I. INTRODUCTION TO THE CURRICULUM

The undergraduate program in computer science is oriented to:

(i) majors in computer science,

(ii) students majoring in other areas (mainly the sciences, mathematics, engineering, and business) who are interested in a working knowledge of computer problem solving, and

(iii) students interested in a general introduction to computers, their range of application, and their impact on society.

A major in computer science is expected to acquire the knowledge and skills needed to hold professional positions in areas of design, implementation and operation of computer systems (hardware and software) and of applications software; to teach computer science topics at the high school and college levels; and to prepare him for further graduate study in computer science. Courses that cover prerequisites for graduate study in computer science at Rutgers are available in the undergraduate program.

Because of the rapid rate of growth in the field, a student majoring in computer science must acquire the capability to follow the research and professional literature, and to adapt to changes in methods, languages and systems. Furthermore, he must develop the ability to plan and work independently on substantial computer projects, and to communicate effectively with other people in the course of his work. As the impact of computers on society is becoming more significant from day to day, the student must prepare himself to participate intelligently in managerial and social decisions that involve computers. In view of these requirements, our undergraduate program in computer science emphasizes concepts, theories and general methods which underlie applications of computers and which can be used to guide the design of computer systems. The program also stresses independent study, doing active work on computers with specific languages, and participating in advanced seminars.

The structure of the undergraduate curriculum in computer science is summarized diagrammatically in Figure 1, which appears at the end of this section. Boxes stand for courses, and lines denote relations between courses and their prerequisites. Courses with odd numbers are given in the Fall, and even-numbered courses are given in the Spring. Students may plan their overall program of study with the help of this diagram.

The location of a course in the diagram roughly indicates its level and type. Introductory courses are at the top of the diagram, and more advanced courses appear lower in the diagram. In our program there are four broad types of courses: mathematical foundations and theory; computer programming, and system software; applications of computers to numerical problems; and applications to non-numerical problems. In the diagram, courses of the same type appear in a vertical arrangement. The course types that correspond to these arrangements are shown in the top of the diagram. Not each
course can be accurately classified within the four types; the classification should be taken as a rough guide for planning a program of study. More detailed information on courses and prerequisites are given in the course descriptions of the next section.

A student majoring in computer science is expected to take at least fourteen semester courses in the program (the courses 090:119-120 cannot be used towards this count); eleven of the fourteen are required courses, and the remaining three can be chosen in accordance with the students specific interests and orientation. In Figure 1 the required courses are shown with darker boxes. Four of them (105, 106, 205, and 206 shown in a column at the left) provide the necessary mathematical concepts and methods for work in computer science; students with insufficient previous mathematical preparation are required to take the introductory two-semester course 090:119-120 before taking any of these four courses. Three of the required courses (111, 211, and 212 shown at the upper right of Figure 1) are concerned with the fundamentals of computer problem solving and with programming in high level languages (BASIC, FORTRAN, PL/1, SNOBOL, LISP); Four of the required courses (311, 312, 401, and 402 shown in the lower middle of Figure 1) provide the basic concepts and techniques in computer organization, machine language programming, and systems software. The remaining three courses that complete the minimal program for the computer science major may be chosen from offerings in advanced computer applications (321, 322, 419, and 432 shown in the lower right of Figure 1), social implications (431 shown at the extreme bottom right), and computer theory (422 shown at the extreme bottom left). Our present program enables a student to choose up to six courses in computer science beyond those that are minimally needed for a major. Independent Study (223, 224, 495, and 496) provides an additional opportunity for the student interested in topics and experience not covered in the formal courses. Students will be strongly urged to devote at least one course to independent study or to a seminar. Courses relevant to computer science are also available in the departments of Mathematics (RC, DC), Philosophy (LC), Industrial Engineering (RC), Statistics (RC), and Electrical Engineering (RC).

A student majoring in computer science would be encouraged to develop an overall study plan (in consultation with his advisor) sometime during his first two years in college. The plan would include the student's course requirements for major; courses in computer science beyond the requirements for major, as well as courses in other departments that are relevant to computer science, all chosen so as to enhance the student's professional educational goals; and courses in other areas in accordance with the student's general interests.

Students majoring in the physical and biological sciences, as well as in mathematics and engineering, will gain a good grasp of computer concepts and a strong competence in the use of computers in their respective fields by taking a sequence of two courses in numerical methods and in computer problem solving (211 followed by 322). Alternatively, students in these fields may gain a knowledge of computers, and programming which is less extensive, but sufficient for many of their needs, by taking the introductory course 112.
Students in the social and behavioral sciences, in business, and in the humanities, will receive a strong preparation in the use of computers in their work by taking a sequence of two courses in nonnumerical methods and computer problem solving (212 followed by 321 or 432). A less demanding alternative would be to take the introductory course 112.

Students interested in a general acquaintance with computers and their uses will find the introductory course 112 sufficient for their needs.

Those interested in studying the broad impact of computers in various aspects of scientific, economic and social life should participate in the seminar course 431 after taking at least one computer science course (such as 112).

Even though the introductory mathematical course (105, 106, 205, 206) are specifically designed for computer science majors, their emphasis on applied problems and on constructive processes, and their articulation with courses in computer problem solving, make them well suited for students majoring in certain areas of biology, psychology, anthropology, economics and urban planning.

Notes concerning Figure 1 (which appears on following page):

\[ \rightarrow \] points to a course from its prerequisite(s).

\& means that all entering arrows are required.

\& V means that any entering arrow is acceptable.

M1 satisfactory high school mathematical preparation.

M2 one year of college level mathematics.

M3 intermediate logic at the level of Philosophy 730:332 or consent of instructor.

M4 advanced high school mathematical preparation.

E engineering sophomores required to take the course.

B1 at least one computer science course and consent of the instructor.

B2 consent of the instructor.

B3 no special mathematical background required, knowledge of high school algebra is desirable.

\[ \Delta \] the course is not offered in 1971-72.

A,B course 212 has two sections, see course description.

Dark boxes indicate the eleven required courses.
Figure 1. Undergraduate Program in Computer Science
II. THE COURSES

This section presents a description of each course in the undergraduate program for the year 1971-72. Courses with odd numbers are given during the fall semester and even numbered courses are given in the spring (except for the course 103 which is given both in the fall and spring semester of 1971-72). Each course description specifies the academic objectives of the course (also, for what students it is intended); the preparation expected of a student taking the course (prerequisites and other required knowledge or experience); the work expected of a student in the course (assignments, examinations, computer projects); an outline of the course; and references (textbooks and articles) that provide a further basis for identifying the subject matter and level of the course.

Prerequisites are to be taken as strong guidelines that should help a student to decide whether to register in a specific course. Normally, students taking a course should satisfy the course prerequisites. However, special cases will be considered individually by the instructor in charge of the course.
INTRODUCTORY APPLIED MATHEMATICS
(to be given in the Fall and Spring)

Instructor: H. Eng & Staff

Objectives: This course is intended for students with insufficient high school mathematical preparation who are interested in pursuing studies in areas that require an applied mathematical background.

Prerequisite:
Section 1 - none.
Section 2 - one year of high school algebra

Outline: This course is designed to acquaint students with principles, methods, and procedures for solving problems in elementary applied mathematics. Topics include numbers and sets, variables and open sentences, addition and multiplication of real numbers, solving equations and problems, solving inequalities, operations with polynomials, special products and factoring, operations with fractions, functions, relations and graphs, systems of open sentences in two variables, rational and irrational numbers, quadratic equations and matrices.

Expected Work: Weekly homework assignments and at least four quizzes, plus midterm and final.

References:

April 1971
090:120  INTRODUCTORY APPLIED MATHEMATICS  (4 credits each term)
(To be given in the Fall and Spring)

Instructor:  M. Eng & staff

Objectives:
Section 1 is intended for students with insufficient high school mathematical preparation who are interested in pursuing studies in the behavioral science areas that require an applied mathematical background.

Section 2 is intended for students with insufficient high school mathematical preparation who are interested in pursuing studies toward a computer science major or in other scientific areas that require an applied mathematical background.

Prerequisites:  Two years of high school mathematics or 090:119.

Outline:
Section 1: This course is designed to acquaint students with principles, methods and procedures for solving problems in elementary applied mathematics. Topics include sets, linear and polygraphs, linear programming, mathematics of gambling, matrices, non-linear algebraic functions, log functions, sequences and series, mathematics of derivatives and differentiation of algebraic functions. Additional topics are fundamentals of probability and introduction to statistical methods in experimental science.

Section 2: This course is designed to acquaint students with principles, methods and procedures for solving problems in elementary applied mathematics. Topics include sets, inequalities, functions and relations, vectors and matrices, algebraic functions exponential and logarithmic functions, trigonometric functions, analytic geometry, sequences, series, intuitive differentiation, and intuitive integration.

Expected Work:  Weekly homework assignments and at least four quizzes, plus midterm and final.

References:
Freund, John E., STATISTICS A FIRST COURSE, Prentice Hall, 1970


April 1971
198:103 BASIC COMPUTER PROGRAMMING (1 credit each term)

(This course will be given for the last time in 1971-72.)

Instructor: S. Droege

Objectives: Intended exclusively for engineering sophomores. It is a course in FORTRAN programming. The material of this course, much strengthened by a general treatment of high level algebraic languages and numerical problem solving, is included in the course 198:211 (Numerical Problems and Computer Programming). Engineering students who are seriously interested in this subject are advised to take the course 198:211 (instead of 198:103).

Prerequisites: Sophomore engineer status.

Outline: The use of the computer language FORTRAN IV in formulating and solving numerical problems on a computer. Introduction to algorithms, programs, and computers. Debugging and verification of programs.

Expected Work: Students will write and debug a number of programs. Laboratory sessions will meet weekly.

193:105-106  INTRODUCTORY COMPUTER ORIENTED MATHEMATICS  (4, 4 credits)

Instructor:  David R. Beaucage

Objectives:
This course is intended to give computer and other science majors a basic facility in the manipulation of the tools of calculus and linear algebra: derivatives, integrals, and linear maps. The course is also open to non-science students. A major in computer science is required to take this course or, alternatively, the two-semester calculus sequence and the course in linear algebra offered by the Mathematics Department (MC). The course provides useful mathematical foundations for work in computer science.

Students are recommended to take 193:111 during the Fall semester concurrently with 193:105 since the computer is used in 105 and extensively in 106 as an aid for teaching mathematical ideas.

Prerequisites:  Grades of at least B (2) in high school algebra, geometry, and intermediate algebra; passing a placement test; or 090:119-120. The ability to add fractions, plug values into formulas, and draw graphs by plotting points must not be in doubt. Trigonometry and analytic geometry are not prerequisites but will probably help. Those who have successfully passed a year of college calculus may not receive credit for 193:105-6 and should take a semester of linear algebra instead. Priority will be given to computer science majors.

Outline:
The computer programs that students study and write in this course help deepen and extend their understanding of mathematical concepts and procedures. The programming language used is BASIC, which is easy to learn and was designed for use in education.

FALL SEMESTER:  Algorithms and BASIC procedures; functions and mapping diagrams; complex arithmetic and geometry; squaring as a mapping of the plane; local differential behavior of squaring; the derivative; linear maps and $\mathbb{R}^n$; examples: multiplication by a constant, summing, evaluation at a point, differentiation at a point; linear equations; matrices; matrix multiplication; the derivative as a matrix; the Cauchy-Riemann equations; analytic functions; the chain rule; differentiation of polynomials; linear independence; graphs; partial derivatives; slope; maxima and minima; vector derivatives; dot products; open sets; sequential limits; exp; the derivative of exp; exp as a mapping of the plane; $\sin$ & $\cos$ and their derivatives; dot-products with $\cos$; correlation coefficients; derivative of a product; rates; tangent lines and planes.
SPRING SEMESTER: Iterative processes; Newton's method; roots and inverses; inverse matrix by Gaussian elimination; derivative of the inverse function; log, square root, arc-sine, arccos, and their derivatives; derivative of a quotient; tan & arctan; graphs; higher derivatives; critical point tests; dot products and projections; work; contour integration; existence of logs; area: the definite integral; the fundamental theorem of calculus; integration by parts; continuous functions; Rolle's theorem; the mean-value theorem; proof of the fundamental theorem; Simpson's rule; independence of path; determinants; change-of-variable formula for integration; the inverse function theorem; rank; kernel; direct sum; series and convergence; differentiation and integration term by term; Taylor's theorem with integral remainder; computer approximations to transcendental functions; error bounds; projection onto a subspace; Fourier series; cyclic subspaces; eigenvectors & eigenvalues; canonical forms.

Expected Work: One 75-minute lecture per week; 125 minutes of discussion periods per week. Midterm and final examinations. Problem sets. Frequent short quizzes. Homework assignments will often require use of the computer.

References:


Movies: A Function is a Mapping, I Maximize, The Theorem of the Mean, Continuity of Mappings, Newton's Method, all in the MAA Calculus Films series, dist. by Modern Learning Aids, Summit, N. J.
INSTRUCTOR TO COMPUTING (4,4 credits)

Instructor: J. Cox

Objectives: This course is required for computer science majors and is also suitable for other students who are interested in learning fundamentals of computing. It will provide the computer science major with the necessary background for more advanced computer science courses. Courses 111 and 112 cover essentially the same material. Computer science majors are advised to take 111 in the fall since the computer is used extensively in 106 as an aid for teaching mathematical ideas. Non-majors are strongly advised to register for 112 in the spring.

Prerequisites: Satisfactory high school mathematical preparation or 090:119-120.

Outline: Algorithms
Interactive programming using the BASIC language
Programming numerical problems (e.g., solution of systems of linear algebraic equations).
Programming non-numerical problems (e.g., sorting and generating a menu by computer.)
Block diagram of a computer
Computer simulation
Other programming languages.

Expected Work: Students are assigned ten lab programs. The programs are designed to give the student a background in various aspects of what a computer can and cannot do. These programs are written in BASIC, a simple language, which is taught during the course. The programs are written using the Time Sharing System (TSS). Each program takes about one hour of terminal time plus about two hours of outside preparation. There will be a midterm and a final examination.


April 1971
INTRODUCTION TO DISCRETE STRUCTURES

Instructor: R. Orgass

Objective: This course is required for computer science majors. It is also recommended for students in the biological, behavioral, and social sciences, as well as in urban planning who have a theoretical orientation and are interested in computer methods. The course is designed to introduce students to various topics in mathematics which are useful when considering practical problems in computer science. The main emphasis of the course will be on topics in mathematics, but some attention will be devoted to applications.

Prerequisite: Two semester courses in mathematics at the College level (e.g., 183:105-106 or equivalents) or advanced high school mathematical preparation. The important prerequisite is the ability to work with mathematical concepts.

Outline: The course introduces the student to those fundamental algebraic and logical concepts that are needed for theoretical and advanced practical work in computer science and illustrates applications of these concepts to problems in the field. Topics include set theory, propositional calculus, boolean algebra, algebraic structures including semigroups and groups. Relationships between these structures will be presented, and applications to the study of computer hardware and to algorithms will be discussed. Finally, there will be a study of the first-order predicate calculus at an introductory level.

Expected Work: There will be a number of class assignments, a mid-term and a final examination.

References: With fairly high probability, the text for the course will be:


Additional References:

Thomason, R. H., SYMBOLIC LOGIC, Macmillan, 1970


Fitch, F. B., SYMBOLIC LOGIC, Ronald, 1952.

April 1971
198:206 INTRODUCTION TO COMBINATORIAL ANALYSIS Spring (4 credits)

Instructor: C. V. Srinivasan

Objectives: This course is required for majors in Computer Science. It is also recommended to students in the sciences and in urban planning who have a theoretical orientation and are interested in computer methods for solving problems in enumerative analysis, theory of graphs, optimization, and design of experiments. The course includes a brief introduction to discrete probability theory.

Prerequisites: 198:206 or equivalent and 198:105,106 or equivalent. No prior knowledge of combinatorial mathematics is assumed. Concepts in modern algebra and probability theory will be introduced in class as necessary. Knowledge of programming will help, but is not essential. Students who like programming would benefit by taking 158:212 in parallel with this course.

Outline: The following topics will be covered in this course: Permutations and Combinations, elementary concepts in discrete probability theory, Generating Functions, Recurrence Relations, the Principle of Inclusion and Exclusion, Fundamental concepts in the Theory of Graphs, Transport Networks, Linear Programming, and Block Designs. In each area the basic concepts will be introduced through examples, and associated algorithmic techniques will be discussed. Interested students will be encouraged to write programs implementing the algorithms.

Expected Work: There will be weekly assignments of homework. There will be two examinations, mid-term and final. Knowledge of programming not required but desirable.


April 1971
Instructor: F. G. Fender

Objectives: The course concentrates on the study of high level programming languages and their use in the formulation of algorithms for the solution of numerical problems. Required of computer science majors. It is also recommended for students majoring in the sciences in mathematics and engineering who are seriously interested in computer programming and numerical methods.

Prerequisites: A working knowledge of calculus and linear algebra such as 198:105-106.


Expected Work: Approximately 12 computer problems will be assigned in FORTRAN IV. A few introductory problems in PL/I. A mid-term and a final examination.


IBM Manual 360 - FORTRAN IV LANGUAGE, Form No. C28-6515.


April 1971
Instructor: M. C. Paull

Objectives: To explore the application of computer languages to the formulation of algorithms for the solution of non-numerical problems.

Prerequisites: This course will be given in two forms: One, Section A for those with some programming background including experience with at least one higher level language. This requirement is satisfied by courses 111 or 211 or both. The other, Section B will require no computer background, but a satisfactory high school mathematics background or 090:119-120 is assumed. Section A of this course is required for majors in computer science.

Outline: In section A SNOBOL 4 will be introduced. Because of the background expected in this section, students will be expected to learn the language rapidly.

Section B will start with an introduction to programming in higher level languages using PL/I as the major language. A brief introduction to SNOBOL 4 is also planned. Beyond this introduction, the same type of material will be covered in both sections.

The course will be built around the study of a series of non-numerical problem types. The formulation of algorithms and their programs for problems including language manipulation (such as text editing), sorting, symbolic simplification and expansion of symbolic mathematical expressions, symbolic differentiation, language analysis and translation. Related problems include generations of samples of a language from its grammar and enumerations of algebraic expressions of a certain kind. Another class of problems are graph manipulations, including shortest path problems such as the traveling salesman problem. Also wiring, layout, and maze problems, graphics (synthesis and analysis of pictorial displays such as charts, cartoons and mechanical drawings), game playing, and theorem proving problems. The common techniques applicable to these problems will be emphasized.

Expected Work: There will be about six computer problems assigned. There will be a mid-term and a final examination.

References: Reference 1 is required for section A. Reference 2 for section B and Reference 3 is a good general reference; it is not required.

198:223-224 INDEPENDENT STUDY A (Variable credits)

Instructor: Staff (by arrangement)

Objectives: This course is intended for computer science majors who are interested in going deeply into a subject which is not covered comprehensively in a formal course. This course also provides a vehicle by which computer science majors may gain experience in software design and implementation by undertaking a substantial programming project.

Prerequisites: Consent of the instructor (students should obtain and complete an application form at the department office prior to registration for independent study).

Outline: To be arranged.

Expected Work: Details to be arranged with supervising instructor. A number of suggestions for practical projects will be supplied by the computer center, CCIS.

References:

April 1971
Instructors: Staff

Objectives: This course concentrates on the structure of computer systems and on fundamental programming concepts and techniques. It is required for computer science majors. It is also recommended for students majoring in mathematics and engineering who are interested in computer organization, programming schemes and systems software. The first semester is devoted to computer organization and programming techniques; the second introduces system software.

Prerequisites: A course in computer problem solving (198:211, but 198:212 or 198:111 will also do) and one year of college mathematics.

Outline: Material to be covered in the first semester (311):

1. Organization of a simple computer (e.g., MIX) functional units, registers, memory organization, instruction formats; data representation, addressing techniques, machine language programming.
2. Elementary Logical Design—construction of registers, half-adders, decoders, etc., from simple digital building blocks.
3. IBM System 360 organization and machine language.
4. Symbolic Assembly Language.
5. Classical Data Structures—arrays, lists queues, stacks, strings, and trees.
6. Selected Programming Techniques—searching, sorting, string manipulation, syntactic recognition, etc.
7. Program Segmentation and Linkages, subroutines, coroutines, and functions.

Material to be covered in the second semester (312):

1. Assembly language macro facilities.
2. Operation and Organization of assemblers, loaders and linkage editors.
3. I/O devices and interrupt handling.
4. Elementary discussion of operating systems batch processing, multi-programming, time-sharing.
5. Interpretive routines and simulation.

Expected Work: In the first semester there will be about six computer problems assigned at two-week intervals. There will be mid-term and final exams. Short class quizzes may also be given at appropriate points. Programming will be entirely in S/360 assembly language.
Expected Work: (continued) In the second semester there will be about four computer problems assigned at two-week intervals. There will also be a required term (programming) project. All programming will be done in assembly language. There will be a mid-term and final examination.


198:321  DATA PROCESSING METHODS         Fall         (4 credits)

Instructor:  S. Baxendale

Objectives:  Recommended for majors in computer science, as well as for
students in social sciences, humanities and business who
are interested in applications of computers in data process-
ing and in information systems.

The course will treat the data processing requirements of
information and business oriented computer systems.

Prerequisites:  198:212. A knowledge of PL/1 programming will be assumed.
The COBOL language will be used extensively. Because of the
programming background assumed, students will be expected
to learn the language rapidly on their own with some
guidance from the instructor. Students in doubt of their
preparation should consult the instructor prior to registration.

Outline:  Description of a data base and its structure. Concepts of
functions, arrays, records, files, trees, lists and list
structures.

Fixed length, variable length and mixed records. Updating
and addition to records. Absolute and relative referencing
of a record. Linkages and transformation vectors and matrices.
Series and parallel files; hierarchies of storage.
Sorting, searching and retrieval from files; the role of
programs and the data base. Their relocation and allocation
of storage. Text processing, Transformations from one storage
medium to another.

Business record keeping: Case study inventory control. Manage-
ment Information Systems, Enquiry and interactive systems.
Personnel records, Air reservations, Automatic Indexing and
Information Retrieval System simulation and control.

Properties of languages useful in description of simulation
models. Design of models including selection of scope,
identification of exogenous and endogenous events, entities,
establishment of facilities and queues, synchronization and
collection of information. Development and testing of models
including queueing models, feedback systems, storage systems,
priority systems and flow in complex networks.

Expected Work:  COBOL programming and file processing. Homework assignments.
Mid-term and final examinations. Large project in record-
keeping using modular programming approach.


April 1971
Instructor: S. Baxendale

Objectives: Recommended for majors in computer science, as well as students in physical and biological sciences, mathematics, and engineering who are interested in applications of computers to numerical problems.

The orientation of this course is toward the use of numerical methods on computers. Some emphasis will also be placed on graphic display of output data.

Prerequisites: 198:211


Expected Work: Use of Fortran IV and PL/I programming. Class assignments and individual projects. Use of a digital plotter and time sharing terminal, mid-term and final examinations. Grade assessment is dependent on satisfactory completion of programming assignments.


April 1971
198:401  LANGUAGE SOFTWARE  Fall  (4 credits)

Instructors:  Staff

Objectives:  This course concentrates on the structure, formal properties, and software implementation of programming languages. It is required for computer science majors. It is also recommended for other students interested in computer languages and their manipulation.

Prerequisites:  198:311-312 (or 198:301-302), and 198:205 (or equivalent); also experience with one or more higher level languages (e.g., Fortran).

Outline:  Material to be (un)covered:

1. ASSEMBLERS & LOADERS: One and two pass assemblers, symbol table organization and searching methods, relocation and linking of subroutines.

2. COMPILERS: Review and comparison of various high-level languages (PL/I, Fortran, Algol), language specification (BNF), top-down and bottom up parsing methods, code generation and optimization, block structure and run time stacks, recursion.

3. MACROS: As extensions to assembly language and as a string processing functions.

4. LIST PROCESSING: Advantages, problems, and techniques for manipulating linked structures, higher level list processing languages.

Expected Work:  The orientation will be primarily classroom work and perhaps a few small computer problem assignments. There will be a mid-term and a final examination. A term project involving the implementation of an assembler will be enjoyed by all.


April 1971
Instructors: Staff

Objectives: This course concentrates on the structure and implementation of operating systems software. Rather than present detailed descriptions of any one specific system, the course will focus on the basic facilities required to construct such systems in general. The course is required for computer science majors. It is also recommended for other students interested in the overall organization of computer operations.

Prerequisites: 198:311-312 (or 198:301-302) and 198:205,206 (or equivalent). It is assumed that the student has at least a passing familiarity with the external characteristics of one operating system.

Outline: Material to be (un)covered:

1. BATCH PROCESSING: Evolution of operating systems, loading, execution, overlap of I/O and computation, buffering, interrupt processing.

2. MULTIPROGRAMMING & MULTIPROCESSING: Time sharing, dynamic memory allocation, scheduling, protection, segmentation, virtual memory, addressing schemes, parallel processing, system modeling and simulation.

3. FILE ORGANIZATION & MANAGEMENT: Structure and accessing of external files, privacy and sharing of data.

Expected Work: Since the course covers many topics, its orientation will be primarily to the classroom with small computer problem assignments, but no term project. There will be a midterm and a final examination.


198:419 NUMERICAL METHODS IN ORDINARY DIFFERENTIAL EQUATIONS
Fall (3 credits)
(This course will not be offered in 1971-72).

Instructor: F. G. Fender

Objectives: For science and engineering majors who may need to solve
differential equations by computer and others curious about
how a computer can give a solution to a problem in analysis.

Prerequisites: 198:211

Outline: Finite Differences, Solution of Finite Difference Problems,
Approximation of Differential Equations by Difference
Equations, Methods of Euler, Milne-Simpson, Adams-
Bashforth-Moulton, and Runge Kutta. Unusually difficult
cases. Stability and consistency and their consequences.

Expected Work: About 16 computer solutions and computer experiments.

Grades will be based on the computer assignments, a
take-home mid semester examination and a final examination.

References: Henrici, P., DISCRETE VARIABLE METHODS IN ORDINARY
DIFFERENTIAL EQUATIONS, John Wiley and Sons, Inc.,

April 1971
Instructor: R. Orgass

Objectives: The objective of this course is to introduce students to advanced topics in the theory of computation including the theory of computable functions, finite state machines, Turing machines, and switching circuits. This course is strongly recommended for students who are planning to pursue graduate work in computer science.

Prerequisites: 198:205-206 or equivalent; 198:311-312 (or 198:301-302) or equivalent; a course in logic such as Philosophy 332 (Intermediate Logic) is desirable. Students who register for this course should have strong preparation in the prerequisites. Those who have some doubt about their qualifications should meet with the instructor to discuss the course.

Outline: The following topics will be discussed in the course: finite state machines, neural networks, memories of events in finite-state machines, computability, effective procedures, algorithms, Turing machines, Universal Turing machines, limitations of effective computability, relations between Turing machines and recursive functions, Post symbol-manipulation system, Post's normal form theorem, and a survey of very simple bases for computability.

Expected Work: The course will consist primarily of lectures and discussion sessions. Students will have homework assignments regularly. The homework problems will ask the student to expand or elaborate results discussed in class and in the text. Regular completion of homework problems will be essential for learning the subject matter of the course. There will be either a final examination or a term paper; the decision will be made early in the semester.

Instructor: Staff

Objectives: To provide a forum for the study and discussion of the impact of computers on man and society. For students majoring in computer science, and also for others interested in exploring the social consequences of computer developments.

Prerequisites: At least one computer science course, and consent of the instructor in charge.

Outline: Review of computer applications in various domains—scientific, economic, social. Effects on individuals and organizations; effects on values, education, employment and management. An effort will be made to bring viewpoints from various disciplines (e.g., sociology, anthropology, economics, philosophy).

Expected Work: Study of reports and papers on selected topics; class presentations and discussions; work on small projects.

References: Selected papers in the literature will be introduced in the course.

April 1971
198:432  SEMINAR IN NON-NUMERICAL METHODS  Spring  (3 credits)

Instructor:  Staff

Objectives:  To provide a view of how non-numerical techniques (as
introduced in the course 198:212) may be applied to
various areas of current interest in research. Applications to language analysis will not be covered, as these
are handled in the course 198:401.

Prerequisites:  198:212 and consent of the instructor in charge.

Outline:  A limited number of areas of application will be covered
in some depth. Emphasis will be on the selection of
methods of representation of the universe of information
to be manipulated, and of efficient methods of handling
the information. These areas may include (though not
be restricted to) theorem proving, game playing, model
simulation, graphical manipulation, and analysis of al-
gorithms.

Expected Work:  The course will involve a good deal of reading of original
papers, and will require each student to make at least
one class presentation during the term. Programming
work will be a number of small problems rather than a
term project. The programming language LISP will be intro-
duced; a knowledge of SNOBOL will be assumed.

References:  Feigenbaum & Feldman, COMPUTERS AND THOUGHT, McGraw-Hill


Weissman, Clark, LISP 1.5 PRIMER, Dickenson Publishing

April 1971
INDEPENDENT STUDY B

198:495-496

Instructor: Staff (by arrangement)

Objectives: This course is intended mainly for computer science majors who are interested in gaining experience in software design and implementation by undertaking a substantial programming project.

Prerequisites: Consent of instructor (students should obtain and complete an application form at the department office prior to registration for independent study).

Outline: To be arranged.

Expected Work: Details to be arranged with supervising instructor. A number of suggestions for practical projects will be supplied by the computer center, CClS.

References:

April 1971
III. TYPICAL SCHEDULES

A. COMPUTER SCIENCE MAJORS:

Let us consider two examples of course schedules (only courses offered in our program are shown) for majors in computer science. In the first example, the student has satisfactory mathematical background upon entering the program; in the second example the student’s mathematical background is insufficient.

<table>
<thead>
<tr>
<th>Year</th>
<th>Semester</th>
<th>First Example</th>
<th>Second Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1st</td>
<td>105,111</td>
<td>090:119</td>
</tr>
<tr>
<td></td>
<td>2nd</td>
<td>106,</td>
<td>090:120</td>
</tr>
<tr>
<td>2</td>
<td>1st</td>
<td>205,211</td>
<td>105,111</td>
</tr>
<tr>
<td></td>
<td>2nd</td>
<td>206,212</td>
<td>106,</td>
</tr>
<tr>
<td>3</td>
<td>1st</td>
<td>311,321</td>
<td>205,211,311</td>
</tr>
<tr>
<td></td>
<td>2nd</td>
<td>312,322</td>
<td>206,212,312</td>
</tr>
<tr>
<td>4</td>
<td>1st</td>
<td>401,431</td>
<td>401,321,431</td>
</tr>
<tr>
<td></td>
<td>2nd</td>
<td>402,432</td>
<td>402,496</td>
</tr>
</tbody>
</table>

It is clear that both the student who is well prepared mathematically and the student with an insufficient mathematical background can complete the requirements for the major in four years (even though in the latter case, the first year is devoted to the introductory sequence in applied mathematics (090:119,120). Both students can accommodate in their schedules the fourteen semester courses required for the major, and they could also add more computer science courses in their third and fourth year if they so wished.

The following three subsections suggest typical course sequences for students who are not computer science majors. A student should choose a sequence based on his or her background and interest and the level of competence desired in the subject matter. We shall refer to the sequences as PLAN 1, PLAN 2, PLAN 3, etc. In general PLAN (j + 1) goes into the material in more depth than PLAN j. Sequences called PLAN 3A and PLAN 3B cover different material but at about the same depth. Clearly, combinations of plans are possible, and they can be formulated to fit the type of material and the depth in which the student is interested.

B. STUDENTS IN THE PHYSICAL AND BIOLOGICAL SCIENCES, MATHEMATICS, AND ENGINEERING

(a) The goal here is to give the student a working knowledge of computer problem solving techniques and numerical methods.
(b) The goal here is to give the student a background in computer organization, machine language programming, and systems software. (Note: 112 or 212 can be substituted for 211 in the plans below.)

<table>
<thead>
<tr>
<th>PLAN 1</th>
<th>PLAN 2</th>
<th>PLAN 3A</th>
<th>PLAN 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>211</td>
<td>211</td>
<td>211</td>
<td>211</td>
</tr>
<tr>
<td>311</td>
<td>311</td>
<td>311</td>
<td>311</td>
</tr>
<tr>
<td>312</td>
<td>312</td>
<td>512</td>
<td>401</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>402</td>
</tr>
</tbody>
</table>

PLAN 3B

| 211    |
| 311    |
| 312    |
| 402    |

C. STUDENTS IN SOCIAL AND BEHAVIORAL SCIENCES, HUMANITIES, AND BUSINESS

The goal is to give the student a working knowledge of computer problem solving techniques and non-numerical methods.

<table>
<thead>
<tr>
<th>PLAN 1</th>
<th>PLAN 2</th>
<th>PLAN 3A</th>
<th>PLAN 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>112</td>
<td>212</td>
<td>212</td>
<td>212</td>
</tr>
<tr>
<td></td>
<td>321</td>
<td>321</td>
<td>432</td>
</tr>
</tbody>
</table>

PLAN 3B

| 212    |
| 432    |

D. STUDENTS INTERESTED IN A GENERAL INTRODUCTION TO COMPUTERS, THEIR USES, AND THEIR SOCIAL IMPACT

(a) For those interested in a general introduction to computers:

<table>
<thead>
<tr>
<th>PLAN 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>112</td>
</tr>
</tbody>
</table>

(b) For those interested primarily in the social impact of computers:

<table>
<thead>
<tr>
<th>PLAN 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>112</td>
</tr>
<tr>
<td>431</td>
</tr>
</tbody>
</table>
IV. TRANSFER OF CREDITS

The regulations of Livingston College require that a student must receive a grade of B or better in a course in order to receive transfer credit at Livingston College. The Department of Computer Science at Livingston requires that a student complete fourteen Computer Science courses in order to receive a Bachelor's degree with a major in Computer Science. In some cases, it will be possible for a student to receive credit toward this requirement for courses taken at other institutions. In order to receive such credit, the student must request that the Department of Computer Science evaluate his transcript. Unless the student requests such an evaluation, he will not receive transfer credit toward his Computer Science major requirements for courses taken at other institutions.

The evaluation procedure has been designed to minimize the effort required both of students and of faculty members. The basic requirement for this procedure is that it provide sufficient information to enable the faculty to respond.

To request evaluation of his transcript, the student should write to:

Undergraduate Examinations and Scholastic Standing Committee
Department of Computer Science
Livingston College
Rutgers University
New Brunswick, New Jersey 08903

This may be done either by letter or by using a form provided by the Department of Computer Science. The student's letter should include a list of Livingston Computer Science courses for which he wishes to receive credit. To support this claim, the student should cite the courses which he has taken elsewhere, which he believes are equivalent. Upon completion of the evaluation, the student will receive a letter informing him of the courses which have been accepted as counting toward his Computer Science major requirements. A copy of this letter will be sent to the Livingston College Registrar.

If at all possible, the student should include with his request the following materials:

(1) a transcript of his record

(2) the catalog of the other institution

(3) a list of texts used in the courses for which he claims credit.

These materials will be returned to the student. They are required to reduce the time spent searching other records.
In some situations, it is essential for a student to have a fast response to evaluation requests. To accommodate this need, a fast evaluation procedure has been established. In order to use the fast procedure, the student must provide the items (1) to (3) above. A response will be available in two working days.
V. FACULTY

Members of the Computer Science Faculty and their research interests are listed below.

Saul Amarel, Professor, Chairman, and Director of Graduate Programs; B.S. (Technion, Israel); M.S., D.Eng.Sci. (Columbia); Artificial Intelligence, computer linguistics, theory of algorithms, information systems.

Stanley Baxendale, Associate Professor and Vice-Chairman; B.S. (Leeds), Computer graphics, automatic indexing.

David R. Beaucage, Assistant Professor; A.B., M.S. (Rutgers), Ph.D. (New York at Stony Brook); Non-numerical methods, computers in mathematical research.

William J. Carroll, Professor and Director of Educational Computer Center; A.B. (Catholic), M.B.A. (Detroit), Ph.D. (New York University); System selection, operations research. (on leave in 1971-72)

John T. Cox, Assistant Professor; B.S. (Purdue), M.E.E. (New York University); Time sharing, switching theory, automata theory, programming.

Sarah Ann Droese, Assistant Professor; B.S. (Michigan Technical University); Programming methods.

William B. Earston; Lecturer; B.S. (Cornell), Ph.D. (Princeton); Operating Systems, Time sharing.

William J. H. Fabens, Assistant Professor; B.A. (Cornell), M.S., Ph.D. (University of Wisconsin); Programming languages, interactive systems, instructional processes.

Gilbert Falk, Assistant Professor; B.S., M.S. (MIT), Ph.D. (Stanford); Artificial Intelligence, visual information processing, data structures.

Fred Fender, Professor of Computer Science and Mathematics; B.S., M.S., Ph.D. (University of Pennsylvania). Numerical analysis, computer solution of differential equations.

Kenneth R. Kaplan, Lecturer; B.E.E., M.E.E., Ph.D. (Polytechnic Institute of Brooklyn); Simulation, machine organization, stochastic processes.

Donald R. King, Associate Professor and Associate Director of Research, Center of Computer and Information Services; A.B., M.S. (Rutgers). Non-numerical data processing, information retrieval. (on leave in 1971-72)
Faculty (continued)

Casimir Kulikowski, Assistant Professor; B.S., M.S. (Yale); Ph.D. (University of Hawaii); Pattern recognition, decision processes.

Thomas H. Mott, Jr., Professor and Dean of The Graduate School of Library Service; A.B. (Rice), Ph.D. (Yale); Switching theory, programming theory, information systems.

Richard J. Orgass, Associate Professor; B.S. (City College, N.Y.); M.S., Ph.D. (Yale); Theory of computation, applications of logic to computer science.

Marvin C. Paull, Professor; B.S. (Clarkson); Theory of programming languages, translators, switching theory.

Irving N. Rabinowitz, Professor, and Director of Research, Center for Computer and Information Services; B.S. (City College of N.Y.); M.A., Ph.D. (Princeton). Programming languages, language processors, software design.

Chitoor V. Srinivasan, Research Associate Professor; B.Sc., D.M.I.T. (Madras), M.S., D.Eng.Sci. (Columbia); Description Languages, data structures, computer organization, coding theory.