systems including coding concepts, physical characteristics of sounds and problems associated with transmission.

7. The concept of feed back in automatic control and biological systems.

8. The concept of stability in feed back systems such as traffic flow, the epidemic model, and physical systems.

9. The concepts of man as controller, limited by environmental needs, his senses, and prosthetics.

10. The concepts underlying the design and operation of general purpose digital computers.

III. Methods of Evaluation:

1. The Man Made World feed back instruments
2. Participation in class activities and projects
3. Completion of assigned problems and projects

IV. Course Content and Scope:

1. Matching Technology to Society
   a. Reaction time
   b. Multiphasic health testing
2. Decision Making
   a. Types of decisions
   b. Algorithms
   c. Criteria
   d. Optimization with few alternatives
   e. Dynamic programming
3. Optimization
   a. Queueing
   b. Games
   c. Linear programming
4. Modeling
   a. Graphs
   b. Models for resource management
   c. Populations models
   d. Exponential growth
5. Systems
   a. Definition
   b. Subsystems
   c. Simulation on analog and digital computers
6. Communication Systems
   a. Language
   b. Sinusoidals
   c. Spectrograms
7. Feedback Systems
   a. Goal seeking
   b. Disturbance control
   c. Instability

8. Stability

   a. Epidemic model
   b. Instability in physical systems

9. Man and Machines

   a. Man as controller
   b. Environmental needs
   c. Senses
   d. Prosthetics

V. Text:

Man Made World: Staff of E.C.C.P.; McGraw-Hill, 1971.

VI. Procedures and Methodology:

The methodology to be used to cover the content of the course includes lectures dealing with the concepts, text readings, questions for class discussion and individual study, problems for both group and individual solution, and laboratory experiments and projects.

VII. Text and Materials:

The text to be used for the course is The Man Made World developed by the staff of the Engineering Concepts Curriculum Project and published by McGraw-Hill Book Co.- Price: $15.00

Special equipment needed for the course includes the analog and digital computer, logic circuit boards, sound meters, and the cardiac computer.

VIII. Students:

Since the course does not carry a prerequisite, the approach to the topics in the course is not highly mathematical and theoretical, and should therefore be available to all students of the college. However, since the main thrust of the course deals with the impact of science and technology on society, the course should satisfy the general studies requirement in the natural science area. Moreover, since the course includes many of the concepts and methods which came out of computer technology, it is suggested that the course be listed under the interdisciplinary area of Computer Studies, and therefore be counted as part of the 24 credit requirement of students concentrating in this area.

Another possibility is to list it in the area of Urban Studies. The methodology of the course is the methodology of science and technology, but the focus is on man and his relationship to a technologically based society with its pitfalls and promises.

Since the problems associated with technological societal interaction bear down most heavily on people living in the centers of our urban areas, familiarity with rational approaches to the solution of these problems would be particularly appropriate for students studying at a college which has a stake in their solution.

IX. Evaluation:

The performance of the students in the course is to be made by use of the Man Made World Feedback Instruments, by student participation in class activities and projects, and by completion of assigned problems and projects.

Evaluation of the degree to which the course meets the objectives of the general studies program is to be accomplished by student self-evaluations which are part of the Man Made World Feedback Instruments. A summary of the AAAS Guidlines and Standards for the Education of Secondary School Teachers is attached with those guidelines checked which Man Made World is designed to meet.

SUBJECT:	Course Guide
COURSE NUMBER:	CS 201
COURSE NAME:	Assembly Language
DATE OF APPROVAL:	
CREDIT:	3 Semester Hours
PREREQUISITES:	Computer Science I
TERM:	Fall and Spring Semester
MAJOR:	Required
MINOR:	Required
GENERAL STUDIES:	No

COURSE DESCRIPTION:

Computer structure, machine language, instruction execution, addressing techniques, and digital representation of data. Symbolic coding and assembly systems, macro definition and generation, and program segmentation and linkage. Computer projects to illustrate machine structure and programming techniques.

I. Objectives:

This course, together with Computer Science I and II, provides the basic concepts needed for further work in computer science. The emphasis is on the basic structure of the machine from a functional and programming point of view and it is intended to provide the student with and understanding of the internal behavior of computers. A major objective is to provide the student with some facility in assembly language programming.

II. Content and Scope of the Course:

A. Computer Software System Space

1. System space parameters
2. Computer system space
3. The assembly language environment

B. Number Systems

1. The binary number system and arithmetic

2. The hexadecimal number system and arithmetic
3. Binary and hexadecimal conversions
4. Complement notation

C. System Architecture

1. Computer storage
2. System main memory
3. Independence of address and content
4. Data and arithmetic concepts
5. Control concepts
6. Base register addressing
7. Condition code concept
8. Program interruption concept

D. Register Address Logic and Integer Arithmetic

1. Register address logic
2. Integer arithmetic

E. Concepts of Branching and Indexing

1. The Instruction Address Counter
2. Branching and linking operations
3. Branching and counting operations
4. Branching on condition operations
5. Branching and indexing operations

F. Character Operations

1. Standard character codes
2. Packed decimal representation
3. Character oriented operations
4. Conversion instructions

G. Logical Operations

H. Packed Decimal Operations

I. Floating-Point Arithmetic Operations

1. Floating point data representation
2. Floating point operations
3. Extended precision and rounding

J. Status Switching and the Program Status Word

1. Machine state and PSW
2. The interrupt system
3. Instructions for status switching
4. Concept of macro programming
5. Macros for program termination, linkage, interrupt control timer operations, core management

K. Concepts of Input and Output

1. Input/Output devices, control units, and channels

L. Queued Sequential-Access Method

M. Basic Direct-Access Method

N. User Defined Macros

1. Macro definition components
2. Conditioned-assembly concepts

III. Procedures, Techniques and Methods:

1. Lectures covering theory and areas of application
2. Verification of Assembly Language programs using facilities of Center for Computing Services
3. Assignment of problems

IV. Instructional Materials:

1. Text, IBM 360/370 Computing System (batch and terminal)

V. Basic Requirements for Successful Completion of Course:

1. Satisfactory run of 3 to 12 programs using the computer
2. Final examination
3. Class participation

VI. Basic Text for the Course:

Introduction to Machine and Assembly Language; Systems 360/370, by Frank D. Vickers (Holt, Rinehart and Winston, Inc.) 1971 Price- $10.00

VII. Bibliography

D. W. Barron, Assemblers and Loaders, (2nd ed.), Elsevier, 1972.

Harrington C. Breasley, Jr., Introduction to Assembler Language Programming for the IBM 360/370, Macmillan, 1974.

Barbara J. Burian, A Simplified Approach to S/370 Assembly Language Programming, PH, 1977.

M. Campbell-Kelly, An Introduction to Macros, Elsevier, 1973.

Thomas J. Cashman, Gary B. Shelly, Introduction to Computer Programming IBM System/360 Assembler Language, Anaheim, 1973.

Bibliography Continued

Ned Chapin, 360/370 Programming in Assembly Language, (2nd ed.) McGraw, 1973.

Irving Allen Dodes, Assembly Language Basics: An Annotated Program Book, Hayden, 1975.

Ivan Flores, Assemblers and BAL, PH, 1971.

Ivan Flores, The BAL Machine, Allyn, 1972.

Ivan Flores, Computer Programming System/360, (2nd ed.), PH, 1971.

Ralph Grishman, Assembly Language Programming for the Control Data 6000 Series and Cyber Seventy Series, Algorithmic, 1974.

Reino Hannula, Computers and Programming, A System 360-370 Assembler Language Approach, Houghton, 1974.

Floyd E. Haupt, Elementary Assembler Language Programming, Merrill, 1972.

Gopal K. Kapur, IBM 360 Assembler Language Programming, Wiley, 1971.

Donald E. Knuth, MIX, AW, 1970.

Shan S. Kuo, Assembler Language for FORTRAN, COBOL, and PL/1 Programmers, AW, 1974.

Herbert D. Leeds, Gerald M. Weinberg, Computer Programming Fundamentals: Based on the IBM System 360, McGraw, 1970.

Kevin McQuillen, System/360-370 Assembler Languages (DOS), Murach, 1974.

W. H. Payne, Machine Assembly and Systems Programming for IBM 360, Harper, 1970.

Wilson T. Price, Elements of IBM 1130 Programming, Holt, 1968.

James Rosenberg, Introduction to IBM/360 Assembler Language, Holden, 1970.

Walter G. Rudd, Assembly Language Programming and the IBM 360 and 370 Computer, PH, 1975.

G. Shelly, T. Cashman, IBM System/360 Assembler Language, Anaheim, 1974.

Bibliography Cont.

G. Shelly, T. Cashman, IBM System/360 Assembler Language, Disk/Tape Advanced Concepts, Anaheim, 1974.

Wilson E. Singletary, Ross A. Overbeek, Assembler Lang- with ASSIST, SRA, 1976.

D. H. Stabley, System/360 Assembler Language, Wiley, 1967.

D.H. Stabley, Logical Programming with System/360, Wiley, 1970.

Richard H. Stark, Donald W. Dearholt, Computer Concepts and Assembler Programming 360/370 Systems, Academic, 1975.

Michael J. Stimson, Richard E. Linder, Assembly Language Programming: IBM System/360-370 (DOS), AW, 1975.

George W. Struble, Assembler Language Programming: The IBM System/360-370 (2nd ed.), AW, 1975.

Alex Thomas, Jr., System/360 Programming, Holt, 1971.

Sharron Tuggle, Assembler Language Programming: System 360 and 370, SRA, 1975.

Frank D. Vickers, Introduction to Machine and Assembly Language: System 360/370, Holt, 1971.

Eric Weiss (ed.), Computer Usage/360 Assembly Language, McGraw, 1970.

Walter J. Weller, Assembly Level Programming for Small Computers, Lexington, 1975.

Robert J. Wimmert, Computer Programming Techniques: Vol. I: Machine/Assembly Language, Holt, 1968.

SUBJECT:	Course Guide
COURSE NUMBER:	CS 202
COURSE NAME:	Data Structures
DATE OF APPROVAL:	
CREDIT:	3 Semester Hours
PREREQUISITES:	Computer Science I, II
TERM:	Fall
MAJOR:	Required
MINOR:	Required
GENERAL STUDIES:	No

COURSE DESCRIPTION:

This course consists of a survey of topics in graph theory, including directed and planar graphs, trees, properties of graphs such as connectedness, completeness, isomorphisms, etc., adjcency and path matrices for representing graphs. Algorithms for traversing trees, the shortest path problem, generating linked structure, sorting, etc. Topics in the algebra of strings and string manipulation. Applications of the above concepts in handling linear lists, arrays, storage data structures, etc. using the PL/I programming system.

I. Aims and Objectives:

1. To introduce the student to the elements of graph theory from both a mathematical and an applications point of view.

2. To introduce methodology for representing relations with applications to relations which hold among data items.

3. To introduce concepts involved in representing and characterizing non-numeric data such as bit strings, character strings, etc.

4. To provide programming experience in verifying algorithms for solving problems which depend on operations on data structures.

5. To help the student develop a mathematical intuition for the structure of relations in discrete finite sets of data elements.

6. To provide experience with the PL/I programming system and its capabilities for handling various data types and data structures.

7. To introduce algorithmic analysis and design criteria for data manipulation in a database environment.

II. Content and Scope of the Course:

A. Functions and Relations 10%

1. The ordered pair and related concepts
2. Relations
3. Equivalence relations
4. Equivalence classes and partitions

B. Graph Theory

1. Basic definitions in the theory of digraphs 20%
2. Digraphs, matrices and relations
3. Connectedness
4. Linear formulas of digraphs
5. Isomorphism of digraphs
6. Planar graphs

C. Strings 10%

1. Algebraic structures
2. Algebra of strings
3. Marker algorithms

D. Trees 10%

1. Trees as grammatic markers
2. Representation of prefix formulas
3. Sort trees and dictionaries
4. Decision trees and decision tables

E. Paths and Cycles in Digraphs 10%

1. Shortest path problem
2. Cycles
3. Critical Path scheduling

F. Arrays 10%

1. Storage media and their properties
2. Storage of arrays
3. Sparce matrices

G. Pushdown stores, Lists and List Structures 10%

1. Pushdown stores
2. Prefix, postfix, and infix formulas
3. Lists and list structures
4. Threaded and symmetric lists
5. PL/1-type data structures

H. Organization of Files 10%

1. Records and files
2. Indexed files
3. Scatter storage techniques
4. Sorting

III. Procedures, Techniques and Methods:

1. Lectures covering theory and areas of application
2. Verification of PL/1 programs using facilities of the Center for Computing Services
3. Assignment of problems

IV. Instructional Materials:

Text, IBM 360 Computing System (batch and terminal)

V. Basic Requirements:

1. Satisfactory run of programs on the computer
2. Class participation
3. Final examination
4. Above requirements made known during first lecture

VI. Basic Text:

A.T. Berztiss, Data Structures: Theory and Practice, 2nd edition, (Academic Press) 1975.

VII. Bibliography

A.T. Berztiss, Data Structures: Theory and Practice, (2nd ed.), Academic, 1975.

Peter C. Brillinger, Doron J. Cohen, Introduction to Data Structures and Non-numeric Computation, PH, 1972.

Mark Elson, Data Structures, SRA, 1975.

Bibliography Continued

Ivan Flores, Computer Sorting, PH, 1969.

I. Flores, Data Structure and Management, (2nd ed.), PH, 1977.

J.M. Foster, List Processing, Elsevier, 1968.

T.R. Gildersleeve, Design of Sequential File Systems, Wiley, 1971.

P.A. Hall, Computational Structures, Elsevier, 1975.

Malcolm C. Harrison, Data Structures and Programming, Scott, 1973.

William Haseman, Andrew Whinston, Introduction to Data Management, Irwin, 1977.

Ellis Horowitz, Saratj Sahni, Fundamentals of Data Structures, Comp. Sci., 1976.

D. R. Judd, Use of Files, Elsevier, 1972.

A. Klinger, K.S. Fu, T.L. Kunii, Data Structures, Computer Graphics and Pattern Recognition, Academic, 1977.

Donald E. Knuth, The Art of Computer Programming, Vol. I, Fundamental Algorithms, (2nd ed.), AW, 1973.

D.E. Knuth, The Art of Computer Programming, Vol. III: Sorting and Searching, AW, 1973.

David Lefkovitz, File Structures for on-line Systems, Hayden, 1969.

T.C. Lewis, M.Z. Smith, Applying Data Structures, Houghton, 1976.

Harold Lorin, Sorting and Sort Systems, AW, 1975.

H.A. Mauer, Data Structures and Programming Techniques, PH, 1977.

John Pfaltz, Computer Data Structures, McGraw, 1977.

Robert P. Rich, Internal Sorting Methods Illustrated with PL/1 Programs, PH, 1972.

Harold S. Stone, Introduction to Computer Organization and Data Structures, McGraw, 1972.

H.S. Stone, Daniel P. Siewiroek, Introduction to Computer Organization and Data Structures,(PDP-11 ed.), McGraw, 1975.

Bibliography Cont.

Jean Paul Tremblay, Paul Gordon Sorenson, Introduction to Data Structures, McGraw, 1976.

W.M. Waite, Implementing Software for Non-numeric Applications, PH, 1973.

Peter Wegner, Programming Languages, Information Structures and Machine Organization, McGraw, 1968.

SUBJECT:	Course Guide
COURSE NUMBER:	CS 203
COURSE NAME:	Discrete Structures
DATE OF APPROVAL:	
CREDIT:	3 Semester Hours
PREREQUISITES:	Calculus I, II
MAJOR:	Elective
MINOR:	Required
GENERAL STUDIES:	No
TERM:	Spring

COURSE DESPRIPTION:

The course introduces basic theoretical tools for describing algorithmic processes. Topics include: propositional logic, set theory, groups, rings, fields, graphs, inductions, sequential machines, error detecting codes, introduction to computabilities.

I. Objectives:

General Aims

1. To acquaint the student with the axiomatic approach to mathematical theory.
2. To acquaint the student with methods of proof used to develope algebraic theory.
3. To provide the student with experience of developing the theory of algebraic structures in the abstract and to show the relationship between the theory and particular models.

Specific Objectives

1. To help the student understand important methods, constructions, proofs, and ideas used in developing the properties of discrete structures.
2. To help the student understand the unifying idea of factorization and decomposition in the investigation of algebraic systems.
3. To develope the theoretical tools useful for describing algorithmic applications.

II. Course Content and Scope:

1. Basic Forms and Operations

   a. Basic set operations, set algebra
   b. Computer representation of sets
   c. Relations, mappings
   d. Equivalence relations and classes
   e. The lattice of subsets

2. Directed Graphs

   a. Basic definitions
   b. Matrix representations
   c. Special classes of graphs
   d. Minimal cost flows, shortest path problems, critical paths
   e. Graphs of multiprocessing systems
   f. Information networks

3. Trees

   a. Basic definitions
   b. Prefix representation and tree forms
   c. Searching, subroutines, and theorem proving

4. Groups

   a. Basic definitions
   b. Basic theorems
   c. Groups of graphs
   d. Permutation groups

5. Partial Orders and Lattices

   a. Basic definitions
   b. Partial orders
   c. Lattices, atomic lattices

6. The Propositional Calculus

   a. Basic definitions
   b. Well-formed formulas
   c. Minimal sets of operators
   d. Polish notation

7. Combinatorics

   a. Permutations and combinations of objects
   b. Enumerators for permutations and combinations
   c. Cycle classes
   d. Partitions and compositions

8. Finite Fields

   a. Basic definitions
   b. Representation and structure
   c. Minimal polynomials, irreducible polynomials
   d. Error correcting codes
   e. Sequential machines

III. Procedures, Techniques, and Methods:

1. Lectures covering the theory and areas of application
2. Facilities of the Center for Computing Services
3. Assignment of problems

IV. Instructional Materials:

1. Textbook
2. Batch and interactive computing systems

V. Basic Requirements:

1. Class participation
2. Completed assignments
3. Programming assignments
4. Final Examination
5. Above requirements made known during first week of course

VI. Basic Text:

R. R. Korfhage, Discrete Computational Structures, (Academic Press) 1974. Price $15.00

VII. Bibliography

Nicos Christofides, Graph Theory-An Algorithmic Approach, Academic, 1975.

Narsigh Deo, Graph Theory with Applications to Engineering and Computer Science, PH, 1974.

Arthur Gill, Applied Algebra for the Computer Sciences, PH, 1976.

Frank Harary, Graph Theory, AW, 1969.

Robert Korfhage, Discrete Computational Structures, Academic, 1974.

Bibliography Continued

Seymour Lipschutz, Discrete Mathematics Including 600 Solved Problems, McGraw, 1976.

C.L. Liu, Discrete Mathematics for Computer Science, McGraw, 1977.

Clifford W. Marshall, Applied Graph Theory, Wiley, 1971.

R.E. Prather, Discrete Mathematical Structures for Computer Science, Houghton, 1976.

Franco P. Preparata, Raymond T. Yeh, Introduction to Discrete Structures, AW, 1973.

Ronald C. Read, Graph Theory and Computing, Academic, 1972.

Donald F. Stanat, David F. McAllister, Discrete Mathematics in Computer Science, PH, 1977.

Harold S. Stone, Discrete Mathematical Structures and Their Applications, SRA, 1973.

Jean Paul Tremblay, Ram P. Manohar, Discrete Mathematical Structures with Application to Computer Science, McGraw, 1975.

Mark B. Wells, Elements of Combinatorial Computing, Pergamon, 1971.

SUBJECT:	Course Guide
COURSE NUMBER:	CS 204
COURSE NAME:	Systems Analysis and Design
DATE OF APPROVAL:	
CREDIT:	
PREREQUISITE:	Total Systems Development Concepts
AUDIENCE:	Computer Science Majors (Business Majors)
TERM:	Fall
MAJOR:	Elective
MINOR:	Elective
GENERAL STUDIES:	No

I. COURSE DESCRIPTION:

Systems Analysis and Design (SAD) is designed to allow the student to practice the concepts presented in Total Systems Development Concepts. It is centered around the theory and model called "data flow diagrams'. Data flow diagrams were developed by Chris Bane and Trish Sarson and are used to analyze and define the logical model of a business information system.

II. General Aim:

The student will be able to build a logical model of a business information system. This model provides the logical, or non-physical, descriptions of the system's functions, data elements and objectives.

III. Instructional Strategy:

The theory and definitions will be presented through lecture and discussion. Exercises will be employed to reinforce each objective. The material will be made cohesive with one real-life, comprehensive case study.

IV. Basic Requirements for Successful Completion:

1. Satisfactory completion of Case Study

2. Class participation
3. Final examination

V. Basic Text:

Structured Systems Analysis: Tools and Techniques by Chris Gane and Trish Sarson, (Improved System Technologies, Inc.; 888 Seventh Avenue, New York City 10010; 1977)

VI. Objectives:

Given a description of a typical business environment, the student will:

A) Analyze the problems and/or opportunities that exist in the flow and use of information in that business.

B) Determine the overall needs/goals of the business.

C) Define objectives of the business.

D) Define objectives of the information system to support the business objectives.

E) Identify and name the logical functions of the information system.

F) Identify and name the data elements and data stores.

G) Create and document a data flow diagram of the logical flow of the data elements through the system's functions and the data stores which they access.

VII. Outline:

A. Review of Concepts and Language of Systems Analysis and Design.

B. Need for good "Analysis" tools.

C. Description of the tools.

1) Data Flow Diagram
   a) Data Stores
   b) Data Flows
   c) Processes
   d) Source/Destination of data

2) Data Dictionary

3) Logic Descriptions

D. Developing the Data Flow Diagrams

1) Explosion conventions
2) Symbol conventions
3) Error and Exception Handling

E. Data Dictionary

1) What to put into one
2) How to get data out of one
3) Considerations for cross-project or organization-wide dictionaries
4) Relationship to distributed processing

F. Process Logic

1) Problems
2) Decision trees
3) Decision tables
4) Structured English, Pseudocode, Tight English

G. Data Stores

1) Normalization of data
2) Building relation structures; Projection and Join

H. Response Requirements

1) Define access requirements
2) Types of data structures
3) Types of query
4) Determining user's needs

I. Application of Structured Methodology--some pitfalls and stopgaps.

J. Beyond the logical model: or, Designing what has been analyzed.

1) Objectives of design
2) Changeability
3) Performance tuning
4) Top Down Developement

K. Summary/Review/Test

VIII. Bibliography

Milton J. Alexander, Information Systems Analysis: Theory and Applications, SRA, 1974.

Jerry Atwood, The Systems Analyst; How to Design Computer-Based Systems, Hayden, 1977.

J. E. Bingham, G. W. Davis, A Handbook of Systems Analysis, Halsted, 1973.

Bibliography Continued

H. D. Clifton, Systems Analysis for Business Data Processing, (revised ed.), Petrocelli, 1974.

Robert J. Condon, Data Processing Systems Analysis and Design, Reston, 1974.

Alan Daniels, Donald Yeates, (eds.), Systems Analysis, SRA, 1971.

W. Everling, Exercises in Computer Systems Analysis, Springer, 1972.

P. Faurre, M. Depreyrot, Elements of System Theory, North Holland, 1976.

John M. Fitzgerald, Andra F. Fitzgerald, Fundamentals of Systems Analysis, Wiley, 1973.

Marvin Gore, John Stubbe, Elements of Systems Analysis for Business Data Processing, Brown, 1975.

Paul Gross, Robert D. Smith, Systems Analysis and Design for Management, Dun, 1976.

Harry Katzan, Jr., Systems Design and Documentation: An Introduction to the HIPO Method, Van, 1976.

Alton R. Kindred, Data Systems and Management: An Introduction to Systems Analysis and Design, PH, 1973.

George J. Klir, (ed.), Trends in General Systems Theory, Wiley, 1972.

Richard W. Lott, Basic Systems Analysis, Canfield, 1971.

C. T. Meadow, The Analysis of Information Systems, (2nd ed.), Wiley, 1973.

M.D. Mesarovic, Y. Takahara, General Systems Theory: Mathematical Foundations, Academic, 1974.

Jerome T. Murray, Systems Analysis and Design in an IBM Environment, McGraw, 1973.

Stanford L. Optner, Systems Analysis for Business Management, (3rd ed.), PH, 1974.

Brian Rothery, Art of Systems Analysis, PH, 1971.

Bernard H. Rudrick, Systems Analysis for Effective Planning: Principles and Cases, Wiley, 1969.

Bibliography Cont.

Philip C. Semprevivo, Systems Analysis, SRA, 1976.

Gary Shelly, Tom Cashman, Business Systems Analysis and Design, Anaheim, 1975.

Gerald A. Silver, Joan B. Silver, Introduction to Systems Analysis, PH, 1976.

Robert J. Thierauf, Systems Analysis and Design of Real-Time Management Information Systems, PH, 1975.

Leon Youssef, Systems Analysis and Design, Reston, 1975.

SUBJECT:	Course Guide
COURSE NUMBER:	CS 205
COURSE NAME:	Computational Statistics
DATE OF APPROVAL	
CREDIT:	3 Semester Hours
PREREQUISITES:	Calculus II
TERM:	Fall
MAJOR:	Elective
MINOR:	Elective
GENERAL STUDIES:	No

COURSE DESCRIPTION:

Statistical topics include measures of central tendency and deviation with grouped and ungrouped data, frequency graphs, probability formulas, mean and variance of discrete and continuous random variables, binomial and normal distributions, confidence intervals, and hypothesis testing for means and proportions. Computer Science topics include flowcharting and coding in the Basic Language as well as the use of prepackaged programs for most of the statistical topics.

I. Aims and Objectives:

1. To introduce those elements of the Basic Language required to write programs for the statistical topics and to understand and be able to use the prepackaged programs.

2. To teach the statistical topics while emphasizing that the computer programs are a tool to save labor but not a substitute for understanding the purpose and nature of the topics.

3. To provide laboratory experience in using prepackaged programs and self-generated programs for statistical topics.

II. Content and Scope of the Course:

A. Using the Computer

1. Flowcharts
2. Basic Language

B. Descriptive Statistics

1. Measures of Central Tendency - Ungrouped Data
2. Measures of Deviation - Ungrouped Data
3. Descriptive Statistics - Grouped Data
4. Graphing Frequency Distributions
5. Related Prepackaged Programs

C. Counting Formulas

1. Permutations
2. Combinations
3. Binomial Theorem
4. Related Prepackaged Programs

D. Probability Formulas

1. Sample Spaces and Events
2. Conditional Probability
3. Bayes' Theorem
4. The Binomial Experiment

E. Random Variables and Special Distributions

1. Means and Variance of Discrete Random Variables
2. Binomial Distribution
3. Normal Distributions
4. Related Prepackaged Programs

F. Hypothesis Testing

1. The Basic Idea of Inferential Statistics
2. Confidence Intervals
3. One-Tail Test Versus Two-Tail Test
4. Difference between Sample Means and Population Means
5. Difference Between Sample Means
6. Test for Proportions
7. Related Prepackaged Programs

III. Procedures, Techniques, and Methods:

1. Lectures covering the theory and applications
2. Assignment of problems
3. Modifications of prepackaged programs
4. Actual use of programs on the computer

VI. Instructional Materials:

1. Textbook
2. Computer terminal
3. Prepackaged programs

V. Basic Requirements:

1. Periodic quizzes
2. Class participation
3. Final examination
4. Student - generated programs
5. Results from running prepackaged programs
6. Project to collect data and analyze
7. Above requirements made known during first lecture period

VI. Basic Text:

Elementary Computer-Assisted Statistics, Revised Ed.; Frank Scalzo and Rowland Hugh;(Van Nostrand Reinhold Company) 1978.

VII. Bibliography

Frank Scalzo, Rowland Hughes, Elementary Computer-Assisted Statistics, Petrocelli, 1976.

SUBJECT:	Course Guide
COURSE NUMBER:	CS 206
COURSE NAME:	Computational Linear Algebra
DATE OF APPROVAL:	
CREDIT:	3 Semester Hours
PREREQUISITES:	Computer Science, Pre-Calculus or Equivalent
TERM:	
MAJOR:	Elective
MINOR:	Elective
GENERAL STUDIES:	No

COURSE DESCRIPTION:

Topics include basic operations on and with matrices, determinents, computation of inverses, Gaussian elimination, systems of equations, vector spaces, linear programming, and eigenvalue problems. Numerical methods, computer arithmetic, convergence of an algorithm, and efficiency of a computational method will be discussed as they relate to the computer programs used to expand upon the linear algebra theory. APL will be used as a programming language.

I. Aims and Objectives:

1. To introduce enough APL to perform simple operations with matrices on the computer
2. To develop the basic theory of linear algebra
3. To expand upon the basic theory through emphasis on computational aspects
4. To provide some laboratory experience in using the computer terminal to solve linear algebra problems

II. Content and Scope of the Course:

A. Overview

1. Basic Theorems
2. Mathematical Induction

B. Matrices

1. Basic Definitions

2. Matrix Products
3. Diagonal, Symmetric, and Hermitian Matrices
4. Matrix Partitioning
5. APL Program to Perform Operations

C. Determinants

1. Properties
2. Computation of Determinents

D. Solutions of Systems of Equations

1. Elimination Methods
2. Computation of Inverse
3. Iterative Methods
4. Convergence of Iteration
5. Pivoting in Gaussian Elimination
6. Iterative Improvement of a Solution
7. Rank
8. Linear Independence
9. Homogeneous Systems

E. Linear Programming (Optional)

1. Graphical Solution
2. Simplex Method

F. Vector Spaces

1. Definition
2. Dimension and Basis

G. Eigenvalue Problems

1. Unitary and Orthogonal Matrices
2. Hermitian and Real Symmetric Matrices
3. Jordan Cannonical Form
4. Numerical Calculation of Eigenvalues

III. Procedures, Techniques, and Methods:

1. Lectures covering the theory and areas of application
2. Assignment of Problems
3. APL programs for linear algebra problems
4. Use of the computer to solve linear algebra problems

IV. Instructional Materials:

1. Textbook
2. Computer Terminal

V. Basic Requirements:

1. Periodic quizzes over material
2. Class participation

3. Final examination
4. Project to solve system of equations using computer program
5. Above requirements made known during first lecture period

VI. Basic Text:

Computational Matrix Algebra, David Steinberg, McGraw-Hill Book Co. 1974

VII. Bibliography

James R. Branch, Donald J. Rose, (ed.), Sparse Matrix Computations, Academic, 1976.

Noel Gastinel, Linear Numberical Analysis, Academic, 1970.

H. Schwartz, et al, Numerical Analysis of Symmetric Matrices, PH, 1973.

G. W. Stewart, Introduction to Matrix Computations, Academic, 1973.

SUBJECT:	Course Guide
COURSE NUMBER:	CS 207
COURSE NAME:	Digital Electronics
DATE OF APPROVAL:	
CREDIT:	3 Semester Hours
PREREQUISITES:	Physics II, Computer Science I
MAJOR:	Elective
MINOR:	Elective
GENERAL STUDIES:	No

COURSE DESCRIPTION:

Principles of Boolean Algebra; Digital gating circuits including NAND and NOR gates. Flip flops; shift registers; counters. Arithmetic circuits. Implementation in terms of integrated circuits.

I. Aims and Objectives:

1. To acquaint students with the logic basis of computer circuits and their relationship to current hardware.
2. To give students some practice in assembling circuits.

II. Content and Scope of the Course:

A. Introduction to Boolean Algebra
   1. Logic functions and conventions
   2. Boolean Operations
   3. Associative and Distributive Laws
   4. DeMorgan's Law
   5. Logical gates
   6. Relay gates

B. Combinational Logic
   1. Standard forms - MAXTERM and MINTERM
   2. Equivalence of Standard forms
   3. Algebraic reduction of logic expression
   4. Karnaugh maps
   5. Standard forms using NAND and NOR logic

C. Logic Families

1. Diode gates
2. Discrete gates
3. Integrated circuits
4. Resistor - transistor logic
5. TTL gates
6. MOS gates

D. Implementation of combinational logic circuits using TTL and MOS gates

E. Flip - Flops

1. Basic RS flip-flop
2. D flip flops
3. Clocked (gated) flip-flops
4. JK and Master-slave flip-flops
5. Glitches

F. Counters

1. Binary counters
2. Divide by N counters
3. Synchronous counters
4. Commercial counters

G. Number Systems and Codes

1. Binary related number systems
2. Coded number systems
3. Encoders and Decoders

H. Shift Registers

1. The basic shift register
2. Right/Left shift register
3. Serial in, parallel out registers
4. Parallel in, parallel out registers

I. Arithmetic circuits

1. Review of binary arithmetic
2. Adders and subtracters

III. Procedures, Techniques, and Methods:

1. Lectures
2. Laboratory work

IV. Instructional Materials:

1. Textbook
2. Lab book
3. Logic circuits

V. Requirements:

1. Quizzes
2. Lab workshop
3. Class participation
4. Final Examination

VI. Text:

Joseph D. Greenfield, Practical Digital Design Using ICs. (John Wiley)

VII. Bibliography

Thomas C. Bartee, Digital Computer Fundamentals, (4th ed.), McGraw, 1977.

Nripendra N. Biswas, Introduction to Logic and Switching Theory, Gordon, 1975.

Thomas R. Blakeslee, Digital Design with Standard MSI and LSI, Wiley, 1975.

Paul M. Chirlian, Analysis and Design of Digital Circuits and Computer Systems, Matrix, 1976.

John C. Cluley, Computer Interfacing and on-line Operations, Crane, 1975.

David J. Comer, Modern Electronic Circuit Design, AW, 1976.

Wendell H. Cornetet, Jr., Frank E. Barrocletti, Electronic Circuits by Systems and Computer Analysis, McGraw, 1975.

John A. Dempsy, Experimentation with Digital Electronics, AW, 1976.

Arthur Friedman, P.R. Menon, Theory and Design of Switching Circuits, Comp. Sci., 1975.

Arthur D. Friedman, Logical Design of Digital Systems, Comp. Sci., 1975.

Caxton C. Foster, Content Addressable Parallel Processors, Van, 1976.

Bibliography Continued

S. H. Fuller, Analysis of Drum and Disk Storage Units, Springer, 1975.

Victor Grimich, Horace G. Jackson, Introduction to Integrated Circuits, McGraw, 1975.

H. W. Gschwind, E.J. McCluskey, Design of Digital Computers: An Introduction, (2nd ed.) Springer, 1975.

Eugene R. Hnatek, A User's Handbook of Semi-conductor Memories, Wiley, 1977.

Randall W. Jensen, Lawrence P. McNamee, Handbook of Circuit Analysis Languages and Techniques, PH, 1975.

George Kostopoulos, Digital Engineering, Wiley, 1975.

Samuel C. Lee, Digital Circuits and Logic and Design, PH, 1976.

Mitchell P. Marcus, Switching Circuits for Engineers, (3rd ed.), PH, 1975.

Frederic J. Mowle, Systematic Approach to Digital Logic Design, AW, 1976.

H. Troy Nagle, Jr., B.D. Carroll, J. David Irwin, An Introduction to Computer Logic, PH, 1975.

A. Potton, An Introduction to Digital Logic, Hayden, 1975.

C.H. Roth, Fundamentals of Logic Design, West, 1975.

Samuel D. Stearns, Digital Signal Analysis, Hayden, 1975.

Earl E. Swartzlander, (ed.), Computer Design Development: Principal Papers, Hayden, 1976.

H. Taug, D. Schully, Digital Electronics and Systems, McGraw, 1976.

S. Thelliez, Introduction to the Study of Ternary Switching Structures, Gordon, 1975.

John Wakerly, Logic Design Projects Using Standard Integrated Circuits, Wiley, 1976.

J. F. Wakerly, Error Detecting Codes, Self-Checking Circuits and Applications, Elsevier, 1977.

Bibliography Cont.

Charles L. Wilkins, Sam P. Perone, Charles E. K. Lopfenstein, Robert C. Williams, Donald E. Jones, Digital Electronics and Laboratory Computer Experiments, Plenun, 1975.

Susan Wooldridge, Computer Output Design, Petrocelli, 1975.

SUBJECT:	Course Guide
COURSE NUMBER:	CS 208
COURSE NAME:	File Processing
DATE OF APPROVAL:	
CREDIT:	3 Semester Hours
PREREQUISITES:	Introduction to Computer Science I and II
TERM:	Fall
MAJOR:	Elective
MINOR:	Elective
GENERAL STUDIES:	No

COURSE DESCRIPTION:

The course covers characteristics of, and utilization of different levels storage devices. Topics include file design, sequential access, random access, file input-output.

I. Objectives:

The objectives of this course are:

1. To introduce concepts and techniques of structuring data on bulk storage devices.
2. To provide experience in the use of bulk storage devices.
3. To provide the foundation for applications of data structures.

II. Content and Scope of the Course:

A. File Processing Environment (5%)

1. Definition of record
2. Blocking, compaction
3. Data base concepts

B. Sequential access (30%)

1. Physical characteristics
2. Sort/merge algorithms
3. File manipulation

C. Data structures (20%)

1. Algorithms for linked lists
2. Binary, B-trees, B*-trees, and AVL-trees
3. Algorithms for traversing and balancing trees

D. Random Access (35%)

1. Physical characteristics
2. Algorithms for inverted lists, multilist, indexed sequential, and hierarchial structures

E. File I/O (10%)

1. Control systems and utility routines
2. Space allocation and cataloging

III. Procedures, Techniques and Methods:

1. Lectures, covering theory and practice
2. Verification of algorithms using facilities of the Center for Computing Services
3. Case studies
4. Assignment of problems

IV. Instructional Materials:

1. Textbooks
2. IBM 370 Computing System (batch and interactive)

V. Basic Requirements:

1. Verification of algorithms on the computer
2. Class participation
3. Final Examination
4. Above requirements made known during first week of class

VI. Basic Text:

Davis, C. B., Management Information Systems: Conceptual Foundations, Structure and Development, (McGraw-Hill) 1974.

VII. Bibliography

Ivan Flores, Computer Sorting, PH, 1969.

Ivan Flores, Data Structure and Management, (2nd ed.), PH, 1977.

J. M. Foster, List Processing, Elsevier, 1968.

T. R. Gildersleeve, Design of Sequential File Systems, Wiley, 1971.

William Haseman, Andres Whinston, Introduction to Data Management, Irwin, 1977.

D. R. Judd, Use of Files, Elsevier, 1972.

Donald E. Knuth, The Art of Computer Programming, Vol. III: Sorting and Searching, AW, 1973.

David Lefkovitz, File Structures for on-line Systems, Hayden, 1969.

Harold Lorin, Sorting and Sort Systems, AW, 1975.

SUBJECT:	Course Guide
COURSE NUMBER:	CS 301
COURSE NAME:	Machine Organization
DATE OF APPROVAL:	
CREDIT:	3 Semester Hours
PREREQUISITES:	Assembly Language
TERM:	Fall
MAJOR:	Elective
MINOR:	Elective
GENERAL STUDIES:	No

COURSE DESCRIPTION:

Topics include an axiomatic and formal development of the basic theorems of Boolean Algebra with interpretations in logic, set theory, and switching circuits. Realizations of Boolean function, binary arithmetic and coding, computer components such as the adder, shifter registers, comparators, selection trees, memory elements, counters, etc., and total system organization and machine programming.

I. Aims and Objectives:

1. To introduce the student to the basic elements and logic design techniques of systems making up the components of a computer.
2. To show the relationship between an abstract mathematical system and its different interpretations.
3. To show the connection between the components of the computer hardware system and the features of a machine language programming system.
4. To provide laboratory experience wiring logic circuits using logic circuit boards.

II. Content and Scope of the Course:

A. The Algebra of Logic (20%)

1. Truth-function operations
2. Connectives
3. Statement forms

4. Truth tables
5. Tautologies and contradictions
6. Logical implication and equivalence

B. The Algebra of Sets (10%)

1. Sets, set relations, set operations
2. Propositional logic and the algebra of sets

C. Boolean Algebra (20%)

1. Operations
2. Axioms for a Boolean algebra
3. Sub-algebras, partial orders
4. Normal forms
5. Isomorphisms

D. Switching Circuits and Logic Circuits (20%)

1. Switching circuits
2. Simplification of circuits
3. Logic circuits
4. The binary number system
5. Minimization

E. Computer Components (20%)

1. Majority circuit
2. Odd parity circuit
3. Adder
4. Memory elements
5. Shift register
6. Counters
7. Selection trees
8. Decoders
9. Clock

F. The computer (10%)

III. Procedures, Techniques, and Methods:

1. Lectures covering the theory and areas of application
2. Assignment of problems
3. Construction of logic circuits using logic circuit board
4. Study of a microcomputer system

IV. Instructional Materials:

1. Textbook
2. Logic circuit boards
3. Microcomputer

V. Basic Requirements:

1. Periodic quizzes over material
2. Maintenance of a note book
3. Class participation
4. Final examination
5. Projects on logic board
6. Above requirements made known during first lecture period

VI. Basic Text:

Boolean Algebra and Switching Circuits by Elliot Mandelson, (McGraw-Hill Book Co.) 1970. Price $10.00

VII. Bibliography

A.M. Abd-Alla, A.C. Meltzer, Principles of Digital Computer Design, PH, 1976.

C. Gordon Bell, Alan Newell, Computer Structures: Examples and Readings, McGraw, 1970.

C. Gordon Bell, J. Grason, Alan Newell, Designing Computers and Digital Systems, Digital, 1972.

Gerritt A. Blaau, Digital System Implementation: A Controlled Approach, PH, 1976.

Melvin Breuer (ed.), Design Automation of Digital Systems: Vol. I, Theory and Techniques, PH, 1972.

Melvin A. Breuer (ed.), Digital System Design Automation: Language, Simulation and Data Base, Comp. Sci., 1975.

J. A. Brzozowski, Michael Yoeli, Digital Networks, PH, 1976.

Yaohan Chu, Introduction to Computer Organization, PH, 1970.

Harry J. Gray, High-Speed Digital Memories and Circuits, AW, 1976.

Harry Katzan, Jr., Computer Organization and the System/370, Van, 1971.

Raymond M. Kline, Digital Computer Design, PH, 1970

Teuvo Kohonen, Digital Circuits and Devices, PH, 1972.

Bibliography Continued

Elliot I. Organick, Computer System Organization: The B5700/B6700 Series, Academic, 1973.

James L. Peterson, Computer Organization and Assembly Language Programming, Academic, 1978.

V. Thomas Rhyne, Fundamentals of Digital Systems Design, PH, 1973.

R. K. Richards, Digital Design, Wiley, 1971.

H. H. Rosenbrock, Computer Aided Control System Design, Academic, 1974.

Mischa Schwartz, Computer-Communication Network Design and Analysis, PH, 1977.

M. E. Sloan, Computer Hardware and Organization, SRA, 1976.

Andrew S. Tanenbaum, Structured Computer Organization, PH, 1976.

W. M. Van Cleemput, Computer Aided Design of Digital Systems: A Bibliography, Comp. Sci., 1976.

S. J. Waters, Introduction to Computer Systems Design, Hayden, 1975.

M. Wells, Computing Systems Hardware, Cambridge, 1976.

D. J. Woolons, Introduction to Digital Computer Design, McGraw, 1973.

Elliott Mendelson, Boolean Algebra and Switching Circuits Including 150 Solved Problems, McGraw, 1970.

Amar Mukhopadhyay, (ed.) Recent Developments in Switching Theory, Academic, 1971.

S. H. Unger, Asynchronous Sequential Switching Circuits Wiley, 1969.

Paul E. Wood, Jr., Switching Theory, McGraw, 1968.

SUBJECT:	Course Guide
COURSE NUMBER:	CS 302
COURSE NAME:	Operating Systems
DATE OF APPROVAL:	
CREDIT:	3 Semester Hours
PREREQUISITES:	Assembly Language Data Structures
TERM:	Fall
MAJOR:	Elective
MINOR:	Elective
GENERAL STUDIES:	No

COURSE DESCRIPTION:

A description of the basic functions of an operating system (resource allocation, file management, I/O control, etc.) A survey of the current operating systems and new concepts in operating systems design (independent processes, control levels, software tools). Apply the new concepts in operating system design by defining and implementing a set of software tools on an existing operating system.

I. Objectives:

The objective of this course is to bring the students to grips with the problems of systems programming. Current operating systems will be discussed in terms of their impact on the user and the systems programmer. The advantages of the new concepts in operating system design will be demonstrated.

The course will use the idea of software tools to demonstrate the nature of systems programming. A software tool is defined as a program which solves a well-defined general purpose problem. Software tools range in size from language compilers to small file listing programs. It will be shown that a major portion of an operating system can be considered software tools.

The students will evaluate an existing operating system,

define a set of tools the system needs and then design, implement, test and document the tools. The tools must be practical products that can be used by the college. This will force the student to consider how others will use the tools. It also provides motivation for the student and a useful product for the college.

II. Content and Scope of Course:

A. Review of Data Structures

1. Single Cell
2. Multi-Cell Mapped
3. Multi-Cell Linked
4. Addressing Structures

B. Basic Operating Systems Functions

1. I/O Control
2. File Management
3. Memory Management
4. Time Management
5. Process Management

C. Traditional Operating Systems

1. Single Task
2. Multi-Task (fixed, variable)
3. Time Share
4. Real Time
5. Virtual Storage Systems
6. Virtual Machine Concept
7. Multiprocessors and Networks

D. New Operating System Design Concepts

1. Criteria for New Operating Systems
2. Basic Principles of New Design
3. Some Examples UNIX, MERT, T.H.E.
4. Comparison to Existing Operating Systems

E. Software Tools

1. Definition
2. Examples
3. Writing specifications for a Software Tool
4. Design (completeness property)
5. Coding and Testing
6. Evaluation
7. Documentation
8. Maintenance

III. Procedures, Techniques and Methods:

1. Lectures, reading, examples and demonstrations
2. "Hands on" programming experience
3. Group discussions

IV. Basic Requirements for Successful Completion:

1. Paper evaluating an operating system
2. Paper defining a set of software tools
3. Talk on the design of the tools
4. Meta-language description of the programs
5. Actual programs with examples
6. Documentation and maintenance procedures
7. Paper on the results of the evaluation of the tool
8. Tests on key aspects of the course
9. Required readings:
   a. Operating Systems Principles, by P. Brinch Hansen (Prentice-Hall) 1973
   b. The logical Design of Operating Systems, by Alan C. Shaw (Prentice-Hall) 1974
   c. Systems Programming, by David K. Hsiao (Addison-Wesley) 1975
   d. Principles of Systems Programming, by R.M. Graham (Wiley and Sons) 1975
   e. Software Tools, by Kernighan and Plauger (Addison-Wesley) 1976
   f. The Mythical Man-Month, by F. P. Brooks (Addison-Wesley) 1975

V. Bibliography

D. W. Barron, Computer Operating Systems, Wiley, 1971.

William T. Batten, Understanding the IBM 360 and 370, PH, 1971.

G. M. Bull, S.F.G. Packham, Time-Sharing Systems, PH, 1973.

Harol Carroll, OS Data Processing with Review of OS/VS, Wiley, 1974.

Edward G. Coffman, Jr., Peter J. Denning, Operating Systems Theory, PH, 1973.

Edward G. Coffman, Jr., (ed.), Computer and Job/Shop Scheduling Theory, Wiley, 1976.

Leo J. Cohen, Operating Systems Analysis and Design, Hayden, 1971.

Bibliography Continued

A. J. Cole, Macro Processors, Cambridge, 1976.

A. Colin, Introduction to Operating Systems, Elsevier, 1971.

C. Cuttle, P.B. Robinson, (eds.), Executive Programs and Operating Systems, Elsevier, 1970.

William S. Davis, Operating Systems: A Systematic View, AW, 1977.

John J. Donovan, Systems Programming, McGraw, 1972.

Ivan Flores, Operating Systems for Multiprogramming with a Variable Number of Tasks, Allyn, 1972

D. E. Freeman, O.R. Perry, I/O Design: Data Management in Operating Systems, Hayden, 1977.

E. Gelenble, C. Kaiser, (eds.), Operating Systems,

Terry Gibbons, Integrity and Recovery in Computer Systems, Hayden, 1976.

R. M. Graham, Principles of Systems Programming, Wiley,1975.

A. N. Habermann, Introduction to Operating System Design, SRA, 1976.

Maurice H. Halstead, A Laboratory Manual for Compiler and Operating System Implementation, Elsevier, 1974.

Per Brinch Hansen, Operating System Principles, PH, 1973.

Per Brinch Hansen, The Architecture of Concurrent Programs, PH, 1977.

C.A.R. Hoare, R. H. Perrott, Operating Systems Techniques, Academic, 1972.

Harry Katzan, Jr., Advanced Programming: Programming and Operating Systems, Van, 1970.

Harry Katzan, Jr., Computer Systems Organization and Programming, SRA, 1976.

Leonard Kleinrock, Queueing Systems, Vol. II, Computer Applications, Wiley, 1975.

A. M. Lister, Fundamentals of Operating Systems, Hayden, 1975.

Bibliography Cont.

Stuart E. Madnick, John J. Donovan, Operating Systems, McGraw, 1974.

R. M. McKeag, R. Wilson, Studies in Operating Systems, Academic, 1976.

Elliot I. Organick, The Multics System: An Examination of its Structures, MIT, 1972.

D. Rindfleisch, Debugging 360/370 Programs Using OS and VOS Storage Dumps, PH, 1976.

Anthony P. Sayres, (ed.), Operating Systems Survey, Petrocelli, 1971.

Alan C. Shaw, The Logical Design of Operating Systems, PH, 1974.

Dionysios C. Tsichritzis, Philip Berstein, Operating Systems, Academic, 1974.

M. V. Wilkes, Time-Sharing Computer Systems, 3rd Rev. ed.), Elsevier, 1975.

SUBJECT:	Course Guide
COURSE NUMBER:	CS 303
COURSE NAME:	Data Base Design
DATE OF APPROVAL:	
CREDIT:	3 Semester Hours
PREREQUISITES:	Total Systems Development Concepts Data Structures
MAJOR:	Elective
MINOR:	Elective
GENERAL STUDIES	No

COURSE DESCRIPTION:

An introduction to basic terminology associated with computer data base technology. Topics in data base concepts include classical data storage and retrieval methods, information needs within a business organization, basics of access methods, basics of data structures, basics of a data base management system, data base design methodologies, and a brief tutorial on data normalization.

I. General Aims:

1. To expand the students vocabulary to include terminology commonly associated with computer data base systems.
2. To help the student see the close relationship between intended uses of data and necessary structures of the data.
3. To give the student data primitives as a foundation for data base design.

II. Specific Objectives:

1. Compare the benefits of a data base with other methods of data storage and retrieval.
2. Identify and define the terminology associated with computerized data base, specifically including:
   A. Concepts of information requirements.
   B. Concepts of data storage and retrieval requirements.
   C. Concepts and components of a Data Base Management System (DBMS).

3. Describe the tasks and responsibilities necessary to complete each of the phases of the data base development cycle.
4. Describe the three normal forms of data.

III. Methods of Evaluation:

1. Final examination
2. Class participation
3. Assignment of specific problems to be returned

IV. Course Content:

A. History of the Data Base Concept

1. Manual data storage and retrieval
2. Computer-based "batch" storage and retrieval
3. Computer-based "on line" storage and retrieval
4. The data base concept

B. Concepts of the Data Base

1. Information needs within a business organization
2. Basic data structures
3. Access methods primer

C. Components of a Data Base Managment System

1. Data Description Language
2. Data Manipulation Language
3. Workings of a data base management system
4. Associated components of an information system

A. Query packages
B. Recovery
C. Security

D. Data Base Design

1. Four-step approach to data base design
2. First normal step
3. Second normal step
4. Third normal step

V. Procedures Techniques and Methods:

1. Lectures covering theory
2. Class discussions about specific examples
3. Assignment of problems/ensuing discussion of answers

VI. Basic Text:

James Martin, Computer Database Organization, (Prentice-Hall, Englewood Cliffs, NJ) 1975.

VII. Bibliography

Richard A. Bassler, Jimmie Logan, The Technology of Data Base Management Systems, (3rd ed.), College, 1976.

R. L. Bisco, Data Bases Computers and the Social Sciences, Wiley, 1970.

Carl Cagan, Data Management Systems, Wiley, 1973.

C. J. Date, An Introduction to Database Systems, AW, 1975.

Sakti P. Ghosh, Data Base Organization for Data Management, Academic, 1976.

F. Gruenberger, (ed.), Critical Factors in Data Management, PH, 1969.

William C. House, (ed.), Interactive Decision Oriented Data Base Systems, Petrocelli, 1977.

Donald A. Jardine, (ed.), Data Base Management Systems, North Holland, 1974.

R. A. Kaiman, Structured Information Files, Wiley, 1973.

Harry Katzan, Jr., Computer Data Management and Data Base Technology, Van, 1975.

David Kromeke, Introduction to Data Base Systems, McGraw, 1977.

Borje Langefors, Information Systems Architecture, Petrocelli, 1975.

B. Langefors, Kjell Samuelson, Information and Data in Systems, Petrocelli, 1977.

John K. Lyon, An Introduction to Data Base Design, Wiley, 1971.

James Martin, Principles of Data Base Management, PH, 1976.

J. Martin, Computer Data Base Organization, (2nd ed.), PH, 1977.

Bibliography Continued

B. Mittman, L. Borman, (eds.), Personalized Data Base Systems, Wiley, 1975.

Joseph I. Naus, Data Quality and Editing, Dekker, 1975.

Vivian Prothro, Information Management Systems: Data Base Systems, Petrocelli, 1976.

Randall Rustin, Data Base Systems, PH, 1972.

R. Clay Sprowls, Management Data Bases, Wiley, 1976.

Bo Sundgren, Theory of Data Bases, Petrocelli, 1975.

D. Tsichritzis, F.H. Lochovsky, Data Base Management Systems, Academic, 1977.

Gio Wiederhold, Data Base Design, McGraw, 1977.

SUBJECT:	Course Guide
COURSE NUMBER:	CS 304
COURSE NAME:	Computer Architecture
DATE OF APPROVAL:	
CREDIT:	3 Semester Hours
PREREQUISITES:	
MAJOR:	Elective
MINOR:	Elective
GENERAL STUDIES:	No

COURSE DESCRIPTION:

This course shows how logic devices are assembled to form a computer system. A discussion of CPU, memory and I/O structures. Communication between components. Techniques for speeding up computers, such as pipelining and parallel processing.

I. Aims and Objectives:

To acquaint students with the architecture of various types of computers.

II. Content and Scope of the Course:

A. An Overview of Computer Systems
   1. Uniprocessors
   2. Multiprocessors
   3. Pipeline processors
   4. Parallel processors

B. Review of Logic Circuits

C. Memory Mechanisms
   1. Magnetic storage
   2. RAMS, ROMS, EROMS, PROMS
   3. Exotic devices

D. Implementing Memory Systems
   1. Co-ordinate Addressed Storage

2. Content Addressable Storage
3. Pushdown Stacks
4. Dynamic RAMs
5. Magnetic Discs, Drums and Tape

E. Control Units

1. Instruction formats
2. Microprogramming
3. Control Unit Processing
4. Address Processing
5. Branch and Interupt Processing

F. I/O Handling

1. Interupts
2. Data Channels and Controllers
3. Handshaking

G. Minicomputer Structures

H. Very Large Computers

1. Increasing Memory Speeds
2. Increasing Processor Speeds
3. Pipelining
4. Parallelism

III. Procedures, Techniques and Methods:

1. Lectures

IV. Instructional Materials:

1. Textbook

V. Basic Requirements:

1. Quizzes
2. Homework problems
3. Final examination

VI. Text:

C. C. Foster, Computer Architecture, 2nd ed. (Van-Nostrand Reinhold.) 1970.

VII. Bibliography

Yaohan Chu, (ed.), High-Level Language Computer Architecture, Academic, 1975.

Philip Enslow (ed.), Multiprocessors and Parallel Processing, Wiley, 1974.

Caxton C. Foster, Computer Architecture, Van, 1970.

L. D. Hoggs, D.J. Theis, Joel Trimble, Harold Titus, Ivar Highberg, Parallel Processors Systems, Technologies and Application, Spartan, 1971.

S. H. Lavington, Processor Architecture, Hayden, 1976.

Harold Lorin, Parallelism in Hardware and Software: Real and Apparent Con-currency, PH, 1972.

M. Morris Mano, Computer System Architecture, PH, 1976.

Harold S. Stone, Introduction to Computer Architecture, SRA, 1975.

Kenneth J. Thurber, Large-Scale Computer Architecture: Parallel & Associative Processors, Hayden, 1976.

SUBJECT:	Course Guide
COURSE NUMBER:	CS 305
COURSE TITLE:	Programming Languages
CREDIT:	3 Semester Hours
DATE OF APPROVAL:	1968
TERM:	Spring
MAJOR:	Elective
MINOR:	Elective
GENERAL STUDIES:	No
PREREQUISITES:	Assembly Language, Data Structures

COURSE DESCRIPTION:

Formal definition of programming languages. Introduction to list processing, string manipulation, data description, and simulation languages. Definition of formal grammars. Semantics of grammatical constructs.

I. Aims and Objectives:

The purpose of this course is to present a systematic approach to programming languages. The student is expected to become familiar with several different level languages. Selected topics from the theory of formal languages and syntactic analysis are also discussed.

II. Course Content:

A. Structure of Simple Statements

1. Informal syntax and semantics of expressions and statements
2. Translation between infix, prefix, and postfix notation
3. Pushdown stores for translation and execution
4. Precedence hierarchy of operations
5. Backus normal form
6. Precedence relation, precedence grammars

B. Structure of Algorithmic Languages

1. Grouping statements, declarations, scopes, blocks
2. Function and statement type procedures, parameters
3. Binding time of program constituents
4. Recursive procedures
5. Storage allocations for blocks
6. Coroutinize tasks, interrupt specifications, and control structures
7. Syntactic specification of procedures
8. Formal semantics
9. Generalized arrays

C. List Processing and String Manipulation Languages

1. List Structures
2. String Structures

D. Topics in Programming Languages

1. Simulation languages
2. Formal definition of languages

III. Instructional Materials:

1. Textbook
2. Computer facilities of the Center for Computer Services

IV. Techniques and Methodology:

1. Lectures covering the theory and areas of application
2. Guest lecturers discussing their specialties
3. Assignment of programming projects
4. Independent projects

V. Course Requirements:

1. Class participation
2. Successful run of specified number of programs
3. Presentation of a programming project to class
4. Final examination

VI. Basic Texts:

1. Concepts of Programming Languages by Mark Elson; (Science Research Associates, Inc., 1973)
2. Computer Systems Organiaztion and Programming by Harry Katzan; (Science Research Assoc., 1976)

VII. Bibliography

D. J. Armor, A.S. Couch, Data-Text Primer, Free, 1972.

A.D. Bajpai, H.W. Pakes, R.J. Clarke, J.M. Doubleday, T. J. Stevens, FORTRAN and ALGOL: A Programmed Course for Students of Science and Technology, Wiley, 1972.

D. W. Barron, An Introduction to the Study of Programming Languages, Cambridge, 1977.

G.D. Brown, FORTRAN to PL/1 Dictionary,PL/1 to FORTRAN Dictionary, Wiley, 1975.

Richard Conway, David Gries, E.C. Zimmerman, A Primer on PASCAL, Winthrop, 1976.

John J. Donovan, S. E. Madnick, Software Projects, McGraw, 1977.

Mark Elson, Concepts of Programming Languages, SRA, 1973.

B.A. Galler, A.J. Perlis, A View of Programming Languages AW, 1970.

M.A. Gaurilov, A.D. Zakrevskii (eds.), A Programming Language for Logic and Coding Algorithmic, Academic, 1969.

Richard L. Guertin, Introduction to PL 360 Programming, Wadsworth, 1977.

H.S.Heaps, An Introduction to Computer Languages, PH, 1972.

Lee E. Heindel, Jerry T. Robert, LANG-PAK-An Interactive Language Design System, Elsevier, 1975.

B.A. Higman, A Comparative Study of Programming Languages, Elsevier, 1967.

M.A. Jackson, Principles of Program Design, Academic, 1975.

K. Jensen, N. Wirth, PASCAL-User Manual and Report, (corrected re-print of the 2nd ed.), Springer, 1976.

Harry Katzan, Jr., Introduction to Programming Languages, Petrocelli, 1973.

John E. Nicholls, The Structure and Design of Programming Languages, AW, 1975.

E.I. Organick, A.I. Forsythe, R.P. Plummer, Programing Language Structures, Academic, 1978.

W. W. Peterson, Introduction to Programming Languages,PH,1974.

Bibliography Continued

Terrence W. Pratt, Programming Languages: Design and Implementation, PH, 1975.

Saul Rosen, (ed.), Programming Systems and Languages, McGraw, 1967.

Jean E. Sammet, Programming Languages: History and Fundamentals, PH, 1969.

Peter C. Sanderson, Computer Languages: A Practical Guide to the Chief Programming Languages, Philosophical, 1970.

Lee David Schur, Time-Shared Computer Languages: An Introduction to Conversational Computing, AW, 1973.

Robert F. Steinhart, Seymour V. Pollack, Programming the IBM System/360, Holt, 1970.

Allen B. Tucker, Jr., Programming Languages, McGraw, 1977.

R. A. Vowels, ALGOL 60 and FORTRAN IV, Wiley, 1974.

Dale W. Washburn, Computer Programming: A Total Language Approach, Holt, 1970.

SUBJECT:	Course Guide
COURSE NUMBER:	CS 306
COURSE NAME:	Algorithm Analysis
DATE OF APPROVAL:	
CREDIT:	3 Semester Hours
PREREQUISITES:	Discrete Structures, Linear Algebra, Data Structures
TERM:	Fall
MAJOR:	Elective
MINOR:	Elective
GENERAL STUDIES:	No

COURSE DESCRIPTION:

Realistic models of computation used to study resource requirements. Topics include Turing machines, computation models and techniques, upper bounds, sorting, searching, matrix multiplications, fast Fourier transforms, Sauer bounds, NP-complete problems.

I. Objectives:

1. To enable students to determine resource requirements of computational problems
2. To enable the student to design resource-efficient algorithms
3. To enable students to give a precise analysis of the dynamic behavior of algorithms
4. To enable students to design for optimal solutions
5. To enable students to design algorithms for problems whose solution is intractable.

II. Course Content and Scope:

A. Background

1. Turing machine models and Markov algorithms
2. Random access models
3. Circuits and straightline programs
4. Complexity measures in various models
5. Computation techniques: divide and conquer, dynamic programming, linear recurrences, proofs of correctness

B. Upper Bounds

1. Data structure algorithms, binary search, union find problems, priority grieves
2. Radix sorting, sorting in time o, upper and lower bounds
3. Combinatiorial and graph algorithms
   a. Depth first search
   b. Graph isomorphism
   c. Path problems
4. String matching problems

C. Algebraic Algorithms

1. Matrix multiplication
   a. Strossen's algorithm
   b. Problems reducible to matrix multiplication
2. Fast Fourier transform
   a. Discrete fast Fourier transform
   b. Polynomial multiplication, interpolation
   c. Schonliagge-Stassen. integer multiplication

D. Lower Bounds

1. Diagonalization
2. Reducibilities
3. NP - complete problems
4. DSPACE, DTIME, NSPACE, NTIME hierarchies
5. Exponential time

III. Procedures, Techniques, and Methods:

1. Use of large realistic problems
2. Lectures covering the theory and areas of application
3. Verification of algorithms using the computer

IV. Instructional Materials:

1. Text books
2. Batch and interactive computing system

V. Basic Requirements:

1. Class participation
2. Final examination
3. Completion of assignments
4. Above requirements made known during first week of classes

VI. Basic Text:

Aho, J. Hopcroft, J.D. Ullman, The Design and Analysis of Computer Algorithms, (Addison-Wesley, 1974).

VIII. Bibliography

Alfred V. Aho, John E. Hopcroft, Jeffery D. Ullman, The Design and Analysis of Computer Algorithm, AW, 1974.

Mark A. Aiserman, Leonid A. Gusev, Lev I. Rozonoer, Irina M. Smirnova, Aleksey A. Tal, Logic, Automata and Algorithms, Academic, 1971.

S.E. Goodman, S.T. Hedetniemi, Introduction to the Design and Analysis of Algorithms, McGraw, 1977.

R.T. Gregory, D.L. Karney, A Collection of Matrices for Testing Computational Algorithms, Wiley, 1969.

S.A. Greiback, Theory of Program Structures; Schemes, Semantics, Verification, Springer, 1975.

Matthew S. Hecht, Flow Analysis of Computer Programs, Elsevier, 1977.

Leonard Kleinrock, Queueing Systems, Vol. I: Theory, Wiley, 1975.

Donald E. Knuth, The Art of Computer Programming, Vol. II: Seminumerical Algorithms, AW, 1969.

R. E. Miller, J.W. Thatcher, (eds.), Complexity of Computer Computations, Plenum, 1972.

Albert Nijenhuis, H.S. Wolf, Combinatorial Algorithms, Academic, 1975.

Edward M. Reingold, Jurg Neivergelt, Narsingh Deo, Combinatorial Algorithms, Theory and Practice, PH, 1977.

Jeffrey R. Spirn, Program Behavior: Models and Measurement, Elsevier, 1977.

J. F. Traub, (ed.), Algorithms and Complexity: Recent Results and New Directions, Academic, 1976.

SUBJECT:	Course Guide
COURSE NUMBER:	CS 307
COURSE NAME:	Numerical Analysis I
DATE OF APPROVAL:	
CREDIT:	3 Semester Hours
PREREQUISITES	Calculus I, II, Computer Science I, II
MAJOR:	Elective
MINOR:	Elective
GENERAL STUDIES:	No

COURSE DESCRIPTION:

An introduction to numerical methods. Topics in numerical analysis include error analysis, methods for solution of equations, polynomial approximation, interpolation, quadrature, solution of differential equations with initial conditions, matrix algebra and simultaneous equations.

I. General Aims:

1. To strengthen the students' grasp of basic notions in analysis and algebra, e.g., the idea of a sequence, limit, recursion relation, definite integral, matrix techniques in algebra.
2. To help the student see the connection between an algorithm as a computational procedure, and the mathematical foundations.
3. To help the student appreciate the type of algorithmic approach that enables a problem to be handled by a computer.

II. Specific Objectives:

1. To understand the nature of a recursive formula with specific examples used to solve certain classes of problems.
2. To see how a computational procedure is developed from the mathematical theory.

3. To learn basic principles of computation as an art in so far as it pertains to matters of precision, accuracy, errors, and checking, by carrying out actual numerical calculations with specific problems.
4. To learn the basic techniques used to approximate a given function by simpler functions, e.g., polynomials.
5. To understand the basic theorems concerned with convergence of sequences generated by iterative procedures.

III. Methods of Evaluation:

1. Final examination
2. Class participation
3. Assignment of specific problems to be turned in
4. Writing of programs to be tested on the computer

IV. Course Content:

A. Solution of Equations (10 lessons)

1. Functional iteration
2. Convergence Theorems including Cauchy Criterion
3. Lipschitz condition
4. Aitken's delta-squared method
5. Newton-Raphson method
6. Method of false position
7. Method of chords
8. Bisection method
9. Bairstow's method for polynomial equations
10. VonMise's method

B. Polynomial approximation (15 lessons)

1. Evaluation of polynomials
2. Taylor polynomial
3. Legendre polynomial
4. Least-squate approximation
5. Definitions of norms
6. Error of approximation
7. Lagrange's interpolation formula
8. Gram-Schmidt process
9. Chebyshev polynomials
10. Trigonometric approximations
11. Newton's interpolation polynomial with divided differences
12. Ordinary differences. Forward and backward difference operator

13. Trapezoidal rule
14. Simpson's rule
15. Gaussian quadrature

C. Solution of ordinary differential equations (5 lessons)

1. Numerical differentiation
2. Runge-Kutta with Runge's coefficients
3. Adams-Moulton predictor-corrector method

D. Matrix algebra and simultaneous equations (10 lessons)

1. Elementary operations
2. Gauss-Jordan elimination method
3. Matrix inversion
4. Gauss-Seidel iterative method
5. Eigenvalues and eigenvectors

E. Monte Carlo (5 lessons)

1. Random number generators
2. Solution of problems
3. Statistical analysis

V. Procedures, Techniques and Methods:

1. Lectures covering theory and areas of application
2. Verification of algorithms using facilities of the Center for Computing Services
3. Assignment of problems

VI. Instructional Materials:

1. Textbook
2. IBM 370 Computing System (batch and inter-active use)

VII. Basic Text:

Elementary Numerical Analysis, 2nd ed., by S.D. Conte, (McGraw-Hill Co.) 1974.

VIII. Bibliography

Forman S. Acton, Numerical Methods That Work, Harper, 1970.

Bibliography Continued

Bruce W. Arden, Kenneth N. Astill, Numerical Algorithms: Origins and Applications, AW, 1970.

Brice Carnahan, H. A. Luther, James O Wilkes, Applied Numerical Methods, Wiley, 1969.

B. Carnahan, J. O. Wilkes, Digital Computing and Numerical Methods (with FORTRAN IV, WATFOR, and WATFIV Programming) Wiley, 1973.

A.M. Cohen, J. F. Cutts, R. Feilder, D. E. Jones, J. Ribbans, E. Stuart, Numerical Analysis, Halsted, 1973.

S. D. Conte, Carl deBoor, Elementary Numerical Analysis, (2nd ed.), McGraw, 1972.

W. S. Dorn, H. J. Greenberg, Mathematics and Computing with FORTRAN Programming, Wiley, 1967.

W. S. Dorn, D. D. McCracken, Numerical Methods with FORTRAN IV Case Studies, Wiley, 1972.

Germund Dahlquist, Ake Bjorck, Ned Anderson, Numerical Methods, PH, 1974.

James W. Daniel, Ramon E. Moore, Computation and Theory in Ordinary Differential Equations, Freeman, 1970.

Philip J. Davis, Philip Rabinowitz, Methods of Numerical Integration, Academic, 1975.

G. E. Forsythe, M. P. Malcolm, C. B. Moler, Computer Methods for Mathematical Computations, PH, 1977.

C. William Gear, Numerical Initial Value Problems in Ordinary Differential Equations, PH, 1971.

Curtis F. Gerald, Applied Numerical Analysis, AW, 1970.

Donald Greenspan, Introduction to Numerical Analysis and Applications, Rand, 1971.

Richard W. Hamming, Introduction to Applied Numerical Analysis, McGraw, 1971.

R. W. Hamming, Numerical Methods for Scientists and Engineers, (2nd ed.), McGraw, 1973.

F. G. Hildebrand, Introduction to Numerical Analysis, (2nd ed.), McGraw, 1974.

Bibliography Cont.

John A. Jacquez, A First Course in Computing and Numerical Methods, AW, 1970.

Hans O. Kunzi, H. G. Tzschach, A. A. Zehnder, Werner C. Rheinbolt, Cornelie J. Rheinbolt, Numerical Methods of Mathematical Optimization with ALGOL and FORTRAN Programs, (revised and augmented), Academic, 1971.

Shan S. Kuo, Computer Applications of Numerical Methods, AW, 1972.

L. Lapidus, Hohn H. Seingeld, Numerical Solution of Ordinary Differential Equations, Academic, 1971.

L. Lapidus, W. E. Schiesser, (eds.), Numerical Methods for Differential Systems: Recent Developments in Algorithms, Software and Application, Academic, 1976.

H. Melvin Lieberstein, A Course in Numerical Analysis, Harper, 1968.

G. I. Marchuk, Methods of Numerical Mathematics, Springer, 1975.

Samuel McNeary, Introduction to Computational Methods for Students of Calculus, PH, 1973.

William E. Milne, Numerical Solution of Differential Equations, Dover, 1970.

Jurg Nievergelt, J. Craig Farrar, Edward M. Reingold, Computer Approaches to Mathematical Problems, PH, 1974.

Ralph H. Pennington, Introductory Computer Methods and Numerical Analysis, (2nd ed.), Macmillan, 1970.

G. M. Phillips, P. J. Taylor, Theory and Applications of Numerical Analysis, Academic, 1974.

J. M. Rushforth, J. L. Morris, Computers and Computing, Wiley, 1973.

Francis Shield, Numerical Analysis, Including 775 Solved Problems, McGraw, 1967.

H. Schmid, Decimal Computation, Wiley, 1974.

L. F. Shampine, R. D. Allen, Jr., Num erical Computing: An Introduction, Saunders, 1973.

Bibliography Cont.

L. F. Shampine, M. K. Gordon, Computer Solution to Ordinary Differential Equations: The Initial Value Problem, Freeman, 1975.

Peter A. Stark, Introduction to Numerical Methods, Macmillan, 1970.

Marvin L Stein, William D. Munro', Introduction to Machine Arithmetic, AW, 1971.

Pat H. Sterbenz, Floating Point Computation, PH, 1974.

W. A. Watson, T. Philipson, P. J. Oates, Numerical Analysis - The Mathematics of Computing, (two volumes), Elsevier, 1970.

James H. Wilkinson, Rounding Errors in Algebraic Processes, PH, 1964.

P. W. Williams, Numerical Computations, Barnes, 1973.

David M. Young, Robert T. Gregory, A Survey of Numerical Mathematics, (two Volumes), AW, 1972,(Vol. 2, 1973).

SUBJECT:	Course Guide
COURSE NUMBER:	CS 308
COURSE NAME:	Numerical Analysis II
DATE OF APPROVAL:	
CREDIT:	3 Semester Hours
PREREQUISITES:	Computer Science I, II, Calculus I, II
TERM:	Fall
MAJOR:	Elective
MINOR:	Elective
GENERAL STUDIES:	No

COURSE DESCRIPTION:

The course includes topics in unconstrained maximization/ minimization of univariate functions using such methods as exhaustive search, interval search, random search, Fibonacci search, etc., and of multivariate functions using techniques such as the method of steepest descent. The linear programming problem is introduced and the simplex method for solving it. Topics are covered in the context of decision making. Verification of algorithms is made using a computer.

I. Aims and Objectives:

The purpose of this course is to give the student an idea of the steps an applied mathematician follows in solving a problem in a realistic situation. The steps are: (1) the recognition and analysis of the non-mathematical origin of the problem, (2) formulation of a mathematical model of the problem situation, (3) solution of the mathematical problem and relevant computations, (4) the interpretation of the results of the solution in terms of the original problem. A major objective is to introduce the student to problem solving in the context of the decision making process, namely the construction of a model of whatever it is a decision must be made about, determination of objectives and criteria with respect to the model, the determination of constraints the possible choices, and finally, optimization within the bounds set by the criteria and constraints. The course can be offered on either an elementary or an advanced level. On an elementary level, the course would be useful to majors in the social sciences.

II. Content and Scope of the Course:

A. Maximization and Minimization of Univariate Functions

1. Exhaustive search
2. Interval search
3. Random search
4. Fibonacci search

B. Maximization and Minimization of Multivariate Functions

1. Exhaustive search
2. Interval search
3. Grid search
4. Steepest descent

C. Matrix Algebra

1. Transpose of a matrix
2. Symmetric matrix
3. Triangular and echelon matrices
4. Determinants
5. Inverse matrix
6. Matrix transformations

D. Systems of Linear Equations

E. Convex Sets

1. Linear inequalities
2. Convex sets
3. Maximum and minimum of a linear function over a convex polygon

F. Linear Programming

1. A general linear programming problem
2. The dual problem
3. Simplex method
4. Degeneracy

III. Procedures, Techniques and Methods:

1. Lectures covering the theory and areas of application
2. Verification of algorithms using facilities of the Center for Computing Services
3. Assignment of problems

IV. Instructional Materials:

1. Textbook
2. IBM 360 Computing System (batch and interactive use)

V. Basic Requirements:

1. Satisfactory run of programs on the computer
2. Class participation
3. Final examination
4. Above requirements made known during first lecture period

VI. Basic Text:

1. Methods of Optimization by Cooper & Stein (Wiley, 1970)
2. Man Made World by contributors to E.C.C.P. (McGraw-Hill, 1968) Placed on reserve

VII. Bibliography

James R. Branch, Donald J. Rose, (ed.), Sparse Matrix Computations, Academic, 1976.

Noel Gastinel, Linear Numerical Analysis, Academic, 1970.

Gerald B. Haggerty, Elementary Numerical Analysis with Programming, Allyn, 1972.

James M. Ortega, Numerical Analysis - A Second Course, Academic, 1972.

H. Schwartz, et al, Numerical Analysis of Symmetric Matrices, PH, 1973.

G. W. Stewart, Introduction to Matrix Computations, Academic, 1973.

SUBJECT:	Course Guide
COURSE NUMBER:	CS 309
COURSE NAME:	Computer applications in the Sciences I
DATE OF APPROVAL:	
CREDIT:	3 Semester Hours
PREREQUISITIES:	Computer Science I, Calculus I
MAJOR:	Elective
MINOR:	Elective
GENERAL STUDIES:	No

COURSE DESCRIPTION:

A survey of the broad spectrum of computer use in science, including the computer as a computational tool, the computer as a laboratory tool, and computer as a partner in design and analysis. Additional topics will include introductory concepts in computer graphics, canned programs vs user developed programs, hints for developing large applications. Students will program a number of problems from the areas of physics, engineering, chemistry, biology, and medicine.

I. Aims and Objectives:

1. To acquaint students with the various modes in which the computer can be used in scientific applications
2. To provide students with an opportunity to develop and debug a number of meaningful programs related to the course

II Content and Scope of the Course:

A. Measurement of Analog Signals by Computer

1. Characteristics of Analog and Digital Signals
2. Data Acquisition devices
3. Sampling of Analog Signals and Reconstruction
4. The Sampling Theorem
5. Interfacing the computer to the outside world

B. Concepts in Real Time Computing

1. Event driven systems
2. Time driven systems
3. Computer driven systems
4. Needs of a real time system programming language

C. Techniques for writing Interactive Programs

1. Writing the front end - or, what do you tell the user?
2. A menu driven front end
3. How do you present the output?

D. Review of Mathematical Concepts

1. Integration by computer
2. Solving differential equations by computer

E. Concepts in Graphics

1. World and User Coordinate Systems
2. Plotting on printers
3. Plotting using graphic devices

F. Programming of specific problems chosen from various scientific fields

III. Procedures, Techniques, and Methods:

1. Lectures
2. Assignment of problems and term project
3. Visits to various computer installations

IV. Instructional Materials:

1. Textbooks
2. Computer manufacturers manuals
3. Instructor's notes

V. Basic Requirements:

1. Quizzes
2. Programming problems
3. Oral and written presentation of term project

VI. Basic Text:

William Ralph Bennett, Jr. Scientific and Engineering Problem-Solving with the Computer (Prentice Hall)

VII. Bibliography

Gerald B. Haggerty, Elementary Numerical Analysis with Programming, Allyn, 1972.

Ramon E. Moore, Mathematical Elements of Scientific Computing, Holt, 1975.

SUBJECT:	Course Guide
COURSE NUMBER:	CS 401
COURSE NAME	Advanced Computer Technology Projects/CO-OP
DATE OF APPROVAL:	
CREDIT:	4 Semester Hours
PREREQUISITES:	Senior Status; B GPA in Computer Classes
TERM:	Spring/Fall
MAJOR:	Elective
MINOR:	No
GENERAL STUDIES:	No

COURSE DESCRIPTION:

The course provides the student an opportunity to sharpen computer skills previously learned and to acquire further technical knowledge in a situation closely akin to that he will encounter in future employment. Under the guidance of a faculty advisor and supervision of the Computer Center staff, he will become a contributing member to significant computer projects under development by the Computer Center and various departments within the College. Students opting for CO-OP will derive a similar learning experience with selected companies in business and industry.

I. Aims and Objectives:

The purpose of this course is to give the student actual experience performing a variety of programming and programming related tasks in the real world situation. The successful completion of this course will greatly enhance the employment possibilities of the student and also provide a measure of self-confidence when starting his first career position.

II. Content and Scope of the Course:

Some of the tasks and activities that the student can anticipate are:

1. How to define a problem
2. How to design alternate solutions to a problem
3. Selecting the optimum solution
4. Translating the solution into computer processable form
5. Testing and implementing the solution
6. Documenting the solution so that others may use it

Other by-products that the student will realize are:

1. Functioning as a member of a project team
2. Knowledge of individual components of a complete automated system
3. The functional and operational components of a total Data Processing Department
4. How computers serve diverse areas of users

III. Procedures, Techniques and Methods:

1. Individual instruction from advisor/project leader
2. Works on individual tasks as member of a larger programming and design team.

IV. Instructional Materials:

1. No specific text
2. Uses applicable language and computer manuals as designated.

V. Basic Requirements:

1. Advanced standing in Computer Science
2. Capability of working alone as well as in a co-operative situation.

VI. Basic Text:

None required

VII. Bibliography

R. L. Albrecht, Organization and Management of Information Processing Systems, Macmillan, 1973.

William F. Boore, Jerry R. Murphy, The Computer Sampler: Management Perspectives on the Computer, McGraw, 1968.

Dick H. Brandon, Arnold D. Palley, A. Michael O'Reilly, Data Processing Management: Methods and Standards, Macmillan, 1975.

P. M. Burnham, E.P.Morris, Effective Computer Management, Wiley, 1975.

Thomas M. Cook, Robert A Russell, Introduction to Management Science, PH, 1977.

Arnold E. Ditri, John C. Shaw, William Atkins, Managing the EDP Function, McGraw, 1971.

M. A. L. Farr, B. Chadwick, K.K. Wong, Security for Computer Systems, Hayden, 1972.

Thomas R. Gildersleeve, Data Processing Project Management, Van 1974.

Charles F. Hemphill, Jr., John M. Hemphill, Security Procedure for Computer Systems, Dow, 1973.

Paul W. Howerton, Management of Information Handling Systems, Hayden, 1974.

J. Kanter, Management Guide to Computer System Selection and Use, PH, 1970.

Harry Katzan, Jr., Computer Data Security, Van, 1973.

John K. Lyon, The Database Administrator, Wiley, 1976.

Warren F. McFarlan, Richard L. Nolan, David P. Norton, Information Systems Administration, Holt, 1973.

Philip W. Metzger, Managing a Programming Project, PH, 1973.

Charles Mosmann, Academic Computers in Service, Jossey-Bass, 1973.

Richard L. Nolan, Managing the Data Resource Function, West, 1974.

George Penney, (ed.), Managing Computers, Hayden, 1975.

Bibliography Continued

John C. Shaw, William Atkins, Managing Computer Systems Project, McGraw, 1970.

Herbert A. Simon, The New Science of Management Decisions, (3rd Ed., revised), PH, 1977.

Dennis Van Tassel, Computer Security Management, PH, 1972.

M. D. Wadsworth, The Human Side of Data Processing Management, PH, 1973.

T. B. Ward, Computer Organization Personnel and Control Longman, 1974.

F. G. Withington, The Organizations of the Data Processing Function, Wiley, 1972.

R. Yearsley, R. Graham, (eds.), Handbook of Computer Management, Halsted, 1973.

SUBJECT:	Course Guide
COURSE NUMBER:	CS 402
COURSE NAME:	Special Topics in Computing
DATE OF APPROVAL:	
CREDIT:	3 Semester Hours
PREREQUISITES:	Instructor's Permission
TERM:	Fall and/or Spring
MAJOR:	Elective
MINOR:	Elective
GENERAL STUDIES:	No

COURSE DESCRIPTION:

The course is a variable content and variable instructor course offered at least once a year covering special topics depending on the expertise of the individual instructor assigned to teach. The intent is to cover topics not included in other standard courses or topics that form part of the expertise of particular instructors.

I. Objectives:

The purpose of this course is make available subject matter and methodology not found in the other courses of the curriculum or not normally found in the standard courses in computer science. It is also an experimentation course where instructional technologies and subject matter can be tested before creating a new course.

II. Content and Scope of the Course:

Special topics in computer science to be arranged.

III. Procedures, Techniques, and Methods:

An effort will be made to tap the special expertise of different instructors each time it is offered.

IV. Instructional Materials:

To be selected.

V. Basic Requirements:

To be specified during the first class meetings.

VI. Basic Text:

To be selected.

SUBJECT:	Course Guide
COURSE NUMBER:	CS 404
COURSE NAME:	Physical Design of Computer Systems
DATE OF APPROVAL:	
CREDIT:	3 Semester Hours
PREREQUISITE:	Data Structures Data Base Design Operating Systems
TERM:	
MAJOR:	Elective
MINOR:	Elective
GENERAL STUDIES:	No

COURSE DESCRIPTION:

A description of new programming system design concepts including, structured programming, stepwise refinement, information hiding, transparency and top-down design. Writing specifications for large scale systems, interpreting these specifications. How to manage a project including scheduling and manpower estimation. Coding and debugging techniques, maintaining and documenting a system. The methods and concepts will be demonstrated by using them in the development of a class project.

I. Course Objective:

This course is intended to show the students the formal and practical approaches to the complex problems involved in large scale systems design and development. The students will explore the different methods as they take a class project from the writing of the systems specifications through design, coding, debugging and documentation. This course will provide the students with a rare opportunity to use and evaluate these techniques in an actual development situation.

II. Content and Scope of the Course:

A. Major-elements of the system development process

1. Problem definition
2. Initial definition of system
3. Justification for project
4. Detailed system requirements
5. System design
6. System development
7. System integration and testing
8. User documentation
9. System maintenance

B. Defining the Problem and the Solution

1. Understanding the User and environment
2. Scoping the problem
3. Presentation of information (written, verbal)

C. Justifying the project

1. Estimation of time and staffing
2. Estimating "worth" of system
3. Economics of software development

D. System Requirements

1. Understanding the user (walk in his shoes)
2. Levels of requirements
3. Organization of information
4. Presentation of information
5. Feedback process

E. System Design techniques

1. Top-down design
2. Stepwise refinement
3. Structured programming
4. Information hiding (software watergate)

F. Development

1. Schedules and staffing
2. Division of labor
3. Design review procedures
4. Chief programmer team concept
5. The tarpit porblem
6. Development environment I

G. Integration and Testing

1. Development environment II
2. Module testing
3. Putting it together
4. Trialing software
5. Feedback and evaluation

H. User Documentation

1. User manual
2. Online documentation
3. Training procedures
4. How not to document

I. Maintaining a System

1. System documentation
2. Program comments
3. Program style

III. Procedures, Techniques and Methods:

1. Lectures supplemented by reading examples
2. Class project

IV. Required Readings:

1. The Mythical Man-Month by F. P. Brook, (Addison-Weseley) 1975.

2. Software Tools by Kerniglan & Plauger (Addison-Wesley) 1976.

V. Bibliography

N. Benwell, S. Chomet, Benchmarking: Computer Evaluation and Measurement, Halsted, 1975.

M.E. Drummond, Evaluation and Measurement Techniques for Digital Computer Systems, PH, 1973.

Walter Freiberger, (ed.), Statistical Computer System Performance Evaluation, Academic, 1972.

Herbert Helerman, Thomas F. Conroy, Computer System Performance, McGraw, 1975.

Edward O. Joslin, Computer Selection, AW, 1968.

Michael F. Morris, Paul F. Roth, Computer Performance Evaluation, Petrocelli, 1977.

Liba Svobodova, Computer Performance Measurement and Evaluation Methods: Analysis and Application, Elsevier, 1975.

Maurice Blackman, The Design of Real-Time Applications, Wiley, 1975.

Dick H. Brandon, Max Gray, Project Control Standards, Petrocelli, 1970.

D. H. Brandon, M. Gray, Management Planning for Data Processing, Petrocelli, 1970.

Raymond J. Colemen, M. H. Riley, (eds.), MIS: Management Dimensions, Holden, 1973.

Hamed K. Elden, Max F. Croft, Information Systems: A Management Science Approach, Petrocelli, 1974.

Bibliography Continued

Tom Glib, Reliable EDP Application Design, Petrocelli, 1974.

W. Hartman, H. Metthes, A. Proeme, Management Information Systems Handbook, (2nd Ed.), McGraw, 1969.

Bartow Hodges, Robert N. Hodgson, Management and the Computer in Information and Control Systems, McGraw, 1969.

Jerome Kanter, Management-Oriented Management Information Systems, (2nd ed.), PH, 1977.

Joseph F. Kelly, Computerized Management Information Systems, Macmillan, 1970.

David Lefkovitz, Data Management for On-Line Systems, Hayden, 1974.

David H. Li, Design and Management of Information Systems, SRA, 1972.

Henry C. Lucas, Toward Creative Systems Design, Columbia,1974.

H. C. Lucas, Computer-based Information Systems in Organization, SRA, 1973

Don Q. Matthews, The Design of Management Information Systems, (2nd ed.), Petrocelli, 1961.

F. Warren McFarlan, Richard L. Nolan, The Information Systems Handbook, Dow, 1975.

Robert G. Murdick, Joel E. Ross, Information Systems for Modern Management, (2nd ed.), PH, 1975.

James J. O'Brien, Management Information Systems: Concepts, Techniques, and Application, Van, 1970.

K. J. Radford, Information Systems in Management, Reston, 197

Joel C. Ross, Management by Information Systems, PH, 1970.

J. Vanduyn, Practical Systems and Procedures Manual, Reston, 1975.

Gerald M. Weinberg, An Introduction to General Systems Thinking, Wiley, 1975.

Jerome D. Wiest, F.K. Levy, A Management Guide to PERT/CPM, (2nd ed.), PH, 1977.

Susan Wooldridge, Systems and Programming Standards, Petrocelli, 1977.

Edward Yourdon, Design of On-Line Computer Systems,PH, 1972.

SUBJECT:	Course Guide
COURSE NUMBER:	CS 405
COURSE NAME:	Microprocessors and Microcomputers
DATE OF APPROVAL:	1978
CREDIT:	3 Semester Hours
PREREQUISITES:	Physics II, Digital Electronics Some knowledge of assembly language recommended.
MAJOR:	Elective
MINOR:	Elective
GENERAL STUDIES:	No

COURSE DESCRIPTION:

This course focuses on microprocessor/microcomputer technology and applications. Topics discussed include: what is a microprocessor/microcomputer; hardware architecture; software facilities; peripheral device interfacing techniques. Various popular microprocessors will be compared; i.e. 8080A, 6800, Z80, LSI111, etc. and future trends will be considered. Students will code some programs for a microcomputer.

I. Aims and Objectives:

1. To acquaint students with the architecture of common microprocessors
2. To acquaint students with the structure of common machine languages
3. To give students an opportunity to program microcomputers and to interface them with external devices.

II. Content and Scope of the Course:

A. Microprocessor Architecture

1. Overview
2. Registers
3. Arithmetic/logic units
4. Instruction handling areas
5. Stacks
6. Some examples

B. Memory

1. RAM
2. ROMs and PROMs
3. Addressing memory

C. Microprocessor Instruction Sets

1. Instruction formats
2. Addressing methods
3. Instruction types
4. Some specific examples

D. Assemblers

1. Some features of assemblers
2. Use of cross assemblers

E. Software I

1. Writing programs for micros
2. Debugging and Testing

F. Input/Output

1. I/O consideration for micros
2. Addressing I/O
3. Timing considerations
4. Parallel Interface Adapters
5. Serial Interface Adapters
6. Some specific examples

G. Interrupt Systems

1. Characteristics of Interrupts
2. Simple interupt systems
3. Software interrupt processing
4. Hardware interrupt handling - priority structure

H. Direct Memory Access

I. Software II

1. Software development considerations

III. Procedures, Techniques and Methods:

1. Lectures and demonstrations
2. Assigned programs and problems

IV. Instructional Materials:

1. Textbooks
2. Microcomputers
3. Interface circuitry

V. Basic Requirements:

1. Quizzes
2. Programming assignments
3. Final exam

VI . Basic Text:

Lance A. Leventhal, Introduction to Microprocessors. (Prentice-Hall) 1973.

VII. Bibliography

Ashok K. Agrawala, Tominson C. Rauscher, Foundations of Microprogramming-Architecture, Software and Applications, Academic, 1976.

D. Aspinall, E. L. Dagless, (ed.), Introduction to Microprocessors, Academic, 1977.

Aspad Barna, Dan I. Porat, Introduction to Microcomputers and Microprocessors, Wiley, 1976.

Guy Boulaye, Microprogramming, Halsted, 1975.

Yaohan Chu, Computer Organization and Micro-programming, PH, 1972.

Electronics Magazine, Microprocessors, McGraw, 1975.

Michael Elphick, Microprocessor Basics, Hayden, 1977.

John L. Hilbrun, Paul M. Julich, Microcomputers/Microprocessors; Hardware, Software, and Applications, PH, 1976.

Samir S. Husson, Microprogramming: Principles and Practices, PH, 1969.

Samir S. Husson, Micropramming/Microprocessors, Hardware, Software and Applications, PH, 1969.

Harry Katzan, Jr., Microprogramming Primer, McGraw, 1977.

Ed Klingman, III, Microprocessor Systems Design, PH, 1977.

William F. Leahy, Microprocessor Architecture and Programming, Wiley, 1977.

Daniel R. McGlynn, Microprocessors: Technology, Architecture, and Applications, Wiley, 1976.

A. Osborne, An Introduction to Microcomputers, Vol. I, II, Osborne, 1976.

Bibliography Continued

J. Peatman, Microcomputer Based Design, McGraw, 1977.

Alan B. Salisbury, Microprogrammable Computer Architecture, Elsevier, 1976.

C. J. Sippl, Microcomputer Handbook, Petrocelli, 1977.

C. J. Sippl, D. A. Kidd, Microcomputer Dictionary and Guide, Matrix, 1975.

Branko Soucek, Microprocessors and Microcomputers, Wiley, 1976.

Edward A. Torrero, (ed.), Microprocessors: New Directions for Designers, Hayden, 1975.

Jan Wilmink, Mariajiovanna Sami, Rodnay Zaks, Microprocessing and Microprogramming, Elsevier, 1977.

SUBJECT:	Course Guide
COURSE NUMBER:	CS 406
COURSE NAME:	Theory of Formal Languages
DATE OF APPROVAL:	
CREDIT:	3 Semester Hours
PREREQUISITES:	Programming Languages
MAJOR:	Elective
MINOR:	Elective
GENERAL STUDIES:	No

COURSE DESCRIPTION:

Formal grammars and automata, production systems and languages, regular, context-free, context-sensitive, and recursive grammars, deterministic and non-deterministic finite automata, context-free languages.

I. Objectives:

This course together with Theory of Computability, provides the basic theoretical concepts in the mathematics component of the computer science major. The course serves as an introduction to the theory of context-free grammars and formal languages, and to syntactic recognition techniques for recognizing languages specified by context-free grammars. The course is intended to provide a theoretical framework for understanding the concepts involved in the study of programming languages.

II. Content and Scope of the Course:

A. Foundations of Language Theory-String Generation

1. Basic Definitions and Notation
2. Σ^* - A Combinatorial View
3. Phrase-structure grammars
4. The Chomsky Hierarchy
5. Context-free Grammars and Trees

B. Finite Automata and Linear Grammars

1. Elementary Finite Automata Theory
2. Ultimately Periodic Sets
3. Transition Systems
4. Kleenes Theorem
5. Right Linear Grammars and Languages
6. Two-way finite automata

C. Basic Properties of Context-Free Languages

1. Introduction
2. Reduced Grammars
3. The Substitution Theorem
4. Regular Sets and Self-Embedding Grammars

D. Normal Forms for Context-Free Grammars

1. Norms, complexity, and reduction
2. Elimination of Null and Chain Rules
3. Chomsky Normal Form
4. Operator Grammars

E. Pushdown Automata

1. Basic Definitions
2. Equivalent Forms of Acceptance
3. Characterization Theorems

F. The Iteration Theorem, Nonclosure and Closure Results

1. The Iteration Theorem
2. Context-free Languages Over a One-Letter Alphabet
3. Regular Sets and Sequential Transducers
4. Applications of the Sequential-Transducer Theorem

III. Procedures, Techniques, and Methods:

1. Lectures covering theory
2. Assignment of problems to be solved
3. Reading assignments and exercises

IV. Instructional Materials:

1. Textbook
2. Materials and notes developed as part of a doctoral project

V. Basic Requirements for Successful Completion:

1. Final examination
2. Class participation

3. Solution of assigned problems
4. Completion of reading assignments
5. Periodic quizzes

VI. Basic Text for the Course:

1. Seymour Ginsburg, The Mathematical Theory of Context-Free Languages. McGraw-Hill, New York, 1966.
2. M.A. Harrison, Introduction to Formal Language Theory. Addison-Wesley, 1978.

VII. Bibliography

Alfred V. Aho, (ed.), Currents in the Theory of Computing, PH, 1973.

Mark A. Aiserman, Leonid A. Gusev, Lev. I Rozonoer, Irina M. Smirnova, Aleksey A. Tal, Logic, Automata and Algorithms, Academic, 1971.

Igor Aleksander, Automata Theory: Fundamentals Applications, Crane, 1975.

Michael A. Arbib, (ed.), The Algebraic Theory of Machines, Languages and Semigroups, Academic, 1968.

Michael A. Arbib, Theories of Abstract Automata, PH, 1969.

J. H. Conway, Regular Algebra and Finite Machines, Wiley, 1971.

J. E. Donahue, Complementary Definitions of Programming Language Semantics, Springer, 1976.

Samuel Eilenberg, Automata, Languages and Machines, (2 vols.) Academic, 1973, (vol. 2, 1975).

J. Engelfriet, Simple Program Schemes and Formal Languages, Springer, 1974.

A. Ershov, V.A. Nepomniaschy, (eds.) International Symposium on Theoretical Programming, Springer, 1974.

S. Ginsburg, Algebraic and Automata-Theoretic Properties of Formal Languages, North Holland, 1974.

Abraham Ginzburg, Algebraic Theory of Automata, Academic, 1968.

Bibliography Continued

Michael A. Harrison, Lectures on Linear Sequential Machines, Academic, 1970.

Gabor T. Herman, Gizegerz Rosenberg, Developmental Systems and Languages, Elevier, 1975.

John E. Hopcroft, Jeffrey D. Ullman, Formal Languages and Their Relation to Automata, AW, 1969.

Richard Y. Kain, Automata Theory: Machines and Languages, McGraw, 1972.

N.E. Kobrinskii, B.A. Trakhtenbrot, Introduction to the Theory of Finite Automata, North Holland, 1965.

Zvi Kohavi, Switching and Finite Automata Theory, McGraw, 1970.

J. Loeckx, (ed.), Automata, Languages and Programming, Springer, 1974.

A.A. Lorenz, Stochastic Automata: Constructive Theory, Wiley, 1974.

Marvin Minsky, Computation: Finite and Infinite Machines, PH, 1967.

Randall Rustin, Formal Semantics of Programming Languages PH, 1972.

Arto Salomaa, Theory of Automata, Pergamon, 1969.

SUBJECT:	Course Guide
COURSE NUMBER:	CS 407
COURSE NAME:	Theory of Computability
DATE OF APPROVAL:	
CREDIT:	3 Semester Hours
PREREQUISITES:	Data Structures, Abstract Algebra
MAJOR:	Elective
MINOR:	Elective
GENERAL STUDIES:	No

COURSE DESCRIPTION:

The course covers properties of algorithmic computation, Turing machines, Turing computability, primitive recursive functions, recursive functions, and computability and decidability.

I. Course Content and Scope:

A. Properties of Algorithmic Computation

1. Functions and Algorithms
2. Five Basic Properties
3. Universal, Decision Functions
4. Index Computation
5. Inherent Capabilities and Limitations
6. Standard Families of Algorithms

B. Turing Machines

1. Basic Definitions
2. State Diagrams
3. Copy, Matching, Substitution, and Composite Machines
4. Variations of the Turing Machine Model
5. A Universal Turing Machine

C. Turing Computability

1. Number-theoretic Computations and Indexing
2. The Family of Turing Machines
3. The Enumeration Theorem
4. The S-M-N Theorem

5. Basic Capabilities and Limitations
6. The Domain and Halting Function
7. Index Computation
8. Totality and Equivalence Functions
9. The Recursion Theorem

D. Primitive Recursive Functions

1. Primitive Recursion
2. Bounded Sums and Products
3. Bounded Minimalization
4. The Division Function
5. Cantor and Gödel Numberings
6. Simultaneous Recursion
7. Course-of-Values Recursion
8. Primitive Recursive Predicates

E. Recursive Functions

1. Ackermann's Function
2. Mu-Recursive Functions
3. General Recursive Functions

F. Computability and Decidability

1. Computable Functions
2. Recursive Functions and the Church-Turing Thesis
3. Role of the Basic Properties
4. Relative Computability
5. Computable Sets
6. Criteria for Recursive Enumerability
7. Decision Problems
8. Many-One Reducibility
9. The Correspondence Problem
10. Applications to Context-Free Grammars

II. Procedures, Techniques, and Methods:

1. Lectures covering theory and areas of application
2. Assignment of problems
3. Assigned reading

III. Instructional Materials:

1. Textbook
2. Materials and notes developed as part of a doctoral project

IV. Methods of Evaluation:

1. Final examination
2. Class participation
3. Assignment of specific problems to be turned in
4. Periodic quizzes

V. Basic Text:

F. Hennie, Introduction to Computability, Addison-Wesley, 1977.

VI. Bibliography

Alfred V. Aho, (ed.), Currents in the Theory of Computing, PH, 1973.

Mark A. Aiserman, Leonid A. Gusev, Lev. I. Rozonoer, Irina M. Smirnova, Aleksy A. Tal, Logic, Automata and Algorithms, Academic, 1971.

D. W. Barron, Recursive Techniques in Programming, (2nd revised ed.), Elsevier, 1975.

J. Becvar, (ed.), Mathematical Foundations of Computer Science, Springer, 1975.

R. Bird, Programs and Machines: An Introduction to the Theory of Computations, Wiley, 1976.

A. Borodin, I. Munro, The Computational Complexity of Algebraic and Numeric Problems, Elsevier, 1975.

W. S. Brainerd, L.H. Landweber, Theory of Computation, Wiley, 1974.

Madan M. Gupta, George N. Saridis, Fuzzy Automata and Decision Processes, Elsevier, 1977.

Fred Hennie, Introduction to Computability, AW, 1977.

Neil D. Jones, Computability Theory, Academic, 1973.

Charles A. Kapps, Samuel Bergman, Introduction to the Theory of Computing, Merrill, 1975.

Zvi Kohavi, Azaria Paz, Theory of Machines and Computation, Academic, 1971.

Zohar Manna, Introduction to Mathematical Theory of Computation, McGraw, 1974.

Hartley Rogers, Jr., Theory of Recursive Functions and Effective Computing, McGraw, 1967.

John E. Savage, The Complexity of Computing, Wiley, 1967.

Raymond T. Yeh, (ed.), Applied Computation Theory, Annalysis Design and Modeling, PH, 1976.

Appendix 2

JERSEY CITY STATE COLLEGE

DEPARTMENT OF MATHEMATICS

Undergraduate Program in Computer Science Mathematics

September 1975

Undergraduate Program in Computer Science Mathematics

Table of Contents

I. Objectives of the Program

The undergraduate program in Computer Science Mathematics is oriented to (1) students majoring in Computer Science Mathematics, (2) students majoring in the physical and biological sciences (Physics, Chemistry, Biology, etc.) and the social sciences (Economics, Psychology, Sociology, etc.) who are interested in a working knowledge of computer problem solving, and (3) non-technical students interested in an introduction to computer applications and its impact on society.

The major in Computer Science Mathematics is designed (1) to prepare students to enter professional positions in areas of design, implementation, and operation of computer systems in scientific or business environments, (2) to prepare students for graduate study in mathematical sciences, e.g., computer science, mathematics, or statistics, and (3) to prepare students to teach computer science and computer oriented mathematics on the secondary school level. For non-majors the program contains courses designed to provide a student with the knowledge sufficient to use the computer as a tool for problem solving in his or her area of specialization. For the general student the program includes introductory and survey type courses covering applications of the computer and mathematics in our technical society.

II. Program Resources at Jersey City State College

A. The Department of Mathematics

The Mathematics Department of Jersey City State College has offered a major in mathematics since 1958. Most of the computer science and computational math courses now offered by the department were introduced in 1968, the year the college acquired its first general purpose computer. Thus, the current program in Computer Science Mathematics represents a consolidation, extension and formalization of the program developed over the period since. The Physics Department of Jersey City State College has also developed courses in computer science and

computer applications in scientific areas and these courses represent an essential and important compliment to the program in Computer Science Mathematics.

The model for the undergraduate program was the 1964 and 1971 recommendations on computational mathematics of the Panel on Computing of the committee on the Undergraduate Programs in Mathematics (C.U.P.M.) of the American Mathematical Association. The computational mathematics introduced into the curriculum in 1968 was patterned after the 1964 recommendations of C.U.P.M. and the current program in Computer Science Mathematics is patterned after the 1971 recommendations. The content of certain courses was selected according to the 1968 Recommendations for Academic Programs in Computer Science as given in the Report of the Association for Computing Machinery Curriculum Committee on Computer Science.

B. Computer Facilities

The Center for Computing Services of Jersey City State College provides batch services supplied by an IBM 1130 Computing System linked to an IBM 370-158 maintained by Educational Information Services, Inc. The center also provides time sharing services supplied by 8 terminals linked to an IBM 360 Model 90. Jersey City State College is an active participant in Educational Information Services, Inc., the New Jersey organization of institutions of higher education maintaining a computer network in the state and providing much of the computing capabilities for the colleges and universities of the state. The computer services carried out by the network provide extensive hardware and software facilities available to the student in the program in Computer Science Mathematics at Jersey City State College. Familiarity with the capabilities of these sophisticated systems is one of the objectives of the undergraduate program. The Jersey City State College Center for Computing Services also provides valuable experience to eligible students through employment at the center.

C. Faculty

A list of members of the faculty of Jersey City State College teaching computer oriented courses in the program and their subject matter and research specialisation is given below. Courses in the basic mathematics component are taught by members of the Mathematics Department at large . (See the College catalogue.)

Harold F. Carney: Associate Professor, Mathematics Department; B.S. (Saint Peter's College), M.A. (Seton Hall University), Ph.D. (New York University); combinatorial analysis and discrete probability theory, mathematical programming, business data processing.

Philip W. Caverly; Assistant Professor, Mathematics Department; B.S. (Stevens Institute of Technology), M.S. (Seton Hall University); mathematical analysis, differential equations, analog computing, modeling and system simulation.

George Daneluk; Associate Professor, Mathematics Department; B.A. (University of Wyoming), M.A. (Rutgers University); numerical analysis, graph theory, Boolean Algebra, machine organization, statistical analysis, computer assisted instruction, hospital computer based food service systems, methods of optimization, assembly language, computer generated music.

Philip Goldstein; Associate Professor, Physics Department; B.S. (City College of New York), M.S., Ph.D. (Carnegie Mellon University); computer applications in physics, modeling and simulation of physical systems, computer assisted instruction electronics and switching theory.

Joseph U. Grauman; Assistant Professor, Physics Department; B.S., M.S., Ph.D. (Stevens Institute of Technology); Digital: High-energy relativistic kinematics, statistical analysis of experiments, numerical analysis routines, three-dimensional spatial reconstructions, Monte Carlo simulations of experiments, computer graphics and art; Analog: simulations of various linear and nonlinear, physics problems,

simulations in bioengineering, computer art.

Teresa C. Michnowicz; Assistant Professor, Mathematics Department; A.B., M.S. (Rutgers University); computational methods in linear and abstract algebra, numerical analysis, computer graphics, data processing and Cobol programming, APL.

Richard F. Riggs; Associate Professor, Mathematics Department; A.B. (Knox College), M.A. (Rutgers University), Ed.D. (Rutgers University); combinatorial analysis and probability theory, mathematical programming, computational methods in linear algebra, computer assisted instruction, mathematical modeling, applications of statistics to the social sciences, mathematical statistics.

Howard Schulte; Assistant Professor, Mathematics Department, B.S. (Newark State College) Ed.M. (Rutgers University); computational statistics, statistical inference, mathematical programming.

Philip J. Wallack; Director of Computer Services; B.S., M.S. (City College of New York); operating systems and systems programming, educational systems, compiler techniques, computer applications in physics.

D. Program Administration

The program in Computer Science Mathematics is offered by the Mathematics Department within the School of Arts and Sciences and is coordinated by the Steering Committee for Computational Mathematics, a standing committee within the Mathematics Department. The current chairman of the Mathematics Department is Dr. John Reckzeh; the present chairman of the Steering Committee for Computational Mathematics is George Daneluk. The function of the Steering Committee for Computational Mathematics is (1) to develop and implement a program leading to a B.A. Degree in Computer Science Mathematics, (2) to coordinate the program in Computer Science Mathematics with other programs offered in the Mathematics Department, (3) to coordinate the service aspects of the program with programs

offered by other departments of the college, (4) to determine policy and set guidelines for the program within the rules set by the College Senate, the College Administration, and the Department of Mathematics. The present members of the committee are Harold Carney, Philip Caverly, George Daneluk, Priscilla Haindl, Theresa Michnowicz, and Richard Riggs, P. Goldstein, P. Wallach, & J. Grauman.

III. The Bachelor of Arts Program in Computer Science Mathematics

A. The Curriculum

The curriculum leading to the Bachelor of Arts Degree in Computer Science Mathematics is made up of course work divided into two main components, a basic component and an elective component. The basic component consists of a total of 36 semester hours of courses subdivided into three main subject areas: (1) basic mathematics courses (15 s.h.), (2) computational mathematics courses (9 s.h.), and (3) computer science courses (12 s.h.). The student must complete the specified number of semester hour credits in each subject area. The courses in the elective component are also grouped into these same three categories. An outline of the curriculum showing the course groupings is given on page 8. Outlines of each course are given in Appendix I.

B. Course Requirements

A student majoring in Computer Science Mathematics must complete a minimum of 36 semester hours of course work in the curriculum, of which 24 semester hour credits are made up of required courses. The 36 semester hour credits recommended to the student consist of the courses comprising the basic component shown in the curriculum outline. Substitutions from the elective component or of other courses offered by the Mathematics Department or other departments of the college are permitted but only after consultation with the student's advisor. Substitutions are permitted only from the corresponding categories.

A major in Computer Science Mathematics must satisfy the general course requirements as set forth by the School of Arts and Sciences. These include a total of 128 semester hour credits, 36 of which make up the minimum requirement for a major, 36 of which make up the General Studies requirement, and the remaining consisting of free electives. The student should choose his General Studies courses and his free electives in a manner which will supplement the work in the major. Since computer professionals work in environments involving people from many different fields, a student should select courses in the humanities, arts, etc., which will give him a breadth of experience as well as depth in his or her own specialty. A suggested four year curriculum is given in Appendix II.

COMPUTER SCIENCE MATHEMATICS CURRICULUM

BASIC COMPONENT

a)	Mathematics Courses	
1)	Calculus I, II	6 s.h.
2)	Linear Algebra	3 s.h.
3)	Abstract Algebra	3 s.h.
4)	Mathematical Statistics	3 s.h.
b)	Computational Mathematics	
1)	Numerical Analysis	3 s.h.
2)	Differential Equations	3 s.h.
3)	Applied Mathematics	3 s.h.
c)	Computer Science	
1)	Introduction to Computers	3 s.h.
2)	Intermediate Programming	3 s.h.
3)	Machine Organization	3 s.h.
4)	Data Structures	3 s.h.

ELECTIVE COMPONENT

a)	Mathematics Courses	
1)	Mathematical Statistics II	3 s.h.
2)	Advanced Calculus I, II	3 s.h. each
3)	Modern Geometry I, II	3 s.h. each
4)	Topology	3 s.h.
5)	History of Mathematics	3 s.h.
b)	Computational Mathematics	
1)	Modeling and System Simulation	3 s.h.
2)	Man Made World (Math and Physics Departments)	3 s.h.
3)	Descriptive Statistics	3 s.h.
4)	Computer Methods in Physical Sciences (Physics Department)	3 s.h.
5)	Combinatorial Computing	3 s.h.
c)	Computer Science	
1)	Systems Programming	3 s.h.
2)	Data Processing	3 s.h.
3)	Cobol Programming	3 s.h.
4)	Compiler Techniques (Center for Computer Services)	3 s.h.
5)	Theory of Computability	3 s.h.
6)	Digital Electronics (Physics Department)	3 s.h.
7)	Computer Operations Practicum (Center for Computer Services)	3 s.h.
8)	Advanced Programming	3 s.h.

The following is a list of the required courses in the basic component of the major. The courses are designed to provide basic concepts and methodology in mathematics, computational mathematics, and computer science.

105	Calculus I	4 s.h.
106	Calculus II	4 s.h.
107	Introduction to Computers	3 s.h.
108	Intermediate Programming	3 s.h.
207	Linear Algebra	3 s.h.
210	Applied Mathematics	3 s.h.
302	Numerical Analysis	3 s.h.
409	Data Structures	3 s.h.

C. Area of Study

In addition to the above requirements, every major in Computer Science Mathematics must choose an area of study outside of the Mathematics Department but related to his major. A student must complete a minimum 3 course sequence of study in the selected area, one of which must be an upper division course. The area is chosen by the student in consultation with his or her advisor. Two strongly recommended areas are physics and economics.

D. Elective Courses

The requirements above define minimum requirements for the B.A. degree in Computer Science Mathematics. In order to strengthen his major, the student should take additional electives from the elective component of the curriculum. A student is allowed to take a maximum of 54 s.h. credits of subject matter in his major area.

APPENDIX I

The following are descriptions of all the computer orientated courses and most of the basic mathematics courses making up the program of studies for the major in Computer Science Mathematics. Descriptions of other courses not included in the appendix can be found in the Jersey City State College catalogue. The listing is in the order of the catalogue number of the courses.

Math 107 INTRODUCTION TO COMPUTERS (Spring and Fall) 3 s.h.

Prerequisite: College admission

Objectives: This course is required for students majoring in Computer Science Mathematics and provides the background necessary for the more advanced courses in the program. The course is also suitable for the general student who is interested in learning the fundamental subject matter and methodology of computing.

Description: Concepts of an algorithmic process, flowcharting, and programming in the context of problem solving are covered. The topics in programming include data representation, the principle statements of the PL/1 or Fortran language and verification of programs on the computer. A variety of different methods are discussed for solving both numerical and non-numerical problems. Survey of historical developments, future directions, and discussion of current areas of application of the computer is included. The course involves both interactive and batch use of the computer.

Text: Computers: Introduction to Computers and Applied Computing Concepts: Charles H. Davidson and Eldo C. Koenig; John Wiley & Sons, 1967.

Math 108 INTERMEDIATE PROGRAMMING (Fall) 3 s.h.

Prerequisite: Introduction to Computers

Objectives: This course, together with Introduction to Computers, provides the basic concepts needed for further work in the computer science area of the major. The emphasis is on the basic structure of the machine from a functional and programming point of view and is intended to provide the student with an understanding of the internal behavior of computers. A major objective is to provide the student with some facility in assembly language programming. The course is required for students majoring in Computer Science Mathematics.

Description: Computer structure, machine language, instruction execution, addressing techniques and digital representation of data. Symbolic coding and assembly systems, macro definition and generation, and program segmentation and linkage. Several computer projects to illustrate machine structure and programming techniques.

Text: Introduction to Machine and Assembly Language: System 360/370: Frank D. Vicker; Holt, Rinehart and Winston, Inc., 1971.

Math 109 COBOL PROGRAMMING WITH BUSINESS APPLICATIONS 3 s.h.

Prerequisite: Introduction to Computers

Objectives: This course will introduce the student to the basics of COBOL. Emphasis is placed on the logic and on the application of the language in solving business data processing problems. The course is recommended for majors in Computer Science Mathematics as well as for students in social sciences and humanities who are interested in applications of computers in data processing.

Description: The features of COBOL will be presented by writing and executing a modular set of problems. Topics include creating and updating files, internal data representation, hierarchies of storage, input-output devices, and report generation.

Text: Elementary COBOL: Davis and Litecky; McGraw-Hill, 1971

Math 115 MAN MADE WORLD (Fall, Spring) 3 s.h.

Prerequisite: College Admission

Objectives: This course is a general studies course aimed at the general student interested in the role of mathematics and computers in society. It deals with the relationship between man and machine, between society and technology. Problems are solved in the context of the decision making process, namely the construction of a model of whatever a decision must be made about, the determination of the criteria with respect to the variables of the model, the determination of the constraints on the possible choices, and finally, optimization within the bounds set by the criteria and constraints. An important objective is to introduce the student to the basic concepts involved in programming and computing using a general purpose computer. Since this course is being widely implemented on the secondary school level it would be of interest to teachers.

Description: This course deals with the relationship between society and technology with emphasis on the roles of mathematics and computing. Topics include model building, optimization, systems concepts, etc., in the context of decision making. Basic concepts in logic, set theory, and graph theory are introduced. Concepts in computing and programming of a general purpose computer in the context of problem solving are covered.

Text: The Man Made World: The Engineering Curriculum Concepts Project: McGraw-Hill Book Company, 1971

Math 207 LINEAR ALGEBRA (Spring and Fall) 3 s.h.

Prerequisites: Calculus I, II

Objectives: This course is required for students majoring in computational mathematics. The mathematical progress in pure mathematics is a necessary aspect for students in this program. Linear Algebra provides the background for both computational mathematics and computer science courses. The course involves a balance between concreteness and generality. It is most useful at this level in order to involve the student in a different and more abstract style of reasoning and proof.

Description: An introduction to the algebra and geometry of 3-dimensional Euclidean space and its extension to n-space. A variety of topics useful for other courses in the program are discussed: vectors in R^n and the geometry in R^n, linear mappings from R^n into R^m and their matrix representation, matrix algebra, determinants and canonical forms. These topics are coupled with the solution of systems of linear equations.

Text: Elementary Linear Algebra: Nering; W.B. Saunders Co., 1974.

Math 210 APPLICATIONS OF MATHEMATICS (Fall) 3 s.h.

Prerequisites: Introduction to Computers, Calculus I, II

Objectives: The purpose of this course is to give the student an idea of the steps an applied mathematician follows in solving a problem in a realistic situation. The steps are: (1) the recognition and analysis of the non-mathematical origin of the problem, (2) formulation of a mathematical model of the problem situation, (3) solution of the mathematical problem and relevant computations, (4) and the interpretation of the results of the solution in terms of the original problem. A major objective is to introduce the student to problem solving in the context of the decision making process, namely, construction of a model of whatever it is a decision must be made about, determination of objectives and criteria with respect to the model, the determination of constraints on the possible choices, and finally, optimization within the bounds set by the criteria and constraints. The course can be offered on either an elementary or an advanced level. On an elementary level the course would be useful to majors in the social sciences.

Description: The course includes topics in unconstrained maximization of univariate functions using such methods as exhaustive search, interval search, random search, Fibonacci search, etc., and of multivariate functions using techniques such as the method of steepest ascent. The linear programming problem is introduced and the simplex method for solving it. Topics are covered in the context of decision making. Verification of algorithms is made using a computer.

Text: Introduction to Methods Optimization: Leon Cooper and David Steinberg; W.B. Saunders Company, 1970.

Math 301 DIFFERENTIAL EQUATIONS (Fall) 3 s.h.

Prerequisites: Calculus I, II, III, IV

Objectives: This course is an elective suitable for junior and senior mathematics majors. The course is designed to introduce the student to the broad field of differential equations, how they arise in practice, the classification and methods of solution and simulation of these problems through the solutions of the equations.

Description: The following topics are covered: how differential equations arise in practice, the study of exact equations, integrating factor and the first order linear equation, the development of linear systems of differential equations and their solution by matrix methods, numerical solution of the first order initial value problem using the Runge-Kutta methods, plane autonomous systems and stability.

Text: Modern Elementary Differential Equations; Bellman and Cooke.

Math 305 MATHEMATICAL STATISTICS I, (Spring and Fall) 3 s.h.

Prerequisites: Calculus I, II

Objectives: This course is required for all students majoring in mathematics and it provides the background for further work in the theory of probability and statistics as well as an understanding of elementary probability theory and applications of calculus and linear algebra which is not likely to be obtained from any other course. This course is also suitable for majors in Computer Science Mathematics as well as majors in non-mathematical areas who have a competency in mathematics through the calculus. The student who successfully completes this course will be able to apply counting formulas to stochastic processes, sequences, unordered selections, and partitions. He will be able to use Baye's Theorem, apply conditional probability and independence theorems, derive the mean and variance of the binomial and normal distributions, apply these distributions to problems, approximate the binomial with the normal, and apply various computational methods to achieve approximate answers.

Description: Counting formulas and tree diagrams are covered in detail. Use is made of electronic calculators and computers in achieving numerical results. Applications are made to lotteries, television game shows and games of chance. Axiomatic probability theory is developed on an elementary level with some theorems being proved. Estimation theory is introduced with some emphasis on bias for those estimators commonly used for sampling distributions. The binomial and normal distributions are studied as well as the normal approximation to the binomial. The mean and variance of these distributions are derived. Considerable work is done with various random variables - both discrete and continuous - and their expectations. Small electronic calculators are often used in class and the computer is used for special projects. Students are encouraged to attack interesting problems not solved in class.

MATHEMATICAL STATISTICS I (cont'd.)

References:

Freund, John E. Mathematical Statistics, 2nd ed. Prentice-Hall, 1971.

Hoel, Paul G. Introduction to Mathematical Statistics, 4th ed. John Wiley & Sons, 1971.

Hogg, Robert V and Craig, A.T. Introduction to Mathematical Statistics, 3rd ed., Macmillan, 1970.

Lipschutz, Seymour. Probability, Schaum's Outline Series, McGraw-Hill Book Company, 1968.

Meyer, Paul L. Introductory Probability and Statistical Applications. Addison-Wesley Publishing Co., 1965.

Mosteller, Frederick, Rourke, Robert E.K., and Thomas, George B. Br. Probability with Statistical Applications. Addison-Wesley Pub. Co., 1961.

Math 306 MATHEMATICAL STATISTICS II (Spring) 3 s.h.

Prerequisites: Mathematical Statistics I

Objectives: This course is a natural follow-up to the Mathematical Statistics I course in that the common statistical tests are developed using the probability theory, random variable properties, special distributions, and expectations that were developed in that course. This course is suitable for all students who successfully completed Math. Statistics I including Computer Science majors and majors in non-mathematic areas who have the prerequisite mathematical skills. The student who successfully completes the course will be able to use the most common parametric and non-parametric tests. The theory which supports the t-test, the F ratio, the chi-square statistic, sign tests, and rank tests is fully developed. The student will be able to work with real data, consider the objectives of the scientific investigation that gave rise to these data, study statistical methods for answering relevant questions, and consider the interpretation of the results of statistical analysis. The student will gain insight into how experimental data is generated and how the scientist copes with uncertainty and variability. The knowledge gained in this course should be extremely helpful whether the student continues with applied methods courses or theoretical statistics courses.

Description: Topics covered in this course will include but not be limited to estimation of parameters, consistency, unbiasedness, maximum likelihood, confidence intervals, testing hypotheses, power functions, Typr I and II errors, Neyman-Pearson lemma, likelihood ratio tests, tests for means and variances, regression and correlation and chi-square tests. Some topics from nonparametric statistics and design of experiments are included. Meaningful cross-references between theoretical models and real-world problems are made throughout the course. Students work with both real and contrived data. They utilize

MATHEMATICAL STATISTICS II (cont'd.)

electronic calculators and computers extensively - according to their background and experience. Applications of the tests to meaningful problems in the natural and social sciences are made. Special projects will be assigned for which the student must analyze data obtained from the journals. Students are encouraged to attempt problems which may not be solved in class.

References:

Cutler, S.J. "A review of the Statistical Evidence on the Association Between Smoking and Lung Cancer." Journal of the American Statistical Association, June 1955, pp. 267-83.

Dixon, Wilfred J. and Massey, F.J., Jr. Introduction to Statistical Analysis, 3rd ed. McGraw-Hill, 1969.

Hogg, Robert V. and Craig, A.T. Introduction to Mathematical Statistics, 3rd ed. Macmillan, 1970.

Mood, Alexander M. and Graybill, F.A. Introduction to the Theory of Statistics, 2nd ed. McGraw-Hill, 1963.

Spiegel, Murray R. Statistics, Schaum's Outline Series, McGraw-Hill Book Co., 1961.

Tufte, Edward R. The Quantitative Analysis of Social Problems. Addison-Wesley, 1970.

Winer, B.J. Statistical Principles in Experimental Design. McGraw-Hill, 1962.

Math 402 TOPOLOGY (Spring) 3 s.h.

Prerequisites: Calculus I-IV, Linear Algebra

Objectives: This course is designed to be an upper level elective open to junior and senior mathematics majors intending future graduate study in mathematics. The course is geared to introduce the student to the level of thinking needed in graduate work and includes topics which will be needed in first year graduate courses.

Description: Topics included in the course are the following: topological spaces, continuous mappings and homeomorphisms, a view of general topology including the study of the topological invariants, separation compactness and connectedness, homotopy and the Fundamental group.

Text: Introduction to Algebraic Topology: A. Wallace

Math 405 ADVANCED CALCULUS I (Fall) 3 s.h.

Prerequisites: Calculus I, II, III, IV

Objectives: This course is an elective suitable for junior and senior mathematics majors intending future graduate study. The course is geared to introduce the student to the level of thinking needed in graduate work and to topics which allow participation in upper level electives and independent study programs.

Description: The study of mappings from n-dimensional E^n euclidean space to m-dimensional euclidean space E^m is covered. The continuity, differentiation and differentials are studied from a matrix point of view. The Jacobian as the determinant of the differential is used to study local and global properties of the mappings. The proof of the implicit function theorem is given for mappings from E^n to E^m.

Text: Calculus of Several Variables: C. Goffman

Math 406 ADVANCED CALCULUS II (Spring) 3 s.h.

Prerequisite: Advanced Calculus II

Objectives: This course is a continuation of Advanced Calculus I. The topics covered in this course are selected to be useful to the student in the upper level undergraduate electives as well as in future graduate study.

Description: The study of differential forms and their exterior algebra and development of the theory of manifolds are covered. Also developed are integration of differential forms on manifolds and applications of integration on manifolds to the theorems of Green, Gauss and Stokes.

Text: Calculus of Several Variables: C. Goffman

Math 407 MACHINE ORGANIZATION (Fall) 3 s.h.

Prerequisites: Introduction to Computers, Intermediate Programming

Objectives: (1) To introduce the student to the basic elements and logic design techniques of systems making up the components of a computer. (2) To illustrate the concept of an interpretation of an abstract mathematical system and the concept of a realization of a particular interpretation. (3) To show the connection between the components of the computer hardware system and the features of a machine language programming system. (4) To provide laboratory experience wiring logic circuits using logic circuit boards.

Description: Topics include an axiomatic and formal development of basic theorems of Boolean Algebra with interpretations in logic, set theory, and switching circuits. Realizations of Boolean functions, binary arithmetic and coding, computer components such as the adder, shift registers, comparators, selection trees, memory elements, counters, etc., and total system organization and machine programming.

Text: Boolean Algebra and Switching Circuits: Elliot Mendelson; (The Schaum Outline Series); McGraw-Hill, 1970.

Math 408 ADVANCED PROGRAMMING (Spring) 3 s.h.

Prerequisites: Introduction to Computers, Intermediate Programming

Objectives: The purpose of this course is to present a systematic approach to programming languages. The student is expected to become familiar with several different level languages. Selected topics from the theory of formal languages and syntactic analysis are also introduced.

Description: Formal definition of programming languages. Introduction to list processing, string manipulation, data description, and simulation languages. Definition of formal grammars. Semantics of grammatical constructs.

Text: Advanced Programming: Harry Katzan; Van Nostrand, Rineholt & Co., 1970.

Math 409 DATA STRUCTURES (Fall) 3 s.h.

Prerequisites: Introduction to Computers

Objectives: (1) The main thrust of this course is to introduce the student to the elements of graph theory from both a mathematical and an applications point of view. (2) To introduce methodology for representing relations with applications to relations which hold among the elements of data involved in problems. (3) To introduce concepts involved in representing and characterizing the structure of non-numeric data such as bit strings, character strings, etc. (4) To provide programming experience in verifying algorithms for solving problems which depend on operations on data structures. (5) To develop in the student an intuition for the structure of relations in discrete finite sets of data elements. (6) To introduce the student to the PL/1 programming system and its capabilities for handling the various data types.

Description: This course consists of a survey of topics in graph theory, including directed and planar graphs, trees, properties of graphs such as connectiveness, completeness, isomorphisms, etc., adjacency and path matrices of graphs for computer storage. Algorithms for traversing trees, the shortest path problem, generating linked structures, sorting, etc. Topics in the algebra of strings and string manipulation. Applications of the above concepts in handling linear lists, arrays, storage data structures, etc. in the PL/1 programming system.

Text: Data Structures: Theory and Practice: A.T. Berztiss; Academic Press, 1971.

Math 410 THEORY OF COMPUTABILITY (Summer) 3 s.h.

Prerequisites: Abstract Algebra, Machine Organization

Objectives: This course is aimed at advanced students planning to do graduate work in computer science. It introduces the student to abstract machines as a model in the study of computability and computational complexity. Emphasis is placed on the multitape Turing machine as a suitable model.

Description: Introduction to Turing machines, Wang machines, Sheppard Sturgis and other machines. Godel numbering and unsolved results. The halting problem. Posts' correspondence problem and relative uncomputability. Machines with restricted memory, limited memory, and limited computing time. Recursive function theory and complexity classification. Models of computation including relationships to algorithms and programming.

Text: Computability and Unsolvability: Martin Davis; McGraw-Hill Book Co., 1958.

Math 350 NUMERICAL ANALYSIS (Spring) 3 s.h.

Prerequisite: Calculus I, II; Introduction to Computers

Objectives: (1) To introduce students to numerical analysis as a mathematical discipline. (2) To provide experience in developing computer orientated algorithms for solving numeric problems. (3) To provide experience in writing programs for verifying algorithms for solving numeric problems using the computer. (4) To provide experience in evaluating the advantages and disadvantages of different algorithms for solving a class of problems relative to certain criteria. (5) To develop in the student an intuition with respect to approximation by iterative convergence techniques.

Description: The course consists of a survey of topics in numerical analysis including error analysis, iterative methods for solving equations, methods for maximizing functions, polynomial approximation, iterative and direct methods for solving systems of linear equations, matrix methods, computation of eigen values of matrices, numerical differentiation and integration.

Text: Elementary Numerical Analysis: Samuel D. Conte; McGraw-Hill, 1971.

Math COMPUTER OPERATIONS PRACTICUM (Fall and Spring) 3 s.h.

Prerequisites: Junior or senior level standing in a computer studies program.

Objectives: The purpose of this course is to provide a student with experience in the practical operations involved in maintaining a computer center. It is intended to bridge the gap between the theoretical approach to topics in computer science that a student is exposed to in academic courses and the tasks and problems associated with the performance of the day to day functions of a computer center or data processing center providing computer services to users. It is intended to give the student an idea of how to perform as a professional in an everyday working environment.

Description: A student is assigned to the Center for Computer Services at the college and given a regular work schedule of duties to be performed under the supervision of the full-time staff. In addition, the student attends a one hour per week formal class meeting conducted by the full-time staff for an in-depth discussion of the various functions performed by the staff. Reading assignments from various journals are given.

Math COMPILER TECHNIQUES (Fall) 3 s.h.

Prerequisites: Introduction to Computers or knowledge of Fortran

Objectives: The purpose of this course is to provide the student with an in-depth understanding of the concepts involved in the process of compilation of programs.

Description: A compiler for the Fortran language will be studied as an application of advanced programming concepts. The formal definition of a language, parsing, extraction of language elements, use of the symbol table and compilation of control statements will be discussed.

Text:

1) The Anatomy of a Compiler: John A.N. Lee; Van Nostrand,RRineholt Co., 1967.

2) Compiling Techniques: F.R.A. Hopgood; American Elsemer Publishing Co., 1969.

Math MATHEMATICAL MODELING AND SIMULATION (Spring)

Prerequisites: Calculus I-IV, Linear Algebra, Mathematical Statistics, Introduction to Computers.

Objectives: This course is designed as an upper level elective open to junior and senior mathematics majors and majors of related fields with the proper prerequisites. A mathematical model of a real-world situation may be a system of algebraic differential, difference equations, a stochastic process, or any other mathematical structure in which the problem can be formulated, studied and given a mathematical solution. Hence the course will be broad in scope, choosing real-world situations which are not 'overly complicated' but not so 'simple' as to lose the point.

Description: Real-world situations will be chosen from the fields of biology, business, economics, physics and transportation and the corresponding mathematical models will be developed. The student will be encouraged to use whatever his mathematics background affords to model and simulate that situation. Extensive use will be made of all computing facilities, i.e., IBM 370 System, Monroe calculator, TR20 Analog Computer.

Physics 105, 106 PHYSICS I, II 4 s.h. each
(105 is offered Fall and Spring (eve.))
(106 is offered Fall (eve.) and Spring)

Prerequisites: High School Trigonometry. Math 103 may be taken as co-requisite.

Objectives: These courses will acquaint students with important physics concepts needed for advanced computer work in the sciences, engineering and applied mathematics.

Description: PHYS 105: Introduction to Mechanics, heat and sound, including: statics; dynamics; gravitation; fluids; kinetic theory; heat conduction; first and second laws of thermodynamics.

PHYS 106: Electric and magnetic fields; DC circuit analysis; AC circuit concepts; AC and DC meters; transformers; electronic devices and instruments; principles of geometrical and physical optics; topics in modern physics.

Computer applications are introduced where pertinent in both 105 & 106.

Texts: Miller, College Physics

Sears & Zemansky, University Physics

Physics 207 ELECTRONICS I (Fall) 4 s.h.

Prerequisites: Physics II

Objectives: To acquaint the student with the properties of modern electronic devices and systems so that he can analyze and construct simple electronic systems such as amplifiers and oscillators, and effectively utilize advanced electronic equipment.

Description: Introduction to DC and AC circuit analysis; properties of vacuum tubes, transistors, tranducers, integrated circuits, and operational amplifiers. The use of oscilloscopes and other electronic measuring systems, Medical electronics, and other applications.

Typical Text:
The Malmstadt and Enke Instrumentation for Scientists Series.

Physics 307 DIGITAL ELECTRONICS (Spring) 4 s.h.

Prerequisite: Physics 207 suggested but not required.

Objectives: To acquaint the student with integrated circuit logic modules and to enable him to assemble a number of important digital circuits using such modules.

Description: Principles of Boolean Algebra; Digital gating circuits, including NAND and NOR gates; FLIP-FLOPS; Shift registers; Clocks; counters; Adders and other arithmetic circuits; Memory circuits.

Physics 570, 571 COMPUTER APPLICATIONS IN THE SCIENCES I, II 3 s.h. each

Prerequisites: Calculus, Physics II. Additional work in physics, mathematics and other sciences is recommended but not required.

Objectives: To acquaint the student with the advantages and the limitations of computer and analytic problem solving in the sciences.

Description: Data analysis and a simulations in the sciences. Introduction to computer Model Building Techniques. Examples will be drawn from the fields of physics, chemistry, biology, environmental science and engineering. Typical problems include population dynamics, spread of epidemics, satellite motion, electronic field plots, radioactive decay and chemical kinetics. The material is presented so that students can deal with problems outside of their specialty.

Appendix II

Freshman Year

	Fall	
* 1.	Math 105 Calculus I	4 s.h.
* 2.	Math 107 Introduction to Computers	3 s.h.
3.	Physics 105 Physics I	4 s.h.
4.	English 101 Fundamentals of Communication	3 s.h.
5.	(General Studies Elective)	3 s.h.
	Spring	
* 1.	Math 106 Calculus II	4 s.h.
* 2.	Math 108 Intermediate Programming	3 s.h.
3.	Physics 106 Physics II	4 s.h.
4.	English 102 Fundamentals of Communication	3 s.h.
5.	(General Studies Elective)	3 s.h.

Sophomore Year

	Fall	
1.	Math 205 Calculus III	4 s.h.
* 2.	Math 409 Data Structures	3 s.h.
3.	Physics 207 Electronics I	3 s.h.
4.	Economics 251 Accounting I	3 s.h.
5.	(General Studies Elective)	3 s.h.
	Spring	
* 1.	Math 210 Applications of Math	3 s.h.
2.	Math 408 Advanced Programming	3 s.h.
3.	Physics 208 Electronics II	3 s.h.
4.	Economics 252 Accounting II	3 s.h.
5.	(General Studies Elective)	3 s.h.
6.	(General Studies Elective)	3 s.h.

Junior Year

	Fall	
* 1.	Math 207 Linear Algebra	3 s.h.
2.	Math 301 Differential Equations	3 s.h.
3.	Physics 307 Digital Electronics	3 s.h.
4.	C.C.S. Computer Techniques	3 s.h.
5.	(General Studies Elective)	3 s.h.
6.	(General Studies Elective)	3 s.h.
	Spring	
1.	Math 208 Abstract Algebra I	3 s.h.
* 2.	Math 302 Numerical Analysis	3 s.h.
3.	C.C.S. Computer Operations Practicum	3 s.h.
4.	Math 407 Machine Organization	3 s.h.
5.	(General Studies Elective)	3 s.h.

Senior Year

Fall

1.	Math 305 Mathematical Statistics I	3 s.h.
2.	Math 405 Advanced Calculus I	3 s.h.
3.	Math 409 Modeling & System Simulation	3 s.h.
4.	Physics 570 Computer Applications in the the Physical Science	3 s.h.
5.	(General Studies Elective)	3 s.h.

Spring

1.	Math 410 Theory of Compatability	3 s.h.
2.	Math 406 Mathematical Statistics II	3 s.h.
3.	Free Elective	3 s.h.
4.	Free Elective	3 s.h.
5.	Free Elective	3 s.h.

The above program includes a program of studies in Mathematics and Computer Related Mathematics consisting of the maximum of 54 semester hours of credits. It includes also a minor in Physics consisting of 30 credits. However, the only required courses for the major are the ones marked with an asterix. A student can tailor a program for himself consisting of only a minimum of 36 in mathematics as spelled out in the program description and in Computer Related Studies as a Computer Science Mathematics major at Jersey City State College.

Appendix 3

OPTION IN COMPUTER STUDIES

A special academic option (specialization) is offered in computer studies under the supervision of an interdepartmental committee of advisors.

Computer Studies Advisors

P. Goldstein, Coordinator of Computer Studies	Physics
P. Caverly	Mathematics
G. Daneluk	Mathematics
T. Michnowicz	Mathematics
C. Miller	Biology
C. Wanner	Pol. Science
J. Grauman	Physics

By providing a Computer Studies specialization, the College recognizes that computer and information science is a highly dynamic and growing field. In general, it is a subject that deals with the storage and retrieval of information, methods of analysis, its display and its processing.

The numerous applications of computers are legendary. Computers are essential to the space program. Computers aid in all types of design, they are used to control all types of processes, to make out bills and payrolls. Computers retrieve medial information and keep critically ill patients under constant supervision. Computers aid in urban design and planning. In fact, there is virtually no area of human endeavor which is not making some use of the computer. But even more important than its pervasiveness is that the computer is significantly changing the way things are being done. This trend will accelerate in the future as more and more tasks are performed by computers. It is important that as many people as possible have some understanding of the computer and the role it plays in their fields as well as its impact on society.

The Computer Studies courses are designed to familiarize students with computers and their applications. The courses will be of benefit to those who are majoring in fields such as the sciences, mathematics, and the social sciences; those who are planning to teach in the public schools; and those who have a general interest in computers and want to acquire some knowledge of how they work.

Students who are not sure which courses meet their career objectives should consult the Computer Studies Coordinator or one of the Computer Studies advisors.

The Option is intended to supplement the student's major and provides for some variability in individual programs of study depending on the interests and background of the student. Students who plan to pursue the Computer Studies Option should consult the Coordinator of Computer Studies, or one of the Computer Studies advisors as soon as possible in order to plan a program of study. The importance of early advisement cannot be over-emphasized.

Certificate of Proficiency

To obtain a Certificate of Proficiency students must:

1. Complete the normal requirements of the College and their major department.
2. Complete the specialization in Computer Studies as described.

Admission and Plan of Study

A student is accepted for study in Computer Studies by the Committee Coordinator. A designated member of the committee will help the student develop an appropriate plan of study.

The Computer Studies Option # Consists of 21 s.h. of Courses as Follows:

Required Courses

1.	MATH	107	Introduction to Computers This is a fundamental course and should be taken as soon as possible	3 s.h.
2.	MATH	108	Intermediate Programming*	
	or			
	MATH	408	Advanced Programming*	3 s.h.
3.	MATH	409	Data Structures*	
			or	
			Statistics XX**	3 s.h.
4.	MATH	105	Calculus I	3 s.h.
	or		Physics 115: Man Made World I	
	or		Math 115: Man Made World I	
	or		Numerical Calculus I (this course is being considered)	
5.	MATH	407	Machine Organization*	3 s.h.
	or		Physics 307: Digital Electronics†	
	or		Physics 116: Man Made World II	
	or		Math 116: Man Made World II	
6.			Computer Studies Electives	

*Indicates that Math 107 is a prerequisite.

**This course is being developed and details are not currently available.

†Indicates that Math 107 is a suggested prerequisite.

#A Business Data Processing Sequence is being developed. Students interested in this area should take Computer Science 220, 221: Cobol Programming and Business Applications I, II. Math 109 may be substituted for Comp. Sci. 220.

Computer Studies Electives

MATH 109: Cobol Programming
MATH 115, 116: Man Made World I, II
MATH 210: Applications of Mathematics
MATH 301: Differential Equations: Applications to Numerical Solutions
MATH 302: Numerical Analysis and Programming
MATH 407: Machine Organization
MATH 408: Advanced Programming
PHYSICS 115, 116: Man Made World I, II
PHYSICS 307: Digital Electronics
PHYSICS 570, 571: Computer Applications in the Physical Sciences I, II
POL. SCI. 344: Law and Politics
POL. SCI. 400: Scope and Methods of Political Science
POL. SCI. 415: Research Problems in the Study of Trial Courts
COMPUTER SCI. 200: Assembly Language Programming
COMPUTER SCI. 220, 221: Cobol Programming and Business Applications I, II
COMPUTER SCI. 341, 342: Systems Programming I, II
COMPUTER SCI. 361: Operating Systems
COMPUTER SCI. 373: Analog Computation
COMPUTER SCI. 441, 442: Independent Study
COMPUTER SCI. 521: Computer-Aided-Instruction-Methods & Languages

The following courses are under consideration:

Computers and Society
Computer Applications in the Social Sciences I, II
Computer Applications in Biology I, II
Numerical Calculus I, II

Appendix 4

JERSEY CITY STATE COLLEGE
MATHEMATICS DEPARTMENT

Saturday Course in
Computers and Computer Related Mathematics
For High School Students

State of New Jersey

JERSEY CITY STATE COLLEGE

JERSEY CITY, N. J. 07305

December 12, 1973

Dear Sir:

The Mathematics Department of Jersey City State College is conducting a Saturday course in Computers and Computer Related Mathematics for high school students this coming semester. It is hoped that such a course will provide high school students in our community interested in pursuing studies in mathematical and computer related sciences an opportunity to participate in a program designed to encourage this interest by way of enrichment and reinforcement.

Enclosed are brochures describing the program and application forms which we would very much appreciate your distributing to teachers in contact with students who might be interested in participating.

Thank you for your kind cooperation.

Sincerely,

George Daneluk

George Daneluk, Associate Professor
Mathematics Department

Enc.

PROGRAM OBJECTIVES

The objective of the Saturday Program is to provide high school students an opportunity to take a course designed to: (1) introduce the student to mathematical methods for solving problems using the computer, (2) give first hand experience with the techniques of computer programming, and (3) reinforce and enrich the students current high school studies in mathematics.

PROGRAM

Students meet every Saturday at Jersey City State College campus from 11:00 A.M. to 1:00 P.M. for a one hour lecture session followed by a one hour laboratory session.

COURSE DESCRIPTION

The purpose of the course is to provide a student with an understanding of how to use a computer to solve problems. Participants will be introduced to selected topics in applied mathematics, including methods for the evaluation of functions, methods for the solution of equations and systems of equations, methods for approximating functions, and computational statistics. Students will study Fortran, a symbolic programming system, in order to obtain computer solutions to problems. The course will also include topics designed to provide an understanding of what a computer is from a hardware point of view.

LABORATORY

Students will receive hands-on experience verifying programs on an IBM 1130 Computing System and on remote terminals. Both systems are located on the college campus. The systems are also linked by telephone to the IBM 360 located on the Rutgers campus.

COST

There is no tuition charge or fee to the participant. The course is being funded by Jersey City State College as a service to the community. Students will be expected to make their own travel arrangements to and from the college. Students may also be asked to purchase a book to be used in connection with the course.

CREDIT

No academic credit will be granted for the course but certificates of completion will be given by the Mathematics Department to those who complete the course.

ELIGIBILITY

Participants must be high school students who have completed the second year course in high school algebra or who will be taking it concurrently with Computers and Computer Related Mathematics. Preference will be given to students with an interest in science and mathematics. Thirty-five participants will be selected from the applicants.

APPLICATION PROCEDURE

To be considered for the course a student must fill out an application form and submit it by February 9, 1974. Selected participants will be notified by February 16, 1974. Application forms and further information may be obtained by writing or calling:

George Daneluk
Mathematics Department
Jersey City State College
Jersey City, N.J. 07305
Telephone: 547-3201

Application Form
Computers & Computer Related Mathematics
Mathematics Department
Jersey City State College

Please print the following information and return by February 9, 1974.

NAME:

ADDRESS:

HOME PHONE:

NAME OF HIGH SCHOOL:

AGE_____ SEX_____ GRADE LEVEL _____

LIST ALL MATHEMATICS AND SCIENCE COURSES TAKEN OR IN PROGRESS:

MATH COURSE TITLE	GRADE EARNED	SCIENCE COURSE TITLE	GRADE EARNED

Describe any previous experience with the computer if any.

Describe any extra-curricular activities you have engaged in.

State why you would like to take the Saturday course.

SIGNATURE OF MATHEMATICS TEACHER RECOMMENDING YOU FOR THE COURSE

_____

SIGNATURE OF PARENT OR GUARDIAN PERMITTING YOU TO TAKE THE COURSE

_____

YOUR SIGNATURE _____

PRINCIPAL OF
ST JOSEPHS GIRL
WEST NEW YORK NJ 07093

PRINCIPAL OF
BAYONNE HS
BAYONNE NJ 07002

PRINCIPAL OF
MARIST H.S.
BAYONNE NJ 07002

PRINCIPAL OF
HUBOKEN HS
HOBOKEN NJ 07030

PRINCIPAL OF
EMERSON HS
UNION CITY NJ 07087

PRINCIPAL OF
UNION HILL HS
UNION CITY NJ 07087

PRINCIPAL OF
HOLY ROSARY ACAD
UNION CITY NJ 07087

PRINCIPAL OF
ST. MICHAEL H.S.
UNION CITY NJ 07087

PRINCIPAL OF
ST.MARY H.S.
JERSEY CITY NJ 07302

PRINCIPAL OF
ST.MICHAEL H.S.
JERSEY CITY NJ 07302

PRINCIPAL OF
THE BERGEN SCH.
JERSEY CITY NJ 07304

PRINCIPAL OF
HUDSON CATHOLIC
JERSEY CITY NJ 07304

PRINCIPAL OF
ACA ST. ALOYSIUS
JERSEY CITY NJ 07304

PRINCIPAL OF
ACC EVENING HS
JERSEY CITY NJ 07306

PRINCIPAL OF
ST.ALOYSIUS H.S.
JERSEY CITY NJ 07306

PRINCIPAL OF
JERSEY ACADEMY
JERSEY CITY NJ 07306

PRINCIPAL OF

W N Y MEM HS
WEST NEW YORK NJ 07093

PRINCIPAL OF

ST JOSEPHS BOYS
WEST NEW YORK NJ 07093

PRINCIPAL OF

ACC EVENING HS
BAYONNE NJ 07002

PRINCIPAL OF

HOLY FAMILY
BAYONNE NJ 07002

PRINCIPAL OF

ACA.SACRED HEART
HOBOKEN NJ 07030

PRINCIPAL OF

STEVENS ACADEMY
HOBOKEN NJ 07030

PRINCIPAL OF

WEEHAWKEN HS
WEEHAWKEN NJ 07087

PRINCIPAL OF

HOLY FAMILY SCH
UNION CITY NJ 07087

PRINCIPAL OF

J FERRIS HS
JERSEY CITY NJ 07302

PRINCIPAL OF

ST.ANTHONY H.S.
JERSEY CITY NJ 07302

PRINCIPAL OF

ST. PETERS PREP
JERSEY CITY NJ 07302

PRINCIPAL OF

A LINCOLN HS
JERSEY CITY NJ 07304

PRINCIPAL OF

ST.DOMINIC ACA.
JERSEY CITY NJ 07304

PRINCIPAL OF

H SNYDER HS
JERSEY CITY NJ 07305

PRINCIPAL OF

A H MOORE LAB HS
JERSEY CITY NJ 07306

PRINCIPAL OF

DICKINSON HS
JERSEY CITY NJ 07306

PRINCIPAL OF

ROGOSIN YESH.HS
JERSEY CITY NJ 07306

Appendix 5

State of New Jersey
DEPARTMENT OF HIGHER EDUCATION
225 WEST STATE STREET
P. O. BOX 1293
TRENTON, NEW JERSEY 08625

M E M O R A N D U M

October 18, 1968

TO: Distribution

FROM: Thomas M. Wendel

SUBJECT: Minutes of Meeting Held at Jersey City State College, October 16, to Discuss Computer Network Possibilities

Mr. George Daneluk called the subject meeting with attendance as shown by the distribution list. The meeting was the result of some earlier talks between Mr. Daneluk, Dr. Thomas Mott and others of the computer center staff at Rutgers University.

Mr. Daneluk began his talks with Rutgers to determine the possiblility of using the IBM 1130 computer at Jersey City State College as a terminal to a larger computer at Rutgers. These preliminary discussions indicated that the 1130 could become a terminal to the soon-to-be-installed IBM 360/67 at the Rutgers center in New Brunswick. Since such a tie-up promises significant computing power to the 1130 user and because Rutgers is very much interested in the marriage, Mr. Daneluk called the subject meeting to determine if a wider interest exists.

Dr. Mott briefly outlined the immediate plans of the computer center at Rutgers. The IBM 360/67 is to be installed in January. In April an IBM 1130 at Rutgers will be used as a terminal to the 360/67. By July, a 2700 will be installed. Thus, by the summer of 1969, Rutgers intends to be in a position to offer the State Colleges the opportunity of using the 1130's as input/output terminals to the 360/67.

Charges to the colleges should be as indicated on the attached statement of anticipated expenditures.

There was considerable discussion of the proposed 360 charges by Rutgers. Dr. W. Carroll speaking for Rutgers made two points:

1. The 1130 users will be charged only for the computer resources they require and for the time they are required.

2. The 1130 users will be treated the same as any Rutgers user. This means equality in rates, priorities, and opportunity to influence the character of the services provided.

Mr. Daneluk indicated he wished to see the tie-up become a fact and he asked Tom Wendel how the Chancellor's office would review this marriage. In particular, would the Chancellor's office try to find State or Federal funds to support this effort.

Mr. Wendel replied that any funding agency including the bursar at the individual college would need to know:

1. What is the proposed use, usage and workload of the 1130's in their own environments?

2. What does the 360/67 tie-up offer the college that the 1130 alone does not?

3. What are the economic or service values of these gains and what are the alternatives?

Mr. Wendel felt that the individual schools should be able to articulate these data as a prerequisite to entering into a computer to computer tie-up.

Mr. Daneluck replied that getting data of this form could take months, satisfy no one, result in redundant effort and false starts and finally dissipate in another lost opportunity. During the ensuing discussion it was also pointed out that it would be best to pursue this activity in terms of a computer network for all the State Colleges -- not any one. On this there was unanimous agreement.

It was suggested that the Chancellor's office should be the coordinating force for continuing this effort. Again, there was unanimous agreement. As a result, Mr. Wendel agreed to:

1. Write the minutes of this meeting.

2. Investigate other network possibilities.

3. Obtain some information concerning the availability of NSF or other outside funds and what is required to obtain them.

4. Call the next meeting.

Concurrently State College representatives agreed to begin working to satisfy the three requirements listed above.

At this point the meeting terminated.

Attachments: as stated

D I S T R I B U T I O N

COLLEGE COMPUTER GROUP REPRESENTATIVES

Tom Wendel	Coordinator of Computer Activities - Chancellor's Office
Thomas Mott	Director of Center for Information and Data Processing - Rutgers U
Hank Matelson	Coordinator of Computer Center - Trenton State College
Richard Cwiakala	Coordinator of Data Processing, Newark State College
Tony Notare	Director of Data Processing - Montclair State College
Rudy Saladi	Registrar - Glassboro State College
Bob Kroeckel	Professor of Mathematics - Paterson State College
George Daneluk	Professor of Mathematics - Jersey City State College
Dick Fields	Account Manager - IBM
Al Kovitz	Senior Systems Engineer - IBM

Appendix 6

STUDENT ATTITUDINAL QUESTIONNAIRE No Name Necessary

Directions: Please indicate your response(s) to the questions by placing a check mark in the appropriate blank spaces. Also, please feel free not to respond to any question.

1 - What is your present (if you are a junior or senior) or anticipated (if you are a freshman or sophmore) major area of study?

N = 440

N R = Percent of No Response

Numbers listed in the spaces to the left represents the percent of students checking that response.

15.9% Liberal Arts
15.0% Social Sciences
4.5% Physical Sciences
28.6% Professional Studies
20.7% Education
10.9% Other (specify) _____
4.3% (N R = No Response)

2 - Sex: 49.5% Male 47.3% Female 3.2% N R

3 - Present class standing:

28.0% Freshman
20.2% Sophmore
25.9% Junior
21.1% Senior
2.5% Other (specify) _____
2.2% (N R)

4 - Ethnicity:

61.1% White
15.9% Black
10.5% Hispanic
4.1% Oriental
4.5% Other (specify) _____
3.8% (N R)

5 - What is the approximate annual income of your family?

12.7%	$5,000 or less
12.3%	$5,001 - 10,000
16.8%	$10,001 - 15,000
18.9%	$15,001 - 20,000
17.0%	$20,001 - 30,000
7.0%	More than $30,000
15.2%	(N R)

6 - Do you have a job?

31.4%	No
9.5%	10 hours or less
15.0%	11 to 20 hours
11.8%	21 to 30 hours
23.4%	31 to 40 hours
7.3%	More than 41 hours
1.6%	(N R)

7 - Is a language other than English spoken in your home?

37.5% Yes 61.1% No 1.3% (N R)

8 - Marital status:

69.3%	Single
20.9%	Married
3.9%	Separated
2.0%	Divorced
1.6%	Cohabitating
0.7%	Widowed
0.5%	Other (specify) _____
1.1%	(N R)

9 - Which of the following geographical locations is your legal residence?

36.1%	Jersey City
13.9%	Bayonne
4.5%	Hoboken
4.3%	North Bergen
8.4%	Union City
31.6%	Other (specify) _____
1.1%	(N R)

10 - Age:

41.6%	17 - 21
27.3%	22 - 25
22.3%	26 - 35
8.2%	Over 35
0.6%	(N R)

11 - Do you view yourself as being primarily a (n):

73.2%	Day student (taking most courses between 8:00 am to 4:00 pm)
24.1%	Evening student (taking most courses after 4:00 pm)
2.7%	(N R)

* *

Qyestions 12 - 15 pertain to concerts, lectures, exhibits, and special events sponsored by JCSC during the academic year.

12 - Are you aware of the number and variety of these events?

46.6%	Yes
41.4%	No
10.5%	Don't know
1.6%	(N R)

13 - Which one source among the following is the best means of information?

33.0% Announcements in class
28.6% College newspaper
24.8% Printed brochures of concerts, exhibits, or athletic events
8.2% Other sources (describe) _____
5.4% (N R)

14 - Do you feel foreign-born students are actively participating in social and cultural events?

25.5% Yes
21.1% No
50.9% Don't know
2.5% (N R)

15 - Do you feel American-born students are actively participating in social and cultural events?

43.0% Yes
17.0% No
37.3% Don't know
2.7% (N R)

16 - Knowing what you now know, if you were now deciding upon a college to attend, would you choose to attend JCSC?

53.2% Yes
30.7% No
12.7% Don't know
3.4% (N R)

17 - Are faculty interested in students?

57.0% Yes
19.5% No
19.5% Don't know
3.8% (N R)

18 - Is faculty interest in students important?

80.2% Yes
8.0% No
9.1% Don't know
2.7% (N R)

19 - How important is getting good grades to you personally?

70.2% Very important
25.0% Fairly important
3.6% Unimportant
1.1% (N R)

20 - As a result of your total college experience, in what size class do you feel you would achieve your best grades?

20.0% 10 or less
55.7% 11 - 20
16.8% 21 - 30
2.5% Over 30
4.9% (N R)

21 - How important are social and cultural activities to a good college atmosphere?

2.3% Not at all important
4.1% Not very important
12.5% No feeling one way or the other
41.4% Somewhat important
38.0% Very important
2.1% (N R)

If you answered "Not at all important" on question 21, ignore question 22.

22 - On the following list, indicate by numbering from "1" through "5" the primary reasons for not participating in social and cultural activities (number "1" being of primary importance, number "5" being of little importance).

Rank:		Totals:
2	Not aware of events	893
4	Inadequate publication and advertising	731
1	Conflict with other obligations	1422
3	Activities aren't geared to my interest(s)	767
5	Commuting to and from such activities	715

23 - The majority of faculty with whom you have studied at JCSC (you may check more than one)

58.0%	Are academically well-prepared
30.5%	Take a personal interest in the students
38.9%	Plan well for their classes
28.9%	Make classwork interesting
35.9%	Evaluate students fairly
23.9%	Make reasonable content assignments (assignments that are to the point)
9.5%	Involve students in planning course activities

24 - Which of the items below fit most faculty here at school? (You may check more than one)

58.0%	They are friendly
7.3%	They are too strict
8.2%	They are too easy with school work
31.6%	They understand the problems of college students
14.6%	They are not interested in college students
10.0%	They are willing to sponsor and help out in student activities
8.6%	They are too easy with discipline
19.5%	They are difficult to get to know

25 - The following list represents some areas in which it has been suggested that the college should be more actively involved, or in which changes should be made. Please indicate with a check mark (more than one area, if appropriate) those areas which you feel should be of concern to the college. In the last blank space provided, please add any other areas that you feel should be included.

48.6%	Academic achievement and standards
57.3%	Parking
24.8%	Maintenance of physical campus (halls, classrooms, and grounds)
17.3%	Class attendance policies
29.8%	Grading methods and policies
20.7%	Involvement in the problems of metropolitan
22.5%	Classification of the aims of the college
32.5%	Scheduling of classes
24.3%	Involvement of students in academic affairs
26.6%	Social activites on campus
7.7%	Other (specify) _____

Statement	Strongly Disagree	Disagree	Indiff-erent	Agree	Strongly Agree	NR
37. The library is adequate for your needs.....	11.4%	15.5%	9.8%	50.7%	8.5%	4.3%
38. Course standards are unrealistically high.....	12.5%	44.5%	23.6%	10.7%	2.7%	5.9%
39. Instructors are too lenient with regard to grades they give the students in their classes.	13.4%	47.5%	18.4%	12.7%	2.3%	5.7%
40. Students tend to take pride in the achievement of the school.....	12.7%	26.4%	22.5%	28.0%	3.4%	7.0%
41. Most instructors know their subject matter.....	3.0%	7.7%	8.4%	64.1%	13.0%	3.8%
42. The general quality of instruction is commendable	3.0%	10.0%	15.7%	61.4%	4.8%	5.2%
43. The general climate at JCSC is exciting and intellectually stimulating....	14.1%	30.2%	23.4%	24.8%	2.5%	5.0%
44. The nature of my course work is relavant to my career goals.....	4.8%	8.0%	9.1%	52.0%	20.9%	5.2%
45. Upon graduation, I feel my academic needs will have been met.....	5.7%	15.9%	11.6%	51.4%	9.1%	6.3%
46. The extent to which I am being prepared for my future career goals is due primarily to the academic climate of the college.....	6.1%	26.4%	17.7%	37.0%	4.3%	8.4%
47. Adequate academic counseling is provided for students.....	16.8%	22.3%	14.5%	34.1%	5.7%	6.6%

* *

Statement	Strongly Disagree	Disagree	Indifferent	Agree	Strongly Agree	NR
	Place an "X" in the Appropriate Space					
26. The college has adequate design features for the physically handicapped.....	86.6%	17.7%	22.5%	38.6%	5.0%	7.5%
27. There are adequate parking facilities.....	48.9%	29.3%	5.7%	10.9%	0.9%	4.3%
28. The campus grounds are properly maintained.....	3.9%	13.4%	12.7%	58.9%	5.9%	5.2%
29. There are sufficient accommodations for students who want to live on campus.....	19.5%	17.3%	42.0%	11.4%	1.4%	8.4%
30. Classrooms are maintained properly (heating, cleanliness, etc.).....	20.0%	35.0%	10.7%	26.6%	4.3%	3.4%
31. There is enough landscaping on campus (trees, plants, etc.).....	13.6%	27.5%	15.0%	34.1%	6.4%	3.4%
32. The on-campus eating facilities accommodate students' schedules.....	10.9%	15.2%	28.9%	34.5%	4.3%	6.1%
33. There should be provisions made for child care on campus for student parents.....	6.8%	10.0%	30.7%	30.2%	16.4%	5.9%
34. There are enough buildings to accommodate the needs of the college community (students, administration, & faculty).....	7.7%	18.0%	22.0%	42.3%	3.4%	6.6%
35. There are adequate facilities for students to study on campus.....	9.8%	26.6%	11.6%	42.0%	4.1%	5.9%
36. Jersey City State College is a physically attractive campus.....	10.2%	21.6%	16.8%	40.9%	5.5%	5.0%

48. Which one of the following has given you your greatest personal satisfaction at Jersey City State College (check only one)?

20.5%	General coursework
34.8%	Coursework in major field
6.1%	Individual study activities
3.6%	Organized extra curricullar activities
3.0%	Social life: dating, parties, etc.
2.0%	"Bull Sessions" with fellow students
24.1%	Self-discovery, self-insight, discovery of new interests, talents, etc.

49. List the four most important reasons for selecting JCSC. (With "1" being the most important and "4" being the least important)

Rank:		Totals:
9	Relatives wanted me to come here	154
8	Teacher advised me	159
6	Has good academic reputation	303
5	Offered financial assistance	319
10	Not accepted anywhere else	85
3	Advice of someone who attended	401
4	Offers special education programs	374
1	Has low tuition	826
11	Advice of guidance counselor	83
2	Wanted to live at home	562
7	Friend suggested attending	249
12	College representative recruited my	81

50. If you had to choose one thing about JCSC that you like the most, what would it be?

_____

_____

51. If you had to choose one thing about JCSC that you dislike the most, what would it be?

_____

_____

Appendix 7

DEVELOPMENT TRACE

JERSEY CITY
STATE COLLEGE

MEMORANDUM

TO: Dr. Herman Rosenberg

FROM: George Daneluk

DATE: February 14, 1979

SUBJECT: Curriculum in Information Systems Development

Attached is a preliminary draft of a curriculum in Information Systems Development. Course Outlines and a proposal for adapting the curriculum into the Computer Science Mathematics Specialization and into a minor for the Business student will be forthcoming. The following people from the Systems Technical Education Center (STEC) of A. T. & T. Long Lines and from Bell Laboratories have kindly offered their assistance in developing the modules and possibly teaching the courses:

1. Suzanne Hain, Curriculum Manager in Systems Development Training at STEC.
2. Bruce Kittinger, Instructor in System Development Training at STEC.
3. Lee Ann Whitaker, Instructor in Advanced Programmer Training at STEC.
4. Valerie D'Antonio, Course Developer in Advanced Programmer Training at STEC.
5. George Foley, Member of the Staff at Bell Telephone Lab.

cc: Dr. Joseph Drew, Academic Vice-President
Dr. Mary Jane Clark, Dean of Arts & Science

Att.

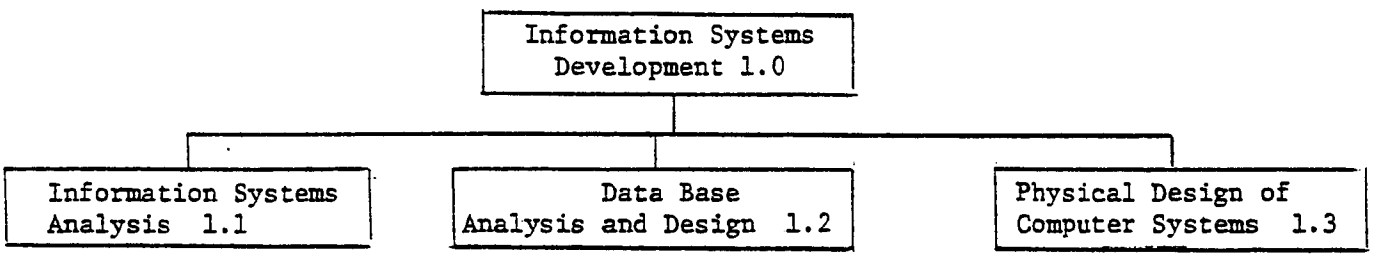
CURRICULUM IN INFORMATION SYSTEMS DEVELOPMENT

Objectives

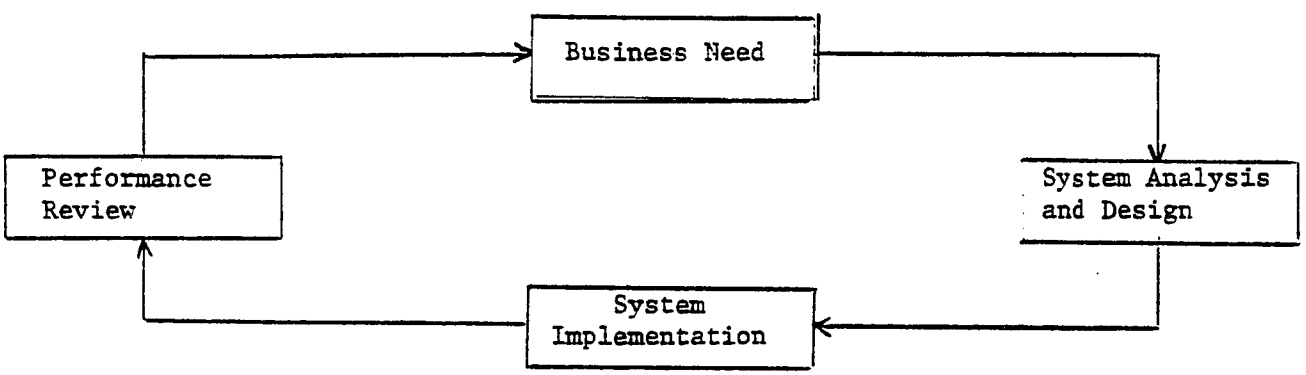
The purpose of the curriculum is to provide the student with an understanding of the total life cycle of information systems development in a business environment. The sequence of courses would constitute a component of the Computer Science Mathematics Specialization and a component of a minor in Information System Development for the Business major.

Content

The basic modules for the curriculum are shown on level 2 of the chart below. Each module represents two semester length courses covering concepts and methodology in the phases of business system development.



Module 1.1 will cover topics in Total Systems Development Concepts, Systems Analysis, Project Management, Structured Systems Analysis and Data Gathering. Module 1.2 will cover topics in Data Base Concepts, Logical Data Base Design, Data Base Management Systems and IMS Total Data Base. Module 1.3 will cover topics in IMS Implementation, Module Design, Structured Design, File Design and Computer Subsystem Design. Topics will be selected and sequenced to show the relationship of Information Systems Development to the need to solve business problems and to take advantage of business opportunities. A schematic representation of the life cycle of the systems development process as related to business needs is shown below.



JERSEY CITY
STATE COLLEGE

MEMORANDUM

TO: Dr. Herman Rosenberg

FROM: George Daneluk

DATE: 3-9-79

SUBJECT: Curriculum in Information Systems Development

I thank you for the suggestion concerning the course development project in Information Systems Development outlined in my statement of intent of 2-14-79. Attached is a memorandum to the members of the Mathematics Department requesting input regarding the content, scope, and appropriateness of the courses.

On the issues raised in your memorandum of 2-20-79 I would like to offer the following comments:

1. Number of courses: Although we are only in the planning stage, the current thinking is that four or five courses would be sufficient to cover the topics suggested.

2. Impact on present major and minor programs: The intent is that the courses would supplement with electives not replace any part of existing major and/or minor programs. I agree, however, that the set of courses constitutes a substantial extension of the committment by the Mathematics Department to Computer Science education.

3. Role of the Business Administration Department: Since almost all of our Computer Science Mathematics graduates are employed by business, the priority objective of the courses is to help meet the needs of our own majors in this respect. The methodology, moreover, has an underlying mathematical base, so it is natural the Mathematics Department have the primary concern. The problem is one of getting a proper balance between theory and practice. Hopefully, the proposed plan for developing the courses would help achieve a greater degree of balance. The courses, nevertheless, are designed primarily for technical students who wish to go into business, not for the business majors per se.

4. Level of Courses: The level of abstraction of such courses offered in business is not as high as most of our own intermediate and advanced level required mathematics courses, e.g. linear algebra, numerical analysis etc. The intent is that students get an idea of how some of theoretical concepts learned in some of our present courses are applied in the real world of business systems development.

5. Interface with Current Offerings: The real question which should be asked is how our current offerings "mesh" with the needs of students after they graduate. The intent of the plan for developing the courses is to help make our current offerings more relevant to those needs. But to the point I would merely suggest that we follow the recommendations and guidelines of professional societies, eg. A.C.M. C.U.P.M. IEEE.

6. Audience: There is no lack of an audience in business for up-to-date training in a rapidly changing technology. Mathematics itself has played a key role in bringing about these technological developments. The intent here is that the courses be developed as part of our overall effort to keep up with the changes. I am convinced that the quality of the product will attract sufficient numbers of students to satisfy any administrative constraints.

Although I have discussed the project with the people named in my memorandum of 2/14/79 and, on the basis of your prior verbal approval, asked them to begin thinking about possible designs, I also asked them to hold off any work until I had received your response to the plan. Since, in my opinion, what we need most at this point is input from experts in course development in the subject matter areas in question, I strongly urge that we <u>now</u> move ahead with the project as planned, including the solicitation of input from the members of the Mathematics Department on the issues you raised. However, if you feel that you cannot authorize the project based on your own assessment of the needs, then I will hold.

Please advise.

/ck

Copy to: Dr. Joseph Drew, Academic Vice-President
Dr. Mary Clark, Dean of Arts and Sciences

JERSEY CITY
STATE COLLEGE

MEMORANDUM

TO: Members of the Mathematics Department

FROM: George Daneluk

DATE: March 9, 1979

SUBJECT: Curriculum in Information Systems Development

Attached is a statement of intent with respect to a curriculum in Information Systems Development. I would appreciate any input you may have regarding the content, scope, and appropriateness of the planned courses. I would also appreciate your comments on the following specific questions:

1. How many courses do you consider necessary to cover the content and methodology in the subject areas listed.

2. Do you think the level of the courses in the subject areas suggested is appropriate for undergraduate students at J.C.S.C.

3. How do you think courses in the subject areas suggested should be "meshed" with current offerings at J.C.S.C.

4. Do you think that there is potential for an adequate audience for such courses at J.C.S.C.

5. What role should the Business Administration Department play in the development of the courses.

6. What impact do you feel the courses would have on the existing majors and minors offered in the Mathematics Department.

JERSEY CITY
STATE COLLEGE

MEMORANDUM

TO: Dr. Mary J. Clark, Dean, Arts and Sciences

FROM: George Daneluk, Mathematics

DATE: 10-31-79

SUBJECT: Computer Science Curriculum

Attached is a structure chart for the Computer Science Curriculum which the Computer Science Curriculum Committee is recommending as the basis for the proposal to the N.J. Board of High Education. The curriculum has a number of general characteristics I would like to draw your attention to:

1. It incorporates most of the computer related courses which have already been developed in the Mathematics and Physics Departments. Of the thirty-one courses represented eighteen already exist.

2. It incorporates most of the courses recommended by A.C.M., and conforms reasonably closely to the recommendations of IEEE concept-wise by having fairly well defined components in Hardware Organization and Software Engineering.

3. A unique feature is the concept underlying the component in Business Information Systems Development. The second attachment describes the underlying concept.

4. The structure is balanced with respect to the distribution of courses over levels, e.g., it is not overloaded with advanced courses nor with introductory courses either.

5. The curriculum allows for substantial specialization in four basic areas. However, it is not pretentious with respect to scope.

6. The mathematics component is a strong one and assures a sound grounding in the theory.

I believe it is a good program; it is feasible in terms of the resources likely to be available at JCSC in the foreseeable future; if implemented, it would provide students with a background sufficient to enable them to go into the field as entry level computer professionals.

The remaining work is to provide supporting documentation on detail and rationale, and a plan for implementation (I will submit a position paper soon regarding the latter.) Your concurrence with regard to concept and structure would be helpful at this juncture.

Att:

Copy to: Mr. Philip Caverly Mathematics
Dr. Philip Goldstein Physics
Mr. Carl Holloman Computer Center
Ms. Magda Efros Computer Center.

JERSEY CITY
STATE COLLEGE

MEMORANDUM

TO: Dr. Mary Jane Clark, Dean, Arts and Sciences

FROM: George Daneluk, Mathematics Department

DATE: 11-7-79

SUBJECT: A Computer Science Department

I support the initiative you have taken to institute a major program in Computer Science at JCSC. The formation of the Computer Science Curriculum Committee to develop a proposal to the New Jersey Board of Higher Education was an appropriate first step. As you saw, however, the concept and structure for a program was slow in coming. The reason was not due to any lack of ideas about curriculum, but rather to uncertainties about how a curriculum should eventually be implemented. Since these uncertainties will continue to plague development and since a proposal to BHE should include a plan for implementation anyway, and since the quality of an eventual program will depend on the quality of prior planning, I am taking an initiative with respect to a position on the key question.

The position is that a distinct Department of Computer Science should be instituted at JCSC to offer major, minor, and service programs in computer science. The scope of computing technology is such that a core of faculty devoted full time to developing and maintaining programs is essential.

Given a role in setting up a department, I would be willing to contribute on a full time basis. I have discussed the possibility with Dr. Phil Goldstein, Carl Holloman, Dr. Richard Riggs, John Raines, George Foley (adjunct), Magda Efros, Gloria Gardner and others and found support for the concept and like willingness to contribute. I have taken the initiative of asking those willing to contribute full time to submit a brief position paper for your consideration. With a core consisting of the above named people and at least three new full time people hired over the next five year period and use of adjuncts for specialized courses, a basis exists for the formation of a quality department offering a full curriculum. A feature of moving in the manner I am suggesting is that the initiative, planning, etc. would be coming mainly from members of the Mathematics Department which is from whom it should come.

I would suggest a meeting of Phil Goldstein, myself and you to discuss options, problems, opportunities, and strategies.

Copy to: Dr. Joseph Drew, Vice-President, Academic Affairs

jersey city state college

memorandum

TO: Dr. Mary J. Clark, Dean of Arts & Sciences

FROM: Mr. George Daneluk, Mathematics Department

DATE: 1-31-80

SUBJECT: Curriculum in Computer Science

Enclosed is a copy (unedited) of the documentation of Curriculum in Computer Sceince being recommended by members of the Computer Science Curriculum Development Committee. The document contains information sufficient to evaluate the quality of the program and its feasibility at JCSC by the standards set by the N.J. Board of Higher Education, and except for bibliographies the course outlines meet the standards for approval of the Senate Curriculum and Instruction Committee.

The document is being submitted for your preliminary consideration. Any input you have before the final editing task begins would be appreciated. The program is a sound one and I recommend it with confidence.

Enc.

Copy to: Dr. P. Goldstein, Physics Department
Magda Efros, Computer Center
P. Caverly, Mathematics Department

P.S. "Thunder is good, thunder is impressive, but it is the lightening that does the work." Mark Twain.

State of New Jersey

JERSEY CITY STATE COLLEGE
JERSEY CITY, N. J. 07305

WILLIAM J. MAXWELL PRESIDENT

TO: Valerie D'Antonio, Staff Supervisor, (ATT Long Lines)
George Foley, Member of the Staff (Bell Laboratories)
Suzanne Hain, Curriculum Manager (ATT Long Lines)
Anthony Montanaro, Director of Administrative Computing (Rutgers)

FROM: George Daneluk, Mathematics Department

DATE: February 1, 1980

SUBJECT: Consultation Service to the Computer Science Curriculum Development Committee

On behalf of the Computer Science Curriculum Development Committee at Jersey City State College, I wish to express our appreciation for the valuable input you provided with respect to the training needs of entry level computer professionals in your organizations. Attached is a copy of a structure chart for the curriculum which we are recommending as the basis for a proposal to the New Jersey Board of Higher Education. We have incorporated features which hopefully will provide our students with the skills and knowledge needed to better perform their tasks in the field after they graduate.

Thanks again for your kind cooperation in meeting with us.

Att.

Copy to: Dr. Mary Clark, Dean of Arts & Sciences
Dr. P. Goldstein, Physics Department
Mr. Carl Holloman, Director, Center for Computing Services
Mr. P. Caverly, Mathematics Department
Mrs. Magda Efros, Center for Computing Services

P.S. "Heaven for climate, hell for company." Mark Twain

jersey city state college

memorandum

To: Members of the Computer Science Curriculum Development Committee

From: George Daneluk

Date: February 26, 1980

Subject: Administrative Structure for the Curriculum in Computer Science

Attached for your consideration is an alternative proposal for an administrative structure for the Curriculum in Computer Science. The proposed structure provides for:

1) A core of faculty whose major commitment would be to the development and maintenance of programs in computer science.

2) Use of the expertise of the faculty of the current Mathematics Department.

3) Use of the expertise of faculty of other departments of the college

4) Close coordination of the development of programs in computer science with the development of other mathematical sciences at the college.

5) Functional independence.

6) Cost effectiveness.

Administrative Structure

The present Mathematics Department shall be reconstituted as the Department of Mathematical Sciences comprised of three divisions, namely, the Division of Computer Science, the Division of Mathematics Education, and the Division of Statistics. The faculty of the Division of Computer Science shall be responsible for the development and maintenance of programs in computer and information science. The faculty of the Division of Mathematics Education shall be responsible for the development and maintenance of programs in basic skills, mathematics education (both graduate and undergraduate), and pure mathematics. The Division of Statistics shall be responsible for the development and maintenance of programs in statistics. All courses offered in the department will have a course number with either a CS, MATH, or STAT prefix.

Each division of the Department of Mathematical Sciences shall be headed by a Division Coordinator elected by the faculty of the respective division. The Division Coordinators shall have the responsibility within their respective divisions for scheduling courses, assignment of staff, and the administration of procedures for monitoring the division's programs and students. The Division Coordinators shall report directly to the Department Chairperson, who shall oversee the division programs, provide the divisions operational parameters, and provide the interface with the college administration. Division Co-ordinators shall be given one hour of release time to perform the administrative functions.

A faculty member of the Department of Mathematical Sciences or any other department of the college shall be defined to be a voting member of a particular division if and only if at least one half of his or her teaching schedule is in the division. During the period of reformation the initial assignment of faculty to divisions will be made by the Dean of Arts and Sciences. Once the divisions are constituted the Division Coordinators will recommend all other transfers and/or new-hires with the approval of the Department Chairperson.

Policies and procedures adopted within the divisions of the Department of Mathematical Sciences shall be arrived by a democratic process of decision making.

JERSEY CITY
STATE COLLEGE

MEMORANDUM

TO: Dr. Mary J. Clark, Dean of Arts and Sciences

FROM: George Daneluk, Mathematics Department

DATE: April 25, 1980

SUBJECT: Computer Science Curriculum Proposal

I am submitting revised documentation of the front end problem analysis and preliminary design of the recommended curriculum in computer science. Section I contains the rationale for the curriculum at JCSC; Section II contains a statement of the deficiencies and/or efficiencies (i.e. problems and/or opportunities) the project could be addressing; Section III contains a detailed skills and knowledge analysis (job study) intended to provide the basis for determining curriculum objectives; Section IV contains the preliminary curriculum design. The latter includes the rationale for the four components of the curriculum, which you requested.

Copy to:
Phil Goldstein
Phil Caverley
Carl Holloman
Magda Efros
Harold Carney

JERSEY CITY
STATE COLLEGE

MEMORANDUM

TO: Dr. Mary J. Clark, Dean of Arts and Sciences

FROM: George Daneluk, Mathematics Department

DATE: May 6, 1980

SUBJECT: Computer Science Curriculum Development Project

Enclosed for your information are:

1. A brief outline of the curriculum development standards of the Data Systems Education Center (AT@T),
2. Parts D,E,F, and G of the Skills and Knowledge Analysis (Job Study) contained in Section III of the documentation previously submitted.

cc:
Dr. Phil Goldstein

To: Dr. Mary Jane Clark
From: George Daneluk
Date: June 24, 1980
Subject: A Curriculum in Computer Science

Attached for your information and whatever use is a copy with revisions of the documentation for the Curriculum Computer Science which I originally proposed to the Computer Science Curriculum Development Committee. I have asked Anthony Montanaro, Division Operations Manager, Rutgers University, George Foley, Supervisor Experimental Administration Systems, Bell Telephone Laboratories, and Suzanne Hain, Manager of IS/EDP National Office, Arthur Young, three of the people who met with the committee on campus last summer, to provide an evaluation of the content, structure, and the problem solving approach to the curriculum I have recommended. I will forward for your information copies of these evaluations as I receive them.

The curriculum voted on in committee at the last meeting in June is in my opinion adequate as far providing basic skills in programming and knowledge of operating systems principles. I chose to abstain from voting at the time, however, because of my reservations about the serious weakening of the mathematics component and the present lack of a well-defined component in business information systems development. I also would like to read the written report of Dr. Malcolm Harrison before I take a position. Never the less, as the curriculum stands it would appear to be sufficient to enable students to assume roles in certain areas of technical support. They would require additional training to perform roles in a development team or in operations. Without a substantial background in mathematical foundations students are likely to be handicapped as far as their ability to do graduate

work in computer science.

My own contribution to a program would be strongest in the area of computational mathematics for reasons having to do with style, interest, and competency.

c.c.
Anthony Montanaro
George Foley
Suzanne Hain
Dr. Phillip Goldstein

PROJECT DEMONSTRATING EXCELLENCE

EXECUTIVE SUMMARY

A. Origin of Report

The material contained in this report is documentation for a Curriculum in Computer Science recommended by the author to the Computer Science Curriculum Development Committee at Jersey City State College. The committee was formed in June 1979 to develope a proposal for a Curriculum in Computer Science to be submitted for approval to the N. J. Board of higher Education and, if and when approved, to implement it in the 1980-81 academic year. The committee consisted of two members of the Mathematics Department, of which the author was one, two members of the Center for Computing Services at JCSC, and one member of the Physics Department.

B. Overview of Curricula Development[8,9]

The past 10 years have witnessed an accelerating pace of curricula development in computer science education. Professional societies have been developing model curricula for colleges and universities, but only with much difficulty and heated debate as to what constitutes a cohesive program. Based on recommendations of ACM and the IEEE Computer Society reports and an examination of curricula in existing departments of academic institutions a number of trends can, however, reasonably be projected for at least the first half of the 80's. The first two years of a B. S. degree program will contain a more rigorous and disciplined approach to programming, the object being to enable a firmer scientific base. Results in abstraction, correctness, and verification will be incorporated into advanced courses, and approaches to languages and algorithm analysis will be more systematic. Topics in logic and design will be introduced earlier in many curricula. Because of the impact of mini and micro computers the systems area will be subject to a higher degree of instability. The implementation of these new trends in curricula and the acquisition of sufficient laboratory facilities may be impeded by the unwillingness of institutions and states to allocate the necessary resources. The large and highly visible expense involved with computing equipment has, infact, tended to remove control of facilities from the immediate users to outside agencies. The changes in course content and methodology will require different teaching techniques and possibly retraining of faculties, which changes have traditionally not come easily.

The general curriculum problem is one of incorporating more organization, science, and engineering into the field. Another major problem is that of simulating realistic applications in the program.

Significant curricula development in academic institutions began in the 60's. Prior to that time computer education and training was provided mainly by computer manufacturers. At present a significant source of computer education is being provided by users of computing systems themselves. So at the present time it is not even clear as to who will assume the predominant responsibility for education in the field. The trend may be to share the responsibility.

One of the major reasons for the creation of programs in computer science is to produce graduates who are able to contribute to the economic welfare of their employers. It is clearly an important motivating factor for students undertaking programs. On entry into the field the contribution of graduates is most likely to be, however, at a more or less technical level before moving into middle level management, and still later, if at all, into corporate management. The tendency for "armed camp" technical outposts to exist within organizations would seem to suggest that greater emphasis be placed on learning the complexities of corporate policy making and decision making as well as on technical training. A more general problem in this respect is the disparity in the perception of the needs between business organizations and academic institutions. A cooperative environment needs to be established in which the quality and type of instruction and research activity in both groups is monitored by the other and staff members are exchanged between colleges and universities and the local industrial, business, and governmental communities. The most significant future challenge in computer science is the development of mechanisms for academic - business cooperation, interaction, and exchange. The significance of the present project rests in large part on the benefits derived from the establishment of such a mechanism between Jersey City State College and AT&T Long Lines. It is not as easy as it might appear to be.

Another fundamental issue in the field is that we are still undergoing changes brought about by "the computer revolution" , and there is no prospect of a period of stability in the near future. The most well designed curriculum in terms of the state-of-the-art suffers a fatal defect; it already is or will soon become outdated. Indications suggest that ways must

be found, in fact, to cope with a circumstance of continuous change. Moreover, each institution must work out its own agenda in this context. Somebody, in particular, had to do it for Jersey City State College. If the present project had no other significance it does represent a reasonable plan for JCSC in terms the capabilities likely to be available at the college in the near future.

As already noted there has been a rapid growth in the number of students taking computer science courses, caused mainly by the demand for such knowledge in the market place. At present the need is so great that graduates of almost any type of computer science program have had a relatively easy time finding employment. One must raise a question about the general quality of these programs, however, considering the problems associated with poor systems design, implementation, etc. Moreover, sooner or later the demand and supply curves must cross, so programs should be planned to deal with the inevitable job crunch experienced in other disciplines. At that stage the quality of the graduating student will be the determining factor in employment.

In a systems approach to problem solving an important function is to determine the usage views of the output of the system, and an important part of this is to precisely identify who the users are, if any. In academic curriculum development the user has generally been assumed to be the students, the output being instruction. An alternative model could and, perhaps, should be to consider students as output of the educational system and to view business, industrial, and governmental organizations, who employ the students, as users. An analysis of the problem/opportunity to be resolved by such a curriculum model would then focus on needs of these societal organizations rather than students. The objective of the curriculum model would be to put out students who meet the needs. In the long run this approach has the possibility of being in the best interests of the students as well. In any case an important issue is to identify the population that the educational system is really serving and this is not as trivial a problem as it may seem.

C. Purpose

The purpose of the present curriculum development project was to design a curriculum in computer science adequate to meet challenges caused by the impact of rapid technological change in the field, adequate to bridge the gap between academic capabilities and the real needs of societal organizations, and with clear views as to the problem/opportunity to be resolved. A case is made to support a rational decision making/problem solving approach on different levels of abstraction, a feature which distinguishes the proposed model from the curricula recommended by ACM and IEEE.[27,30]

D. Method

The procedure followed in the design of the curriculum was an adaptation of procedures developed by AT&T for course development. Those procedures and documentation standards are themselves an adaptation of general principles of systems analysis and design applied to course development. The total systems development process can be conveniently broken down into the following phases:[51]

1. Proposal

The objective of the proposal phase is to initiate the project request and identify the problem/opportunity to be resolved. The processes include identifying the problem/opportunity, analyzing the existing system, determining user needs, and identifying general assumptions and constraints.

2. Feasibility

The objective of the feasibility phase is to analyze the existing environment and identify the specific problem/opportunity and to formulate the alternative solutions which have the greatest potential for improving performance. The processes include (a) a review and analysis of proposal documentation, (b) identification of assumptions, constraints, and user needs, (c) determination of initial organization and system objectives, and performance requirements, (d) identification of alternative solutions, (e) description of system output requirements, system data requirements, and system model, (f) evaluation of alternative solutions, (g) selection of the best alternative, (h) establishment of developmental objectives and development plan, and (i) submission of the feasibility report.

3. Definition

The objective of the definition phase is to identify operating policies, information and data flows, to define the functional requirements of the system, and to design the overall system architecture. The processes include (a) a review and analysis of the feasibility report, (b) definition of system boundaries, (c) definition of system outputs, (d) definition of system inputs, (e) definition of system functions, (f) definition of conversion system, (g) definition of control requirements, (h) refinement of system objectives, specifications, and descriptions, (i) reevaluation of feasibility, and (j) submission of definition report.

4. Preliminary Design

The objective of the preliminary design phase is to design the most effective procedures and programs satisfying the requirements detailed in the definition phase and to assign the appropriate functions to appropriate operations for further development. The processes include (a) a review of the definition report, (b) allocation of system functions, (c) detailing the logical requirements for the system functions, (d) design of the logical view of the system data base, (e) detailing internal interface requirements, (f) design of controls and measurement plan, (g) finalizing conversion design, (h) preparation of system test plan, (i) design of fall back, recovery, and reconstruction requirements, (j) finalizing hardware, software, and communication requirements, (k) development of operating environment and operational documentation, (l) determination of training objectives and requirements, (m) refinement of objectives and performance specifications, and (n) submission of preliminary design report.

5. Detail Design

The objective of the detail design phase is to document the operational programs and procedures identified in the preliminary design phase. The processes include (a) design of computer programs/module structure, (b) design of personnel position/task structure, (c) design of physical data base, (d) design of forms, reports, and displays, (e) design of personnel work stations, facilities, and performance aids, (f) definition of fall back status, design of recovery and reconstruction procedures, and ensurance of reliability, (g) preparation of program test plan and data, (h) preparation of position test plan and data, (i) preparation of user operations training plan, (j) design of conversion programs, (k) refinement of implementation and conversion plans, (l) reevaluation of feasibility, and (m) submission of detail design report.

6. Implementation

The objective of the implementation phase is to build operational programs and procedures to meet the specifications of the preliminary and detail design phases. The processes include (a) construction of the components, (b) preparation of component test data and environment,

(c) testing of components, (d) combining components into subsystems, (e) testing the subsystems (verification), (f) developing training courses and administrative guides, (g) running of pilot courses, (h) training of personnel, (i) combining components into a total system, (j) performing validation tests, (k) preparing operational documentation, and (l) submission of implementation report.

7. Conversion

The objective of the conversion phase is to completely install the new system and phase out procedures, documents, reports, and assignments no longer required. Processes include (a) performance of preinstallation activities, (b) coordination of hardware installation, (c) conversion of data, (d) testing of new hardware, (e) performance of operational trial, (f) resolution of differences in the system, (g) phasing out of old system procedures, etc., (h) performance of certification test, and (i) submission of completion report.

8. Performance Review

The objective of the performance review phase is to determine whether the system meets the system objectives and requirements outlined in the feasibility and definition phases, to insure that all aspects of the project are satisfactorily completed, to certify that the defined business objectives are achieved, and to recommend the systems disposition. The processes include (a) review of the design report, (b) review of the operating system, (c) review of the operating group and user attitudes toward the system, (d) evaluation of the operating system, (e) evaluation of the system development effort, (f) review of feasibility studies, (g) evaluation of the impact on the organizations goals, (h) evaluation of the economics, and (j) submission of the performance review report.

The corresponding phases of the process as applied to course development are:

1. Preproject Study and Job Study Planning

2. Job Study with Implications for Training

3. Training Objectives, Criterion Test Development, and Training Design

4. Materials Development and Plans for Field Test

5. Field Test and Plans for Introduction

6. Training Introduction and Plans for Follow-up Evaluation

7. Follow-up Evaluation and Recommendations for Remedial Action

8. Remedial Action

The activities and documentation standards are precisely represented by procedural flow charts and exhibits developed by the AT&T Training Research Group. The present project involved an application of those standards for course development to a total curriculum in an academic environment. The present project also involved an application of the procedures to problem solving/rational decision making skills, which are not readily quantifiable in behavioral terms. Although the developmental process carried out was fairly thorough with respect to the "front end" analysis and design phases, the documentation follows standards for program approval of the N. J. Board of Higher Education and the JCSC Senate Curriculum and Instruction Committee.

To obtain a view of the systems development process in a business environment and data for the phase analyses, observations were conducted of courses, designed to enable personnel to perform roles in the systems development process, at the Systems Technical Education Center of AT&T Long Lines in Piscataway, N. J. The observations were carried out over a four month period in the summer of 1978 in connection with an internship associated with a doctoral program with Union Graduate School undertaken by the author. Twelve courses in the Systems Development Curriculum and the Advanced Programmer Training Curriculum were observed. In addition, student materials, instructor guides, and administrative guides were reviewed for subject matter content and methodology. Also both formal and informal interviews and discussions with instructors, course developers, standards developers, curriculum managers, administrative staff, outside consultants, and students were conducted.

In order that the curriculum model based on the observations not be put forward merely as an academic exercise, the information and ideas were presented as recommendations for the actual curriculum being developed at JCSC. Since there are no absolutes with respect to the validity of something as complex as a four year academic curriculum, the best test is to subject the recommendations to review and evaluation by "experts" in the field. The validity of

the documentation derives from the fact that it is based in part on expert opinion to begin with and the fact that it underwent such critical examination by members of the Curriculum Development Committee, and outside consultants brought in to review the proposal.

E. Results

1. Phase 1: Preproject Study

The preproject study identified the problem/opportunity for instructional resolution to be skills and knowledge deficiencies in the systems approach to problem solving/rational decision making. The assumption was that the deficiencies in problem solving skills could be ensured by adequate training and selection of personnel. The basis for the assumption is the observation that problem solving methodology constitutes the unifying idea binding the courses taught in a particular but typical business environment.

2. Phase 2: Skills and Knowledge Analysis

The skills and knowledge analysis identified a comprehensive set of functional roles grouped in three categories, namely, in a development team, in operations and maintenance, and in technical support. The roles are precisely those filled by systems generalists to perform the functions required in the phases of the total systems development life cycle. In a development team skill and knowledge are required for (a) project management including application expertise, systems analysis and design, data base administration, personnel subsystem design, position development, training development testing, and data communications design, (b) data center functions on the development team including project leadership, systems design, computer subsystem design, data base management, and programming, (c) audit functions on the development team. In operations and maintenance skills and knowledge are required for (a) system management including application expertise, training administration, course instruction, personnel subsystem maintenance, system use, position supervision, and position operation, (b) data center functions including data center management, computer center maintenance, computer center operation, data base management, computer center technical support, network control, and computer subsystem maintenance.

In technical support skills and knowledge are required for (a) departmental functions including departmental data administration, application planning, and user approval, (b) data services functions including authorizations, standards, and data administration, (c) time share center support, (d) communications support connected with the internal data network, (e) systems technical education including data systems training, and (f) planning and telecommunications. Amplifications of the systems analysis and design and programmer functions indicate the need for skill and knowledge required for studying and analyzing systems for the purpose of identifying problem/opportunity areas for improvement and the design of systems which resolve the problem/opportunity.

3. Phase 3: Curriculum Design

From the data collected on functional roles a case is be made for a curriculum designed to provide students with skills and knowledge required in total systems development, in operations and maintenance of computer systems, and in technical support, i.e. in the systems approach to problem solving. A curriculum should enable students to fill precisely those functional roles identified in one to four years after being employed. An important characteristic is that students be educated as generalists, i.e. in ways to enable them to serve in different roles at different phases of the development process. In the present project a design is put forward consisting of four components, namely, Business Information Systems Development, Hardware Organization, and Software Engineering, and Mathematical Foundations, The unifying idea binding the courses in each component is the problem solving process viewed on different levels of abstraction. Course guides specifying enabling objectives and an outline of topics is given in Appendix 1. The course guides constitute the guts of the program.

F. Summary and Conclusion

1. "The slowness of one section of the world about adopting the valuable ideas of another section of it is a curious thing and unaccountable. This form of stupidity is confined to no community, to no nation; it is universal. The fact is the human race is not only slow about borrowing valuable ideas, —it sometimes persists in not borrowing them at all"
Mark Twain, Europe and Elsewhere, p. 175

REFERENCES

1. AT&T Training Research Group, Training Development Standards Reference Manual I, (Interim Revision) March 1975.

2. JCSC Middle States Accreditation Committee, Comprehensive Self-Study Report to the Commission on Higher Education of the Middle States Association of Colleges and Schools, September 1979, V.1, p. 11.

3. Dr. Glenn Reeling and Psychology Club, Student Attitudinal Questionaire, (Jersey City State College, April 1979).

4. John W. Hamblen, Computer Manpower - Supply and Demand - by State, (3rd Ed.),(St. James, Missouri: Information Systems Consultants, 1979).

5. JCSC Placement Survey, BA and BS Degrees, (1976, 1977, 1978, 1979 Ed.) (JCSC Office of Academic Career Planning and Placement, Jersey City, NJ).

6. Suzanne Hain, Systems Development Curriculum, (AT&T Long Lines Systems Technical Education Center, Piscataway, NJ, July 1979).

7. Dave Rine, "Workshop Report: Computer Science, Engineering and Data Processing Curricula", (COMPUTER, May 1978) p. 82.

8. Udo W. Pooch, Richard H. Austing, Rahul Cattergy, Michael C. Mulder, "Computer Science and Computer Engineering Education in the 80's", (COMPUTER, Vol. II, No. 9, Sept. 1978) p. 69.

9. Anthony I. Wasserman, "Computer Science and Computer Engineering Education in the 80's", (COMPUTER, Vol. II No. 9, Sept. 1978), p. 30.

10. Richard E. Fairley, "MSE 79: First Draft of a Masters Curriculum in Software Engineering", (ACM SIG. SOFT, Software Engineering Notes, Vol. IV, No. 1, Jan. 1979), p. 12.

11. A Curriculum in Computer Science Committee Report, "Software Engineering Subject Area ", (IEEE, EHO, 119-8, Jan. 1977), p. 42.

12. Barstow Hodge, Robert Fleck, "Producing realtime graduates: A "five case" approach to information systems education", (Data Management, Oct. 1977), p. 60.

13. A. A. Lopez, Robert Raymond, Robert Tardiff, "A Survey of Computer Science Offerings in Small Liberal Arts Colleges", (CACM, Vol. 20, No. 12, Dec. 1977), p. 902.

14. Peter Freeman, Anthony I. Wasserman, Richard E. Fairley, "Essential Elements of Software Engineering Education", (Proceedings of the ACM, Vol. No. 1977), p. 116.

15. Fred C. Homeyer, "Why Industry Hires Our Graduates", (Proceedings of the ACM, Vol. No. 1977), p. 264.

16. Michael L. Dertouzos, Michael Athans, Richard N. Spann, Samuel J. Mason, "Networks, Systems, and Computation", (Course presented by the COSINE Committee-Commission on Engineering Education: Computer Science in Electrical Engineering, Princeton University, June 1968).

17. AT&T Training Research Group, "Training Development Standards, Phase I, (Issue 2, AT&T Long Lines, Bedminster, NJ, June 1978).

18. AT&T Training Research Group, "Training Development Standards, Phase II", (Issue 2, AT&T Long Lines, Bedminster, NJ, December 1978).

19. AT&T Training Research Group, "Training Development Standards, Phase III", (Issue 2, Preliminary Draft, AT&T Long Lines, Bedminister, NJ, July 1978).

20. Marvin V. Zelkowitz, Alan C. Shaw, John D. Gannon, Principles of Software Engineering and Design, (Prentice-Hall, Inc., Englewood Cliffs, NJ, 1979).

21. Fredrick J. Hill, Gerald R. Peterson, Digital Systems: Hardware Organization and Design, (John Wiley & Sons, Inc., New York, 1973).

22. Louis Padulo, Michael A. Arbib, System Theory: A Unified State-Space Approach to Continuous and Discrete Systems, (W. B. Saunders Co., Philadelphia, PA, 1974).

23. Anthony Ralson, Mary Shaw, "Curriculum '78 - Is Computer Science Really That Unmathematical?", (CACM, Vol. 23, No. 2, Feb. 1980), p. 67.

24. Committee on the Undergraduate Program in Mathematics, Applied Mathematics in the Undergraduate Curriculum, (Mathematical Association of America, Berkeley, CA, Jan. 1972).

25. Committee on the Undergraduate Program in Mathematics, Recommendations for an Undergraduate Program in Computational Mathematics, (A Report of the Panel on Computing, Mathematical Assiciation of America, Berkeley, CA, May 1971).

26. Committee on the Undergraduate Program in Mathematics, Recommendations on Undergraduate Mathematics Course Involving Computing, (Mathematical Association of America, Berkeley, CA, October 1972).

27. IEEE Computer Society Model Curricula Sub-Committee, A Curriculum in Computer Science and Engineering - Committee Report, (Rev. 1, IEEE Computer Society, Long Beach, CA, January, 1977).

28. ACM Curriculum Committe on Computer Science, "Curriculum '68, Recommendations for Academic Programs in Computer Science", (CACM, Vol. 11, No. 3, March 1968), pp. 229-235.

29. Thomas H. Athey, Jerry Wagner, "Preliminary model curriculum for business systems emerges, addresses "real world" needs", (Data Management, April 1980), p. 42.

30. Richard H. Austing, Bruce H. Barnes, Della T. Bonnette, Gerald L. Engel, Gordon Stokes, A Report of the ACM Curriculum Committee on Computer Science: Curriculum '78, Recommendation for the Undergraduate Program in Computer Science, (Communications of the ACM, Vol. 22, no. 3, March 1979).

31. Total Systems Development Concepts, Developed by the Systems Development Curriculum, AT@T Long Lines, Piscataway, N. J., 1979