## Discrete Mathematics

In the early $1980^{\prime}$ 's, as computer science enrollments ballooned on campuses across the country, the Mathematical Association of America established an ad hoc Committee on Discrete Mathematics to help provide leadership to the rapidly expanding efforts to create a course in discrete mathematics that would meet the needs of computer science and at the same time fit well into the traditional mathematics program. This committee issued a report in 1986 that conveyed their own recommendations together with an appendix that included reports from six experimental projects supported by the Alfred P. Sloan Foundation. This chapter contains the report of the Committee, without the appendix, preceded by a new preface prepared by Committee Chair Martha Siegel.

## 1989 Preface

In the years since 1986 when the Committee on Discrete Mathematics in the First Two Years published its report, there have been many changes in the attitudes of mathematics departments toward curricular change. The Committee had found that faculty in disciplines that required calculus were quite supportive of proposals to introduce more discrete mathematics into the first two years. They frequently complained about the state of calculus and encouraged us to get our house in better order. Many mathematicians also expressed dissatisfaction with the calculus sequence. The threat of replacing some of the traditional calculus material with discrete topics certainly helped to turn attention to the teaching of calculus. This movement toward a "calculus for a new century" is exciting and timely.

It is disappointing, however, that there seem to be only a few attempts to incorporate any significant discrete mathematics into the revision of the curriculum of the first two years. The discrete mathematics course seems to be established in most schools as a separate entity. It is encouraging that the National Council of Teachers of Mathematics has established a Task Force on Discrete Mathematics to help teachers implement curriculum standards for the inclusion of discrete mathematics in the schools.

Many new textbooks for the standard (usually one semester) discrete mathematics courses for the freshman or sophomore student have appeared or are in press. Publishers seem to find the market troublesome
because there is no consensus as to the exact content of the course. Those suggestions offered by the Committee as to topics that should be considered for inclusion may have been loosely followed, but level and attitude of books vary widely. A bibliography which is current as of the end of 1988 appears in a report on discrete mathematics edited by Anthony Ralston (to appear in the MAA Notes Series, 1989), as do final reports of the Sloan Foundation funded discrete mathematics projects.

New freshman-sophomore textbooks continue to appear. For the immediate future, it seems as if a oneor two-semester discrete mathematics course, independent of (but at the same level as) calculus will be the typical one. Recent advances such as ISETL, a computer language for the teaching of discrete mathematics (Learning Discrete Mathematics with ISETL, by Nancy Baxter, Ed Dubinsky, and Gary Levin), and the True BASIC Discrete Mathematics package(by John Kemeny and Thomas Kurtz, Addison-Wesley) may affect how the course evolves. Those few efforts to incorporate discrete mathematics into the calculus and the courseware that are being developed merit our attention for the future.

Who teaches discrete mathematics? Most mathematics departments have a course, though sometimes only on the junior-senior level. Sometimes the elementary course is offered in the computer science department or in engineering. The 1985-86 CBMS survey indicates that more than $40 \%$ of all institutions require discrete mathematics for computer science majors. Of universities and four-year colleges, about $60 \%$ require discrete mathematics or discrete structures. It is also not uncommon that mathematics majors are required to take some discrete mathematics.

The most recent accreditation standards issued by the Computer Sciences Accreditation Board include a discrete mathematics component. The most recent report of the ACM Task Force on the Core of Computer Science defines nine areas as the core of computer science (Communications of ACM, Jan. 1989, 32:1). In all but a few areas, discrete mathematics topics are listed as support areas. For algorithms and data structures, for exarnple, students should be familiar with graph theory, recursive functions, recurrence relations, combinatorics, induction, predicate and temporal logic, among other things. Boolean algebra and coding the-
ory are considered part of the architecture component. Students certainly would need discrete mathematics as a prerequisite for many of the computer core courses. Graph theory, logic, and algebra appear in a significant number of necessary support areas.

Although the Committee on Discrete Mathematics in the First Two Years was dissolved after its 1986 report was issued, concerns of the Committee have been incorporated into the mission of the MAA Committee on Calculus Revision and the First Two Years (CRAFTY). Aside from concentration on the calculus initiatives across the country, CRAFTY is interested in continuing the effort of the earlier group to see discrete mathematics become part of the typical freshman-sophomore curriculum in any of the mathematical sciences. The goal is to increase the effectiveness of the curriculum in serving other disciplines while providing enough excitement and challenge to attract talented undergraduates to major in mathematics.

Martha J. Siegel Towson State University March, 1989

## Introduction and History

The Committee on Discrete Mathematics in the First Two Years was established in the spring of 1983 for the purpose of continuing the work begun at the Williams College Conference held in the summer of 1982. That conference brought to a forum the issue of revising the college curriculum to reflect the needs of modern programs and the students in them. Anthony Ralston and Gail Young brought together 29 scientists (24 of whom were mathematicians) from both industry and academe to discuss the possible restructuring of the first two years of college mathematics. Although the growing importance of computer science majors as an audience for undergraduate mathematics was an important motivation for the Williams Conference, the conference concerned itself quite broadly with the need to revise the first two years of the mathematics curriculum for everyone-mathematics majors, physical science and engineering majors, social and management science majors as well as computer science majors. The papers presented and discussed at the conference, and collected in The Future of College Mathematics [35], reflect this breadth of view.

The word used to describe what was needed was "discrete" mathematics. Most of us knew what that meant approximately and respected the content as good mathematics. To illustrate the discrete mathematics topics
that might be considered for an elementary course, two workshop groups at the Williams Conference produced (in a very short time) a fairly remarkable set of two course sequences:

1. A two year sequence of independent courses, one in discrete mathematics and one in a streamlined calculus, and
2. A two year integrated course in discrete and continuous mathematics (calculus) in a modular form for service to many disciplines.
These course outlines were admittedly tentative and needed refinement and testing. At the same time, the CUPM-CTUM Subcommittee on Service Courses had been examining the traditional service course offerings of the first two years. The syllabi of these courses, in which many freshman and sophomores are required to enroll, are studied periodically for their relevancy. Finite mathematics, linear algebra, statistics, and calculus are considered to be essential to many majors, but with the importance of the computer, the Subcommittee on Service Courses concluded that even the mathematics majors need mathematics of a new variety, not only so they can take computer science courses, but also so they can work on contemporary problems in mathematics.

At that time, there were few or no textbooks or examples of such courses for the community to share. At the suggestion of the Subcommittee on Service Courses, the MAA agreed to help to develop the Williams courses further through the Committee on Undergraduate Program in Mathematics (CUPM) and the Committee on the Teaching of Mathematics (CTUM), standing committees of the Association. That led to the establishment of the committee responsible for this report. Funds for the effort were secured from the Sloan Foundation. The members of the committee were chosen especially to reflect the communities who would eventually be most affected by any changes in the traditional mathematics curriculum.

In addition to the development of course outlines and plans for their implementation, the committee was also involved in the observation of a set of experimental projects which also were begun as a result of the Williams Conference and the interest of the Sloan Foundation. After the conference, the Sloan Foundation solicited about thirty proposals for courses which would approximate the syllabi suggested by the workshop participants. The call for proposals particularly mentioned the need for the development of text material and classroom testing and emphasized the hope that some schools would make the effort to try the integrated curriculum. Six schools were ultimately chosen:

- Colby College, Waterville, Maine.
- University of Delaware, Newark, Delaware.
- University of Denver, Denver, Colorado.
- Florida State University, Tallahassee, Florida.
- Montclair State College, Montclair, New Jersey.
- St. Olaf College, Northfield, Minnesota.

The committee, together with some of the committee which chose the proposals to be funded (Don Bushaw, Steve Maurer, Tony Ralston, Alan Tucker, and Gail Young), monitored their progress for the two year period of funding, ending in August 1985. (Complete reports of these funded projects will appear in an MAA Notes volume on discrete mathematics to be published in 1989.)

## Summary of Recommendations

1. Discrete mathematics should be part of the first two years of the standard mathematics curriculum at all colleges and universities.
2. Discrete mathematics should be taught at the intellectual level of calculus.
3. Discrete mathematics courses should be one-year courses which may be taken independently of the calculus.
4. The primary themes of discrete mathematics courses should be the notions of proof, recursion, induction, modeling, and algorithmic thinking.
5. The topics to be covered are less important than the acquisition of mathematical maturity and of skills in using abstraction and generalization.
6. Discrete mathematics should be distinguished from finite mathematics, which as it is now most often taught, might be characterized as baby linear algebra and some other topics for students not in the "hard" sciences.
7. Discrete mathematics should be taught by mathematicians.
8. All students in the sciences and engineering should be required to take some discrete mathematics as undergraduates. Mathematics majors should be required to take at least one course in discrete mathematics.
9. Serious attention should be paid to the teaching of the calculus. Integration of discrete methods with the calculus and the use of symbolic manipulators should be considered.
10. Secondary schools should introduce many ideas of discrete mathematics into the curriculum to help students improve their problem-solving skills and prepare them for college mathematics.

## General Discussion

In its final report, the committee has decided to present two course outlines for elementary mainstream discrete mathematics courses. Our unanimous preference is for a one-year course, at the level of the calculus but independent of it. It is designed to serve as a service course for computer science majors and others and as a possible requirement for mathematics majors. The Committee on the Undergraduate Program in Mathematics (CUPM) has endorsed the recommendation that every mathematics major take a course in discrete mathematics, and has agreed that the year course the committee recommends is a suitable one for mathematics majors. It is expected that the course will be taken by freshmen or sophomores majoring in computer science so that they can apply the material in the first and second year courses in their major. The ACM recommendations [21, 22] for the first year computer science course presume, if not specific topics, then certainly the level of maturity in mathematical thought which students taking the discrete mathematics course might be expected to have attained. Hence, the Committee recommends that the course be taken simultaneously with the first computer science course. The Committee understands that at some schools the first computer science course may be preceded by a course strictly concerned with programming. At the very least, the Committee expects that the discrete mathematics course will be a prerequisite to upper-level computing courses. For this reason, the Committee has tried to isolate those mathematical concepts that are used in computer science courses. The usual sequence of these courses might determine what should be taught in the corresponding mathematics courses.

In addition to the Committee's concern for computer science majors, there is a high expectation that mathematics majors and those in most physical science and engineering fields will benefit from the topics and the problem-solving strategies introduced in this discrete mathematics course. Subjects like combinatorics, logic, algebraic structures, graphs, and network flows should be very useful to these students. In addition, methods of proof, mathematical induction, techniques for reducing complex problems to simpler (previously solved) problems, and the development of algorithms are tools to enhance the mathematical maturity of all. Furthermore, students in these scientific and mathematically-oriented fields will want to take computer science courses, and will need some of the same mathematical preparation that the computer science major needs.

Thus, the Committee has agreed to recommend that
the course be part of the regular mathematics sequence in the first two years for all students in mathematicallyrelated majors. Our contacts with physicists and engineers reinforce the idea that their students will need this material, but, of course, there is the concern that calculus will be short-changed.

The Committee will make several suggestions regarding the calculus, but individual institutions will best understand their own needs in this regard. We do not recommend that the third semester of calculus be cut from a standard curriculum. Serious students in mathematical sciences, engineering, and physical sciences need to know multivariable calculus. Many in the mathematical community recognize that the content of the calculus should be updated to acknowledge the use of numerical methods and computers, and promising initiatives along this line are being taken. Engineers have been especially anxious for this change. John Schmeelk surveyed 34 schools and compiled suggestions for revising the standard calculus. (Schmeelk's survey was included in the appendix to the original report of the Committee.) At some of the Sloan-funded schools and others, there have been attempts to revise the calculus to incorporate some discrete methods and to use the power of the symbolic manipulator packages. We describe these attempts later in the report. (A complete report on the Sloan-funded projects will be published in the MAA Notes Series in 1989.)

There is, inherent in our proposal, the possibility that some students may be required to take five semesters of mathematics in the first two years-a year of discrete mathematics and the three semesters of calculus. But, there is no reason why students cannot be allowed to take one of the five in the junior year. We point out that some linear algebra is included in the year of discrete mathematics. Additionally, the use of computers via the new and powerful symbolic manipulation packages may reduce the amount of time needed for the traditional calculus sequence.

A one-semester discrete mathematics course will be described in the appendix to this report as a concession to the political realities in many institutions. It has become obvious to the Committee over the last two years that at some colleges, there is a limitation on the number of new elementary courses that can be introduced at this time.

The Committee believes strongly that mathematics should be taught by mathematicians. Although there are some freshman-sophomore courses in discrete mathematics in computer science departments, the course presented here should, the Committee believes, be taught by mathematicians. The rigor and pace of this
course are designed for the freshman level. Some topics necessary for elementary computer science may have to be taught at an appropriate later time, either in a junior-level discrete mathematics course or in the computer science courses.

## Needs of Computer Science

What do the computer science majors need? In teaching the first year Introduction to Computer Science course, Tony Ralston kept track of mathematics topics he would have liked the students to have had before (or at least concurrently with) his course:

- Elementary Mathematics: Summation notation; subscripts; absolute value, truncation logarithms, trigonometric functions; prime numbers; greatest common divisor; floor and ceiling functions.
- General Mathematical Ideas: Functions; sets and operations on sets.
- Algebra: Matrix algebra; Polish notation; congruences.
- Summation and Limits: Elementary summation calculus; order notation, $0(f n)$; harmonic numbers.
- Numbers and Number Systems: Positional notation; nondecimal bases.
- Logic and Boolean Algebra: Boolean operators and expressions; basic logic.
- Probability: Sample spaces; laws of probability.
- Combinatorics: Permutations, combinations, counting; binomial coefficients, binomial theorem.
- Graph Theory: Basic concepts; trees.
- Difference Equations and Recurrence Relations: Simple differential equations; generating functions.
Many of the ideas are those that students should have had in four years of the traditional high school curriculum. In addition, there are some ideas and techniques that are probably beyond the scope of secondary school mathematics. An elaboration of this list appears in the Appendix and in an article by Ralston in ACM Communications [33].

Many proposals have been coming from the computer science community. Recommendations for a freshmanlevel discrete mathematics course from the Educational Activities Board of IEEE probably are the most demanding. Students enrolled in the course outlined in the appendix are first semester freshman also enrolled in the calculus according to the IEEE recommendations published in December 1983, by the IEEE Computer Society [19].

Accreditation guidelines passed recently by ACM and IEEE also require a discrete mathematics course. The recommendations for the mathematics component
of a program that would merit accreditation appear below. The criteria appear in their entirety in an article by Michael Mulder and John Dalphin in the April 1984 Computer [28].

Certain areas of mathematics and science are fundamental for the study of computer science. These areas must be included in all programs. The curriculum must include one-half year equivalent to 15 semester hours of study of mathematics. This material includes discrete mathematics, differential and integral calculus, probability and statistics, and at least one of the following areas: linear algebra, numerical analysis, modern algebra, or differential equations. It is recognized that some of this material may be included in the offerings in computer science ....
Presentation of accreditation guidelines which require one and one-half years of study in computer science, one year in the supporting disciplines, one year of general education requirements, and one-half year of electives induced quick and angry response. The liberal arts colleges and the small colleges unable to offer this number of courses or unwilling to require so many credits in one discipline, have responded in many ways. This Small College Task Force of the ACM issued its own report, approved by the Education Board of the ACM [5]. We emphasize only the mathematics portion of those guidelines.

Many areas of the computer and information sciences rely heavily on mathematical concepts and techniques. An understanding of the mathematics underlying various computing topics and a capability to implement that mathematics, at least at a basic level, will enable students to grasp more fully and deeply computer concepts as they occur in courses .... It seems entirely reasonable and appropriate, therefore, to recommend a substantial mathematical component in the CSIS curriculum . . . . To this end, a year of discrete mathematical structures is recommended for the freshman year, prior to a year of calculus.
The Sloan Foundation supported representatives of a few liberal arts schools in their attempt to define a high-quality computer science major in such institutions. Again, we put the emphasis on the mathematics component of the proposed program.

From Model Curriculum for a Liberal Arts Degree in Computer Science by Norman E. Gibbs and Allen B. Tucker [12]:

The discrete mathematics course should play an important role in the computer science curriculum .... We recommend that discrete mathematics be either a prerequisite or corequisite for CS2. This early positioning of discrete mathematics reinforces the fact that computer science is not just programming, and that there is substantial mathematical content throughout the discipline. Moreover, this course should have significant theoretical content and be taught at a level
appropriate for freshman mathematics majors. Proofs will be an essential part of the course.
Alfs Berztiss, a member of the Committee, led a number of mathematics and computer science faculty at a conference at the University of Pittsburgh in 1983 at which an attempt was made to formulate a high-quality program in computer science which would prepare good students for graduate study in the field. Details are available in a Technical Report (83-5) from the University of Pittsburgh Department of Computer Science [8]. Both that program and the new and extensive bachelor's program in computer science at Carnegie-Mellon University depend on an elementary discrete mathematics course.

In addition to the proposals for programs, the computer science community is in the process of revising elementary computer science courses. Though old courses stressed language instruction, a more modern approach stresses structured programming and a true introduction to computer science. The beginning courses CS1 and CS2 are described by the ACM Task Force on CS1 and CS2. We quote from the article by Elliot Koffman, et al. [21] about the role of discrete mathematics in the structure of these computer science courses.

We are in agreement with many other computer scientists that a strong mathematics foundation is an essential component of the computer science curriculum and that discrete mathematics is the appropriate first mathematics course for computer science majors. Although discrete mathematics must be taken prior to CS2, we do not think it is a necessary prerequisite to CS1. ... We would . . . expect computer science majors and other students interested in continuing their studies in computer science to take discrete mathematics concurrently with the revised CS1.
If high schools and colleges take the recommendation seriously, the student enrolled in CS1 would be enrolled in a discrete mathematics course concurrently. That mathematics course would be required as a prerequisite for CS2. Of all the recommendations, this is likely to have the largest impact on enrollments in discrete mathematics courses.

## Syllabus

What are the common needs of mathematics and computer science students in mathematics? The Committee agrees that all the students need to understand the nature of proof, and the essentials of propositional and predicate calculus. In addition, all need to understand recursion and induction and, related to that, the analysis and verification of algorithms and the algorithmic method. The nature of abstraction should be part
of this elementary course. While some of the Committee supported the introduction to algebraic structures in this course, particularly for coding theory and finite automata, others felt that those concepts were best left to higher-level courses in mathematics. The basic principles of discrete probability theory and elementary statistics might be considered to be as important and more accessible to students at this level. Professionals in all disciplines cite the importance of teaching problem solving skills. Graph theory and combinatorics are excellent vehicles. All these students need some calculus.

The Committee recommends the inclusion of as many of the proposed topics as possible with the understanding that taste and the structure of the curriculum in each institution will dictate the depth and extent to which they are taught. The ability of students in a course at this level must be considered in making these choices. While one of the goals of the course is to increase the mathematical maturity of the student, some of the mathematical community who have communicated to the panel about their experiences teaching this course have indicated that there are prerequisite skills in reading and in maturity of thinking that really are needed, perhaps even more than in the calculus.

The Committee recognizes that it might be some time before there is as much agreement on the content of a discrete mathematics sequence as there is now about the calculus sequence. In the meantime, diversity and variety should be encouraged so that we may learn what works and what does not. In any case, the Committee strongly endorses the notion that it is not what is taught so much as how. If the general themes mentioned in the previous paragraph are woven into the content of the course, the course will serve the students well. Adequate time should be allowed for the students to do a lot on their own: they should be solving problems, writing proofs, constructing truth tables, manipulating symbols in Boolean algebra, deciding when, if, and how to use induction, recursion, proofs by contradiction, etc. And their efforts should be corrected.

We have been asked about the role of the computer in this course. To a person we have agreed that this is a mathematics course and that while students might be encouraged, if they have the background, to try the algorithms on a computer, the course should emphasize mathematics. The skills that we are trying to teach will serve the student better than any programming skills we might teach in their place, and the computer science departments prefer it that way. Surely the ideal would be that students be concurrently enrolled in this course and a computer science course where the complimen-
tary nature of the subjects could be made clear by both instructors.

Algorithms are, of course, an integral part of the course. There is still no general agreement on how to express them in informal language. While a form of pseudocode might suit some people, others have found that an informal conversational style suffices. The Committee would not want to make any specific recommendations except that the student be precise and convey his/her methods. It is certainly not necessary to write all algorithms in Pascal. Communication is the key.

The recommendations for a one-year discrete mathematics course are presented in several ways. An outline of the course appears below. In the Appendix, the outline has been expanded to include objectives and sample problems for each topic. The scope and level of the course can be appreciated best from the expanded version.

Discrete Mathematics<br>A One Year Freshman-Sophomore Course<br>(Preliminary Outline)

Prerequisite: Four years of high school mathematics; may be taken before, during, or after calculus I and II.

1. Sets. Finite sets, set notation, set operations, subsets, power sets, sets of ordered pairs, Cartesian products of finite sets, introduction to countably infinite sets.
2. The Number System. Natural numbers, integers, rationals, reals, $Z n$, primes and composites, introduction to operations, and algebra.
3. The Nature of Proof. Use of examples to demonstrate direct and indirect proof, converse and contrapositive, introduction to induction, algorithms.
4. Formal Logic. Propositional calculus, rules of logic, quantifiers and their properties, algorithms and logic, simplification of expressions.
5. Functions and Relations. Properties of order relations, equivalence relations and partitions, functions and properties, into, onto, 1-to-1, inverses, composition, set equivalence, recursion, sequences, induction proofs.
6. Combinatorics. Permutations, combinations, binomial and multinomial coefficients, counting sets formed from other sets, pigeon-hole principle, algorithms for generating combinations and permutations, recurrence relations for counting.
7. Recurrence Relations. Examples, models, algorithms, proofs, the recurrence paradigm, solution of difference equations.
8. Graphs and Digraphs. Definitions, applications, matrix representation of graphs, algorithms for path problems, circuits, connectedness, Hamiltonian and Eulerian graphs, ordering relations-partial and linear ordering, minimal and maximal elements, directed graphs.
9. Trees. Binary trees, search problems, minimal spanning trees, graph algorithms.
10. Algebraic Structures. Boolean algebra, semigroups, monoids, groups, examples and applications and proofs; or
11. Discrete Probability and Descriptive Statistics. Events, assignment of probabilities, calculus of probabilities, conditional probability, tree diagrams, Law of Large Numbers, descriptive statistics, simulation.
12. Algorithmic Linear Algebra. Matrix operations, relation to graphs, invertibility, row operations, solution of systems of linear equations using arrays, algebraic structure under operations, linear programming-simplex and graphing techniques.

## Preparation for Discrete Mathematics

A consideration of the topics listed in this course outline reveals that, while the course meets our objectives of scope and level, this is a serious mathematics course. The student will have to be prepared for this course by an excellent secondary school background. Those of us who have been teaching freshmen know that many students are coming unprepared for abstract thinking and problem solving. We are aware that many secondary schools are doing a fine job of educating students to handle this work, but many more schools are not. It seems likely that courses ordinarily taught to mathematically deficient first-year students to prepare them for the calculus would also prepare them for this course. In many cases, with only modest changes, these courses can be adapted to be both prediscrete mathematics and precalculus. The Committee expects that the major in computer science will include at least one year of calculus so that at some time the student will surely reap the full benefits of these traditional preparatory courses.

The additional question still remains unansweredwhat should be taught in the high schools or on the remedial level in the colleges to prepare students adequately for this course? Our suggestion is tentative: some of us feel that perhaps a revived emphasis on the use of both formal and informal proof in geometry courses as a means for teaching methods of proof and analytic thinking would be a step in the right direction.

Others of us are not so sure. Increased use of algorithmic thinking in problem solving could be easily adapted to many high school courses. Readers are encouraged to read Steve Maurer's article in the September 1984 Mathematics Teacher for more on this subject.

The Committee on Placement Exarninations of the MAA will be attempting to isolate those skills that seem to be needed by students taking discrete mathematics. Although this study might not lead to the development of a placement examination for the course, it will help to explain what might be the appropriate preparation for a successful experience in such a course.

Year after year we face students who claim that they have never seen the binomial theorem, mathematical induction, or logarithms before college. These used to be topics taught at the eleventh or twelfth grade levels. What has happened to them? Students also say that they never had their papers corrected in high school so they never wrote proofs. Some of us have students who cannot tell the hypothesis from the conclusion.

Simple restoration of some of the classical topics and increased emphasis on problem solving might make the proposed course much easier for the student. As one studies the list of topics in the discrete mathematics course, it becomes clear that, in fact, there is little in the way of specific prerequisites for such a course except a solid background in algebra; nothing in the course relies on trigonometry, number theory, or geometry, per se. However, the abstraction and the emphasis on some formalism will shock the uninitiated and the mathematically immature.

Recent experimentation at the Sloan-funded schools might tell us something about what we ought to require of students enrolling in this type of course. Results from these schools have not been completely analyzed, but the failure rates seem consistent with those in the calculus courses. Some of the experimental group had taken calculus first and others had not. There seemed to be a filtering process in both cases so that results are not comparable from one discrete mathematics course to another. One Sloan-funded correspondent reported that reading skills might be a factor in success and was following through with a study to see if verbal SAT scores were any indicator of success.

One of the concerns of the Committee throughout its deliberations has been the articulation problem with a course of this kind. We want to be clear that finite mathematics courses in their present form are not the equivalent of this course. We have not totally succeeded in communicating this in presentations at professional meetings. The discrete and finite mathematics courses differ in several ways. First, the discrete mathematics
course is not an all-purpose service course. It has been designed primarily for majors in mathematically-related fields. It presumes at least four years of solid secondary school mathematics and hence the level of the course is greater than or equal to the level of calculus. There is inherent in this proposal a heavy emphasis on the use of notation and symbolism to raise the students' ability to cope with abstraction. Secondly, a heavy emphasis on algorithmic thinking is also recommended.

The pace, the rigor, the language, and the level are intended to differ from a standard finite mathematics course. We do not claim that this course can be taught to everyone. Perhaps at some schools the computer science majors are not very high caliber and college programs naturally are geared to the needs of the students. There is nothing inherently wrong in requiring that such students take the mathematics courses required of the business majors: finite mathematics, basic statistics, and "soft" calculus. Perhaps the finite mathematics courses can be improved and sections for some students be enhanced by teaching binary arithmetic and elementary graphs. This is an alternative that many schools will probably choose. It may reflect the reality on a campus where there is really no major in computer science, but a major in data processing or information science which serves its students well. We have not attempted to define that kind of discrete mathematics course. We specifically are defining a course on the intellectual level of calculus for science and mathematics majors. Our visits around the country indicate that many schools need a course at the level of the present finite mathematics offerings. Such courses are a valuable service to some students, but should not be considered equivalent to the course we have described.

## Two Year Colleges and High Schools

The mathematics faculty at two year colleges have been working through their own organizations and committees toward curricular reform. The Committee on Discrete Mathematics has attempted to consider their proposals in its own. Jerry Goldstein, Chairperson of CUPM and an ex-officio member of the Committee on the Curriculum at the Two Year Colleges, has been working to maintain articulation between the two groups. The Two Year College Committee began its deliberations after our Committee, so this report reflects only preliminary conclusions from that source. A "Williams"-like conference for the two year colleges took place in the summer of 1984 and proceedings are available from Springer-Verlag in New Directions in Two Year College Mathematics, edited by Donald Albers, et
al. The situation at this time in the two year colleges is one of exploration, learning, and waiting.

Just as the calculus sequence at two year colleges is taught from the same texts and in the same manner as at the baccalaureate institutions, discrete mathematics courses at two year schools are expected to conform to requirements of four-year schools to which students hoped to transfer. Faculty at Florida State University, in connection with one of the Sloan projects, introduced the discrete mathematics course at a nearby two year college. The course was taught from the same text and in the same manner at both institutions. The students did well and project directors claim the results were "unremarkable."

Recent conferences of the American Mathematical Association of Two Year Colleges (AMATYC) and associations of two year college mathematics faculty in many state organizations have been devoted to the special problems of the two year schools with regard to discrete mathematics. The primary concern of most schools is that they must wait for the four-year schools to indicate what type of course will be transferable. The Committee urges those teaching at four-year institutions to make a special effort to communicate their own requirements to the two year colleges that feed them.

What about discrete mathematics in the high schools? Perhaps it will be an exciting change to see the secondary schools place less emphasis on calculus and more on some of the topics in the discrete mathematics. We understand that there is considerable pressure from parents to have Junior (or Sis) take calculus in high school. We are confident that that will change as the first year of mathematics in the colleges becomes more flexible to include either calculus or discrete mathematics at the same level. If the high schools continue the trend to teaching more computer science for advanced placement, then they will have to offer the discrete mathematics to their students. The present Advanced Placement Examination in computer science is essentially for placement in CS2. To place above CS2, there will probably be a level II examination which conforms to the course outline for CS1 and CS2 as noted in the Koffman report. An Advanced Placement Examination in discrete mathematics is some time in the future, as there is no universal agreement as to exactly what might be included at this time.

In January 1986 a Sloan-funded conference on calculus was held at Tulane University in New Orleans. More than twenty participants presented papers and participated in workshops on the state of calculus and its future. The Committee concurs with that group's consensus that the goals of teaching (mathematics) are
to develop increased conceptual and procedural skills, to develop the ability of students to read, write, and explain mathematics, and to help students deal with abstract ideas. These are the global concerns for all mathematics teaching. Secondary schools should be working toward such goals too.

The Committee encourages faculty to get students to work together to solve problems. From experience, some of us have found that students cannot read a problem-either they leave out essential words or do not know how to read the notation when asked to read aloud. The word "it" should be banned from their vocabulary for a while. Students who use the word frequently do so because they do not know what "it" really is. Correcting students' homework has always been one of the best ways of understanding their misconceptions. In discrete mathematics courses this is even more so-concept and procedure vary from problem to problem. Students have to think and be creative. That's tough. They need the re-enforcement of the teacher's comments and the chance to try again. Working with other students should be encouraged because this forces students to speak. This oral communication helps them to learn the terminology and helps them to present clear explanations.

## The Impact on Calculus

The concerns of some people that the introduction of discrete mathematics will cause a major change in the calculus will probably prove to be unfounded. However, the Committee believes that there are several important questions to be addressed. We should be asking ourselves if we are doing the best job of teaching calculus. Some of our colleagues outside of mathematics who teach our calculus students have commented to the committee members that there are many aspects of the calculus which seem to be ignored in the present courses. There is widespread dissatisfaction with the problem-solving skills of calculus students. Problems that look even a little different from the ones that they have solved in the standard course are often impossible for students. In addition, we are being held responsible for our students lack of knowledge of numerical techniques. The discrete aspect of the calculus was continually stressed by our respondents. In fact, many commented that we were promoting the idea of a dichotomy in mathematics where there is none by not proposing an integrated program of discrete and continuous mathematics for the first two years. The Committee admits that at this time it is
presenting a feasible solution as opposed to the ideal solution.

Should the teaching of calculus reflect the tremendously powerful symbolic manipulators now on the market? While the most powerful require mainframes, some are available on minicomputers and muMath runs on a personal computer. Can the time previously spent in tedious practice of differentiation or integration be better used to teach the power of the calculus through problem solving and modeling? Two of the Sloan-funded schools-Colby College and St. Olaf College-did experiment with the use of MACSYMA, MAPLE, and SMP in the teaching of calculus. At Colby, in a course offered to those who had high school calculus, the computer packages were used to augment the one year single-variable and multivariate calculus course. At St. Olaf, SMP was used in an elective course during a January Interim between the first and second semesters of the standard calculus course. Kathleen Heid writes in The Computing Teacher [17] about her experience at the University of Maryland where she taught a section of the "soft" calculus using muMath. The results of all these experiments are quite favorable and indicate an important new consideration in our teaching of the subject.

What about the use of the methods introduced in discrete mathematics in the other courses in the curriculum, including calculus and analysis? What of difference equations? The Committee requested that physical scientists and engineers respond to the idea of changing the calculus. We mentioned the possibility that calculus might contain ideas from discrete mathematics in the solving of traditional calculus-type problems. Several engineers and physicists have responded to our query with some interesting endorsements for change. Those who responded felt that the present mathematical training we offer their professions is inconsistent with what many of them were doing in their jobs-for they were using difference equations and other discrete methods in their everyday applications.

We also should be asking what calculus the computer science major needs. Does the computer science student need the calculus to do statistics and probability? If so, how much rigor is needed? What background is needed in numerical methods? Should mathematics departments be teaching numerical methods? Are the requirements different from numerical analysis? Should we emphasize rigor, technique, or problem-solving skills? Do the traditional courses suffice to encourage integration of discrete and continuous mathematics?

## Conclusion

This report is both incomplete and already out-ofdate. Questions will continue to arise; answers are not easily found. Textbooks are now being published that are marketed as suitable for elementary discrete mathematics courses. Our annotated bibliography is undoubtedly incomplete. We know of several forthcoming texts that are in manuscript form but which are unlisted because they could not be properly reviewed.

There has been a great deal of interest, much of it enthusiastic, in the revitalization of the elementary college-level mathematics curriculum. The committee members have had the opportunity to visit schools, speak at sectional and national meetings, and to speak personally with hundreds of our colleagues. We are wrestling with problems of ever-changing demands from other disciplines-some, as computer science, so young there is no standard curriculum. We need to adjust our ideals to the realities of our own academic situation. The Committee attempted to propose a course with enough flexibility to allow institutions with different needs to follow the general course outline, putting emphases where they wanted.

The two year colleges and the high schools are dealing with demands of the four-year institutions, parents, and the College Entrance Examination Board. They feel many pressures to keep calculus as the pivotal course. On the other hand, the proposal to integrate discrete mathematics into the high school and even elementary school curricula got considerable support at the 1985 National Council of Teachers of Mathematics (NCTM) meetings in San Antonio.

The recent publication of many discrete mathematics textbooks suitable for the freshman-sophomore year
has been exciting. We have the opportunity to see what is successful. The Committee agrees that the next step in the development of the curriculum should be the integration of the discrete and the continuous ideas of mathematics into all courses. That would be ideal and we encourage experimentation to that end.

## Committee Members

Martha J. Siegel, Chair, Department of Mathematics, Towson State University; Member of CTUM.
Alfs Berztiss, Department of Computer Science, University of Pittsburgh; Representative of the ACM Education Board.
Donald Bushaw, Department of Pure and Applied Mathematics, Washington State University; Member of CTUM and of CUPM, Chair of MAA Committee on Service Courses.
Jerome Goldstein, Department of Mathematics, Tulane University; Chair of CUPM.
Gerald Isaacs, Department of Computer Science, Carroll College; Representative of the ACM Education Board.
Stephen Maurer, Department of Mathematics, Swarthmore College; then on leave at The Alfred P. Sloan Foundation.
Anthony Ralston, Department of Computer Science, State University of New York at Buffalo; Member of MAA Board of Governors, and organizer of the Williams Conference.
John Schmeelk, Department of Mathematics, Virginia Commonwealth University; Member of The American Society for Engineering Education (ASEE) Mathematics Education Committee.

## Course Objectives and Sample Problems

## 1. Sets

## Student Objectives:

- Understand set notation.
- Recognize finite and infinite sets.
- Be able to understand and manipulate relations between sets, and make proper use of such terms as:


## subsets

proper subsets
supersets
equality
universe and empty set.

- Understand and be able to manipulate indexed collections of sets.
- Understand and use the set-builder notation.
- Understand and manipulate operations on sets:
intersection (finite and countable collections)
union (finite and countable collections)
difference
symmetric difference
complement
Venn diagrams.
- Understand the proofs of theorems and know the laws:
commutative laws
associative laws
distributive laws
DeMorgan's laws.
- Understand Cartesian products of sets and power sets.
- Understand inductive (recursive) definitions of sets.
- Understand a few applications: for example, grammars as sets.
- Be able to do simple proofs by using Venn diagrams or elementary elementwise proofs.


## Sample Problems:

1. List the ordered pairs in the sets

$$
\begin{aligned}
& A=\{(m, n) \in S \times T: m<n\} \\
& B=\{(m, n) \in S \times T: m+1=n\}
\end{aligned}
$$

where $S=\{1,2,3,4\}$ and $T=\{0,2,4,5\}$.
2. True or false?

$$
A \backslash(B \cup C)=(A \backslash B) \cup(A \backslash B)
$$

Verify your answer (use elementwise argument, Venn diagram and algebraic manipulation).
3. Let $\left.A_{n}=\underset{5}{\{ } k \in \mathrm{P}: k \leq n\right\}$ for each $n \in \mathrm{P}$.

Find $\bigcap_{n=1}^{5} A_{n}, \bigcap_{n=1}^{\infty} A_{n}, \bigcup_{n=1}^{5} A_{n}, \bigcup_{n=1}^{\infty} A_{n}$.
Find $A_{n}^{c}$ and $A_{n} \cap A_{m}$ for $n, m \in \mathbf{P}$.
4. For each $n \in N$, let

$$
A_{n}=\left\{x \in \mathbb{Q}: x=m / 3^{n} \text { for some } m \in \mathbf{Z}\right\}
$$

Describe the set $A_{0}, A_{1}, A_{2}$, and $A_{11} \backslash A_{0}$.
Find $\bigcap_{n=0}^{\infty} A_{n}$.
Find $\bigcap_{n=2}^{\infty} A_{n}$.
5. Sketch the following set $S$ in $\mathbf{N} \times \mathbf{N}$ :

$$
\begin{gathered}
(0,0) \in S \text { and if }(m, n) \in S \text { then }(m, n+1) \in S \\
(m+1, n+1) \in S \text { and }(m+2, n+1) \in S
\end{gathered}
$$

Show $S=\{(m, n): m \leq 2 n\}$.
6. Show that if $A \subseteq B$ and $C \subseteq D$, then $A \times C \subseteq B \times D$.
7. Assume $A, B$, and $C$ are subsets of a universal set $U$. Simplify

$$
(A \cap(B \backslash C))^{c} \cup A
$$

## 2. The Number System

## Student Objectives:

- Be able to define
positive integers ( P )
natural numbers (N)
integers (Z)
rational numbers (Q)
irrational numbers
reals (as $(-\infty, \infty)$ ).
- Be able to recognize subsets of $N, P, Q$, and $Z$.
- Be able to use interval notation.
- Understand the division algorithm and divisibility.
- Be able to do simple proofs about even and odd numbers (e.g., the sum of two even integers is even).
- Know the definition of prime number, gcd and lcm.
- Be able to find the prime factorization of a number.
- Know how to write an integer given in base 10 as a numeral in base 2.


## Sample Problems:

1. Find elements in the sets (if the set is infinite, list five elements of the set).

$$
\begin{aligned}
& \left\{n \in \mathbf{N}: n^{2}=4\right\} \\
& \{n \in \mathbf{P}: n \text { is prime and } 1 \leq n \leq 20\} \\
& \left\{x \in \mathbf{R}: x^{2}=4\right\} \\
& \left\{n \in \mathbf{P}: n^{2}=3\right\} \\
& \left\{x \in \mathbf{R}: x^{2} \leq 4\right\} \\
& \left\{x \in \mathbf{R}: x^{2}<0\right\} \\
& \left\{x \in \mathbb{Q}: x^{2}=3\right\} \\
& \{x \in \mathbb{Q}: 2<x<3\}
\end{aligned}
$$

2. Determine how many elements are in each set? Write $\infty$ if the answer is infinite.

$$
\begin{aligned}
\{-1,1\}, & \{-1,0\},[-1,1],[-1,0], P(\mathbf{Z}) \\
P([-1,1]), & P(\{-1,1\}),\{n \in \mathbb{Z}:-1 \leq n \leq 1\} \\
& \{n \in \mathbf{Z}:-1<n<1\}
\end{aligned}
$$

3. List elements in

$$
\begin{aligned}
& A=\{n \in \mathbf{Z}: n \text { is divisible by } 2\} \\
& B=\{n \in \mathbf{Z}: n \text { is divisible by } 7\} \\
& C=A \cup B \\
& D=A \cap B
\end{aligned}
$$

4. Prove that the product of even integers is an even integer.
5. Use the Euclidean Algorithm to determine the greatest common divisor of 741 and 715.
6. Find the prime factorization of 4,978 .
7. Determine the numeral in base 2 to represent 81 (base 10).

## 3. The Nature of Proof

## Student Objectives:

- Be able to identify the hypothesis and the conclusion in sentences of various English constructions.
- Understand the definition of a proposition, its converse, its contrapositive.
- Understand the use of examples as an aid to finding a proof and the misuse of examples as proof.
- Understand the use of counterexamples.
- Be able to do direct proofs, including proof by cases.
- Be able to do indirect proofs.
- Understand the role of axioms and definitions.
- Understand the backward-forward method of constructing a proof.
- Understand and be able to use the principle of mathematical induction.
- See the necessity for the verification of algorithms.
- Do a substantial number of elementary proofs using simple examples from arithmetic.


## Sample Problems:

1. According to the Associated Press, a prominent public official recently said: "If a person is innocent of a crime, then he is not a suspect." What is the contrapositive of this quotation?
2. Prove or disprove the following statement about real numbers $x$ :

$$
\text { If } x^{2}=x, \text { then } x=1
$$

What is the converse of this statement? Prove it or disprove it.
3. After considering some examples if necessary, guess a formula that gives the sum of the interior angles at the vertices of a convex polygon in terms of the number $n$ of sides. Then prove the formula, if you can, by mathematical induction.
4. Write an algorithm for finding the least common denominator of two fractions. Can you think of another?
5. Write an algorithm for finding the median of a list consisting of $n$ (an odd number) real numbers.
6. Prove: if $A \cup B \subseteq A \cap B$, then $A=B$.
7. Prove or disprove: if $A \cap B=A \cap C$, then $B=C$.
8. Prove (by cases): for every $n \in \mathcal{N}, n^{3}+n$ is even.
9. Prove: for every $n \in P$,

$$
1+3+5+\cdots+(2 n-1)=n^{2}
$$

10. Prove (by contradiction): if $\boldsymbol{x}^{2}$ is odd, then $x$ is odd.
11. Prove: There are no integers $a$ and $b$ such that $a^{2}=$ $3 b^{2}$.
12. Prove: $n^{3}-n$ is divisible by 3 for every $n \in P$.

## 4. Formal Logic

## Student Objectives:

- Write English sentences for logical expressions and vice versa.
- Complete the truth tables for the standard logical connectives.
- Give the truth values of simple propositions given in plain English.
- State the definitions of tautology and contradiction.
- Prove and use the standard logical equivalences: commutative, associative, distributive, and idempotent properties; double negation; DeMorgan laws.
- Recognize computer language commands for standard logical operations.
- State and use logical implications, at least: modus ponens, modus tolens, transitivity of $\rightarrow$ and $\leftrightarrow$.
- Negate $p \vee q, p \rightarrow q, p \wedge q$.
- Identify the basic quantifiers, free and bound variables, negations and the generalized DeMorgan laws for quantified statements (e.g., $\neg \forall x p(x) \Longleftrightarrow$ $\exists x \neg p(x))$.
- Build logic circuits with AND, OR, NOT gates.
- Understand the terms consistency, inconsistency, completeness and decidability (optional).


## Sample Problems:

1. Use a truth table to prove that
$(p \wedge q) \rightarrow r$ is logically equivalent to $p \rightarrow(q \rightarrow r)$.
2. Prove that $\neg p \wedge r$ is logically equivalent to $\neg(p \vee \neg r)$ without using truth tables.
3. If $p=$ "cows bark", $q=$ "the Orioles are Baltimore's baseball team" and $r=" 2+4=7 "$, find truth values of $p \wedge q, p \rightarrow q,(p \wedge q) \rightarrow r$.
4. Consider the proposition for $x \in \mathbb{R}$ :

$$
\text { If }(x-3)(x-2)=0 \text { then either } x=3 \text { or } x=2
$$

a) Write its converse.
b) Write its contrapositive.
c) Write its negation.
d) What is the truth value of the proposition, its converse, its contrapositive, its negation?
5. Find the result of
[(0 AND 1) NAND 0] OR NOT [1 IMP 0].
6. Draw a logic circuit representing $(\neg p \wedge q) \vee r$.
7. Prove the following logical argument: $p \wedge q, p \rightarrow r$, $\neg s \rightarrow q$, and $s \rightarrow t$ imply $r \wedge t$.
8. Determine truth values of the following propositions. Assume the universe is $\mathbf{N}$.
(a) $\forall m \exists n\left[m=n^{2}\right]$
(b) $\exists m \forall n\left[m=n^{2}\right]$
9. Write in logical form: for every $x, y \in R$, there exists $z \in \mathbf{R}$ such that $x<z$ and $z<y$.
10. Negate $\forall x[p(x) \wedge \neg q(x)]$.
11. Answer each of the following in the appropriate box.
(a) If the book costs more than $\$ 20$, it is a bestseller. The book costs more than $\$ 20$. Is the book a best-seller? $\square$ yes, $\square$ no, $\square$ not enough information.
(b) If the kite is multicolored, it will fly. The kite flies. Is the kite multicolored? $\square$ yes, $\square$ no, $\square$ not enough information.
(c) If the bed is comfortable, Sally will sleep. The bed is not comfortable. Will Sally sleep? $\square$ yes, $\square$ no, $\square$ not enough information.
(d) If the candidate is elected in Vermont, she will be elected by the country. The candidate is not elected by the country. Is the candidate elected in Vermont? $\square$ yes, $\square$ no, $\square$ not enough information.
12. Simplify the logic circuit below:


## 5. Functions and Relations

## Student Objectives:

- Be able to define "function" and "relation".
- Know the properties of relations:
reflexive
transitive
symmetric
antisymmetric.
- Be able to identify order relations.
- Be able to identify equivalence relations.
- Understand the relationship between equivalence relations and partitions.
- Know the definitions of domain, codomain, image, into, onto (or surjection), one-to-one (or injection), bijection.
- Be able to do simple proofs involving these definitions.
- Be able to work with composition and inverses of relations and, in particular, of functions.
- Be familiar with recursive definitions of functions.
- Be introduced to sequences as functions, again with some emphasis on recurrence relations and recursion.
- Be able to do proofs involving recursion.
- Be able to work with definitions of relations as ordered pairs as opposed to as "rules".
- Know the definition of the characteristic function of a set.


## Sample Problems:

1. Give at least one reason why each of the following does not define an equivalence relation on the set of integers:
a) $x+y$ is odd;
b) $x<2 y$.
2. Recall that a positive integer is prime if it has exactly two positive integer divisors: itself and 1 . Consider the relation defined on the set of all integers greater than 1 by: " $y$ is the smallest prime that is a divisor of $x$."
a) Explain why this relation is a function $y=f(x)$.
b) What is the range of $f$ ?
c) List four elements of $f^{-1}(5)$.
d) Prove that $f \circ f=f$.
3. When the prevailing rate of interest is $100 r \%$, an account that has $P$ dollars in it at the beginning of a year should have how much in it at the beginning of the next year? Express your answer as a recursion formula, and solve it to find the size of such an account after $n$ years.
4. The factorial, usually denoted by $n$ !, of a positive integer $n$ is the product of all positive integers from 1 to $n$ inclusive. Show how $n$ ! may be defined recursively.
5. Prove that $f(x)=2 x+1$ is one-to-one and onto from $\mathbf{R}$ to $\mathbf{R}$. Is $f$ one-to-one from $\mathbf{Z}$ to $\mathbb{Z}$ ? Does $f$ $\operatorname{map} \mathbf{Z}$ onto $\mathbf{Z}$ ? Verify your answers.
6. List five elements in the sequence given by $a_{0}=1$, and $a_{n}=2 a_{n-1}$ for $n \geq 1$ ). Give another formula for $a_{n}$, for any $n \in \mathbf{N}$.
7. Let $\Sigma=\{a, b\}$ and let $\Sigma^{*}$ be the set of words over $\Sigma$. If $w_{1}$ and $w_{2}$ are elements of $\Sigma^{*}$, define $w_{1} \leq w_{2}$ if and only if length $\left(w_{1}\right) \leq$ length $\left(w_{2}\right)$. Is $\leq$ a partial order? Why?
8. Prove: If $h(1)=1$ and $h(n+1)=2 \cdot h(n)+1$ for $n \geq 1$, then $h(n)=2^{n}-1$ for all $n \in P$.
9. For $m, n \in \mathbf{N}$ define
$m \sim n$ if and only if $m^{2}-n^{2}$ is divisible by 3.
Prove that $\sim$ is an equivalence relation on N. Find 8 elements of each of the equivalence classes [0] and [1]. What is the partition of $N$ induced by $\sim$ ?

## 6. Combinatorics

## Student Objectives:

- Be able to apply the basic permutation and combination formulas.
- Be familiar with and be able to provide basic combinatorial identities using combinatorial reasoning.
- Be able to use the binomial theorem.
- Be able to do ball and urn type problems.
- Be able to state and apply the inclusion-exclusion principle.
- Be able to apply the pigeon-hole principle.
- Be familiar with combinatorial algorithms based on recurrence relations.
- Be introduced to the basic ideas of intuitive discrete probability.


## Sample Problems:

1. In many states automobile license plates consist of three (capital) letters followed by three digits. Are there any states in which this probably does not give enough different license plates even if discarded plate numbers can be reused? Are there any states in which three letters followed by three digits or three digits followed by three letters is probably not enough? How many license plates are possible in your state?
2. In a hypnosis experiment, a psychologist inflicts a sequence of flashing lights on a subject. The psychologist has red, blue and green lights available. How many different ways are there to inflict 9 flashes if two are red, four blue and three green?
3. How many triangles are there using edges and diagonals of an $n$-sided polygon if the vertices of the triangle must be vertices of the polygon?
4. Verify by induction and by a combinatorial argument that

$$
\sum_{k=m}^{n} C(k, m)=C(n+1, m+1)
$$

What does this say about Pascal's triangle?
5. Evaluate

$$
\sum_{k=0}^{n} k^{2} C(n, k)
$$

6. How many ways are there to take four distinguishable balls and put two in one distinguishable urn and 2 in another if
a) the order in which the balls are put in the urns makes a difference;
b) the order does not make a difference.
7. How many integers less than 105 are relatively prime to 105 ?
8. Use the algorithm which generates all permutations of length $n$ of $1,2, \ldots, n$ where no digit can be repeated to derive an algorithm to generate all perrnutations when any digit may be repeated an arbitrary number of times.

## 7. Recurrence Relations

## Student Objectives:

- Have lots of exercise in the elements of recursive thinking (e.g., the recursive paradigm-solve problems by jumping into the middle and working your way out).
- Be familiar with recursive definitions of syntax.
- Be familiar with recursive algorithms.
- Understand what a difference equation is.
- See how difference equations can be used to model practical problems.
- Understand the methods for the solution of linear, constant coefficient equations and first order difference equations.
- Be familiar with some applications of difference equations.
- Be able to use the difference calculus (optional).


## Sample Problems:

1. Compute the first ten terms of the sequence defined by

$$
\text { if } n=1 \text { or } 2 \text { then } \begin{aligned}
f_{n} & =3 \\
\text { else } & f_{n}=f_{n-1}^{2}+f_{n-2} .
\end{aligned}
$$

2. What is $f(1)$ if $f$ is defined by

$$
f(n)= \begin{cases}n-3 & \text { if } n \geq 1000 \\ f(f(n+6)) & \text { if } n<1000\end{cases}
$$

3. Describe the strings of characters defined by
<word> ::= <digit>|<letter><word><letter>
with the standard definitions of digit and letter and where := is read "is defined to be".
4. Consider the sum

$$
1+3+5+7+\ldots+2 n-1
$$

Use the inductive and recursive paradigms to conjecture a closed form expression for this sum.
5. Suppose we add to the usual form of the Towers of Hanoi problem the rule that a disk can only be moved from one peg to an adjacent peg (i.e., you can never move a disk from peg 1 to peg 3 or vice versa). Devise an algorithm for solving this version of the problem. Display solutions when $n=2,3,4$.
6. Display a recursive version of the Euclidean algorithm.
7. Consider the recursion

$$
P_{n}=1+\sum_{k=1}^{n-1} P_{k} \quad n>1, \quad P_{1}=1
$$

Compute several terms. Find the pattern and prove that it is correct.
8. Consumer loans work as follows. The Lender gives the Consumer a certain amount $P$, called the Principal. At the end of each payment period (usually each month) the consumer pays the Lender a fixed amount $p$. This continues for a prearranged number of periods (e.g., 60 months $=5$ years). The value of $p$ is calculated so that, at the end of the time, the Principal and all interest due have been paid off exactly. During each payment period the amount owed by the Consumer increases by $r$, the period interest rate, but it also decreases at the end of the period by $p$. Let $P_{n}$ be the amount owed after the nth payment is made. Find a difference equation and boundary conditions for $P_{n}$.
9. Solve

$$
a_{n+1}=5 a_{n}-6 a_{n-1}, a_{1}=5, a_{2}=7
$$

10. Find the general solution to

$$
a_{n}-3 a_{n-1}+2 a_{n-2}=n 2^{n}
$$

11. Solve

$$
S_{n}=(n-1) / n S_{n-1}-1 /\left[n^{2}(n-1)\right] .
$$

12. In ternary search, $t$ and $u$ are the entries closest to $1 / 3$ and $2 / 3$ of the way through the list. Let the search word be $w$. If $w<t$, then search the first third of the list by ternary search. Similarly, if $t<w<u$ or $u<w$ search, respectively, the middle and last thirds of the list. Write recurrence relations for the worst and average case number of comparisons in ternary search. Also show that, if an appropriate sequence $\left\{n_{i}\right\}$ of list lengths is chosen, then $W_{n}$, the number of comparisons in the worst case is given by

$$
W_{n}=2 \log _{3}(n+1) .
$$

(This problem assumes that students have seen a similar analysis for binary search.)

## 8. Graphs and Digraphs

## Student Objectives:

- Understand the definition of the digraph and its use as the picture of a relation.
- Be able to write the matrix representation of a digraph.
- See many applications of digraphs as natural models for networks in real life, such as systems of roads, pipelines, airline routes.
- Know the definitions: connectedness, completeness, complement.
- Be introduced to path problems (and transitive closure) and Warshall's algorithm.
- Be familiar with undirected graphs, the associated definitions and the classical problems of graph theory-the bridges of Koenigsberg, the four color problem, Kuratowski's theorem.
- Be encouraged to solve interesting problems like Mastermind and Instant Insanity to see the usefulness of graph theory.
- Be using algorithms such as Kruskal's algorithm and Dijkstra's algorithm in solving problems.
- Have the opportunity to see the applications to activity analysis (CPM and PERT).
- Be exposed to depth-first search algorithms and topological sorting.


## Sample Problems:

1. Prove that a connected graph of $n$ nodes contains at least $n-1$ edges.
2. Prove that a digraph is disconnected iff its complement is connected. (The complement of a digraph $D$ is defined by the matrix obtained when in the adjacency matrix of $D$ every 0 is replaced by a 1 , and every 1 by a 0 .)
3. A digraph $D=\langle A, R\rangle$ is complete if for all $a, b \in A$, $\langle a, b\rangle \in R$ implies $\langle b, a\rangle \in R$. With respect to this definition, is it true that $\langle a, a\rangle \in R$ for all $a \in A$, or is it true that $\langle a, a\rangle \notin R$ for all $a \in A$ ?
4. A digraph $D=\langle A, R\rangle$ is a tournament if, for all $\langle a, b\rangle \in A,\langle a, b\rangle \in R$ or $\langle b, a\rangle \in R$ whenever $a \neq$ $b$, but $\langle a, b\rangle \in R$ implies $\langle b, a\rangle \notin R$. How many tournaments are there as a function of $n$, where $n=$ $|A|$ ? Draw all tournaments for $n=3$.
5. Prove (by induction) that every tournament contains a Hamiltonian path.
6. How many digraphs on $n$ nodes are there? How many graphs?
7. Find the shortest path from node 1 to every other node in a specific given digraph.
8. Find the transitive closure of the relation represented by this same digraph.

## 9. Trees

## Student Objectives:

- Know the definition of a tree.
- Be able to find the minimal spanning trees for a given graph.
- See the many applications of trees in search problems, with a complete introduction to binary search trees.
- See how to convert digraphs to trees.
- Know how to use digraph algorithms for cycles and critical path analysis.
- Prove the theorems on trees by induction and rely on recursive algorithms for transversal problems on trees.
- Be familiar with sorting and searching algorithms.
- See rooted trees and Polish notation as an application.


## Sample Problems:

1. Show the equivalence of the following definition of an undirected tree: (a) a connected graph without any circuits; (b) a connected graph that becomes disconnected on the removal of any one edge; (c) a connected graph with its number of edges one less than its number of nodes.
2. Prove that the number of leaves in a binary tree with $n$ internal nodes is at most $n+1$.
3. Prove that for every nonnegative integer $n$ it is possible to construct a binary tree with $n$ leaves in which the outdegree of every internal node is 2 .
4. Find all spanning trees of a specific given graph.
5. Find all simple cycles in a specific given graph.
6. Given the drawing of a scheduling network, find the critical path(s) in this network.

## 10. Algebraic Structures

## Student Objectives:

- Be able to define and recognize unary and binary operations.
- Be able to distinguish whether sets are closed with respect to a given operation.
- Be familiar with a variety of operations on a variety of sets:
arithmetic operations on $\mathbf{N}, \mathbf{P}, \mathbf{Q}, \mathbf{Z}, \mathbf{R}$
set operations on $P(S)$
logical operations on propositions
matrix operations on $2 \times 2$ matrices.
- Be able to understand the general definition of an operation on a set via some unfamiliar rule or a table.
- Be able to decide which of the properties hold:
commutative
associative
existence of identity
existence of inverse for given operations on given sets.
- Recognize semigroups, monoids, groups, and cosets.
- Have an elementary knowledge of finite group codes (need cosets).
- Be familiar with and be able to manipulate boolean algebras with many examples.
- Have a rudimentary knowledge of lattices.
- Be able to apply the ideas of homomorphism and isomorphism.


## Sample Problems:

1. Determine if $(\mathbf{N},+)$ is a group.
2. Determine if ( $Z, \circ$ ) is a semigroup, a monoid, a group, when $a \circ b$ is defined to be $a+b-2$ whenever $a, b \in Z$.
3. For $X=011010$ and $Y=100100$, find the Hamming distance from $X$ to $Y$.
4. A code has a minimum distance of 6. How many errors can it detect? How many errors can it correct?
5. Determine if $\left(Z_{5}, *\right)$ is a group.
6. Let $S=\{a, b, c\}$ and define the operation $\oplus$ by the table below. Determine whether $(S, \oplus)$ is a semigroup.

| $\oplus$ | $a$ | $b$ | $c$ |
| :--- | :--- | :--- | :--- |
| $a$ | $a$ | $c$ | $b$ |
| $b$ | $b$ | $b$ | $a$ |
| $c$ | $b$ | $a$ | $c$ |

7. Prove that in every boolean algebra, $\left[B,+, \circ^{\prime}, 0,1\right]$, $x \circ x=x$ and $x+1=1$ for any $x \in B$.
8. Show that the set of $2 \times 2$ matrices with integer entries is a commutative monoid under matrix addition.
9. Construct a logic network for the the boolean expression

$$
x_{1} \circ x_{2}^{\prime}+\left(x_{1} \circ x_{3}\right)^{\prime}
$$

## 11. Discrete Probability and Descriptive Statistics

## Student Objectives:

- Understand basic axioms, simple theorems of probability.
- Understand conditional probability.
- Understand, and be able to do problems involving the discrete uniform, Bernoulli, binomial, Poisson (optional), hypergeometric and geometric probability distributions and their random variables.
- Understand the goal of random number generation.
- Understand the Law of Large Numbers.
- Have a working knowledge of descriptive statistics: populations vs. samples
simple graphing techniques
calculation and meaning of mean, median and mode
calculation and meaning of standard deviation (calculations may be restricted to ungrouped data).
- Discuss the interpretation of sample data including exploratory data analysis.
- Know the meaning of expected value and variance for random variables.
- Know how to use Chebyshev's inequality.
- Know how to simulate some of the probability models discussed.
- Should be doing problems that demonstrate the relationship between probability and difference equations, especially through classical problems like the gambler's ruin problem.


## Sample Problems:

1. Find the probability that in a random arrangement of $n$ files, we find them to be in alphabetical order.
2. Simulate the gambler's ruin problem assuming that the coin being tossed is fair, that $A$ wins a dollar from $B$ when the coin lands heads up and gives a dollar to $B$ otherwise. You may assume that $A$ begins the game with $\$ 3$ and $B$ begins with $\$ 2$. Determine the average length of a game and determine the frequency with which $A$ wins the game.
3. Solve the gambler's ruin problem using a probability model. Use the situation given in problem two and compare your results here to the results of the simulation above.
4. If scores on an examination are normally distributed with $\mu=500$ and $\sigma=75$, find the probability that a score exceeds 700 . Find the 95 th percentile for the scores. Find the probability that a sample mean of more than 510 is found for a random sample of 100 scores.
5. Given the data below, sketch a stem and leaf display, a frequency histogram, and find the mean, median, mode and sample standard deviation. Determine the 75 th percentile of these data.
6. A prize has been put in $2 \%$ of all Sweeties cereal boxes. Find the probability that the fourth box you open contains the first prize you find. Determine the average number of boxes one needs to open in order to get one prize. Simulate the experiment, also.
7. If $80 \%$ of all programs fail to run on the first try, find the probability that in a group of 100 programs
at least 30 programs run on the first try.
8. Show $b(x ; n, 1-p)=b(n-x ; n, p)$ where $b(x ; n, p)=$ $\binom{n}{x} p^{x}(1-p)^{n-x}$.
9. Completion time on a standardized test is normally distributed with an average of 40 minutes and standard deviation 5 minutes. How much time should be allotted if the examiner wants $95 \%$ of the students to finish the test? What if the examiner wishes to leave 5 minutes for checking for $95 \%$ of the class?

## 12. Algorithmic Linear Algebra

## Student Objectives:

- Understand matrix operations and their properties.
- Be able to determine whether a matrix is invertible, and if so, be able to find the inverse.
- See the relationship of matrices to graphs.
- See the use of matrices in representation of linear systems.
- Be able to use row operations to reduce matrices.
- Be able to determine whether a system of linear equations has a solution, a unique solution or no solution.
- Be able to solve a system of linear equations, if a solution exists.
- Have an understanding of linear inequalities, graphing them in the two variable case.
- Be able to solve linear programming problems using the simplex method (and the graphical method in the two variable case).
- Be able to use matrices to solve Markov chain models.
- Use powers of incidence matrices to study connectivity properties of graphs or digraphs.
- See and use the recursive definition of the determinant of a square matrix.


## Sample Problems:

1. Determine the matrix representation of the undirected graph pictured below. Determine the number
of paths of length 3 from $v_{1}$ to $v_{4}$.

2. A firm packages nut assortments: Fancy \& Deluxe. The Fancy assortment contains 6 oz. cashews, 8 oz . almonds and 10 oz . peanuts. It sells for $\$ 2.40$. The Deluxe assortment contains 12 oz . cashews, 10 oz. almonds and 8 oz . peanuts. It is priced at $\$ 3.60$. The supplier can provide a maximum of 3000 oz. cashews, 3600 oz . almonds and 3200 oz . peanuts. Find the number of boxes of each type that would maximize revenue. Use the simplex method and a graphical method.
3. Determine if each of the systems
(a) $4 x+3 y=7$
(b) $6 x+3 y+7 z=4$
$2 x+6 y=8$
$2 x+5 y+8 z=10$
has one solution, no solution, many solutions. Find all solutions in each case.
4. Let

$$
\begin{array}{ll}
A=\left[\begin{array}{ll}
2 & 1 \\
3 & 4
\end{array}\right] & B=\left[\begin{array}{ll}
1 & 0 \\
0 & 0
\end{array}\right] \\
C=\left[\begin{array}{ccc}
1 & 0 & 1 \\
3 & -1 & 2
\end{array}\right] & D=\left[\begin{array}{lll}
1 & 0 & 1 \\
2 & 1 & 1 \\
0 & 3 & 1
\end{array}\right]
\end{array}
$$

Determine if the following exist. If they do, find them; if not, explain. $A+B, A B, A+D, A^{-1}$, $B^{-1}, D^{-1}, \operatorname{det} A, \operatorname{det} C$.
5. Given a Markov chain with transition matrix $P$, find the steady state probability vecotr. Let

$$
P=\left[\begin{array}{ll}
.3 & .7 \\
.4 & .6
\end{array}\right]
$$

(Here we would give a word problem with this transition matrix.)

## Bibliography

## Textbooks

This is a list of textbooks that might be considered for use in courses of the kind discussed in this report. It is as nearly complete as we could make it, but books of this kind are still appearing often and we may well have overlooked some good older ones. Inclusion in the list thus does not imply endorsement, nor does omission imply the opposite. Likewise, the "notes" are not meant to be definitive in any way, but just remarks that users of this bibliography may find interesting.

1. Arbib, M.; Kfoury, A.; Moll, R. A Basis For Theoretical Computer Science. New York: SpringerVerlag; 1981; ISBN 0-387-90573-1.
An introduction to theoretical computer science. Chapter 4 includes techniques of proving theorems. Contents: Sets, maps, and relations; induction, strings, and languages; counting, recurrence, and trees; switching circuits, proofs, and logic; binary relations, lattices, and infinity; graphs, matrices, and machines.
2. Biggs, Norman L. Discrete Mathematics. New York: Oxford University Press; 1985; ISBN 0-19-853252-0.
A mathematically-sound book with plenty of material for a two-semester course. Three main sections are "Numbers and counting," "Graphs and algorithms," and "Algebraic methods." Although written from the viewpoint of mathematics rather than computer science, it does pay a fair amount of attention to algorithms. Probably a bit too rigorous for freshmen and sophomores. Contents: Graphs, combinatorics, number theory, coding theory, combinatorial optimization, abstract algebra.
3. Bogart, Kenneth. Introductory Combinatorics. Boston: Pitman; 1983; ISBN 0-273-01923-6.
A fairly complete coverage of standard combinatorial topics, but the treatment is essentially non-algorithmic; e.g., there is no algorithm for permutations. Somewhat sophisticated; probably requires a year of calculus for maturity. Contents: Introduction to enumeration; equivalence relations, partitions, and multisets; algebraic counting techniques; graph theory; matching and optimization; combinatorial designs; partially ordered sets.
4. Brualdi, Richard A. Introductory Combinatorics. New York, etc.: North-Holland; 1977; ISBN 0-7204-8610-6.
Sophomore level; calculus prerequisite. Sophisticated, but not much algorithmic flavor. Contents: What is combinatorics? the pigeonhole principle; basic counting principles: permutations and combinations; the binomial coefficients; the inclusion-exclusion principle; recurrence relations; generating functions; systems of distinct representatives; combinatorial designs; introduc-
tion to the theory of graphs; chromatic number, connectivity, and other graphical parameters; optimization problems.
5. Cohen, Daniel I.A. Basic Techniques of Combinatorial Theory. New York: John Wiley and Sons; 1978; ISBN 0-471-03535-1.
Assumes one semester of calculus; not inclined to use proof by induction. Contents: Introduction; binomial coefficients; generating functions; advanced counting numbers; two fundamental principles; permutations; graphs. Appendix on mathematical induction.
6. Dierker, Paul; Voxman, William. Discrete Mathematics. San Diego: Harcourt Brace Jovanovich; 1986; ISBN 0-15-517691-9.
College algebra a prerequisite; primarily for freshmen and sophomores. The theme of algorithms is a unifying thread; otherwise, little independence between chapters, so could be used as a text for one- or two-semester courses. Contents: A first look at algorithms; number systems and modular arithmetic; introduction to graph theory; applications of graph theory; boolean algebra and switching systems; symbolic logic and logic circuits; difference equations; an introduction to enumeration; elementary probability theory; generating functions; introduction to automata and formal languages; appendices on set theory, functions, matrices, and relations.
7. Doerr, Alan; Levasseur, Kenneth. Applied Discrete Structures for Computer Science. Chicago: Science Research Associates; 1985; ISBN 0-574-21755-X.

Aimed at freshman-sophomore computer science majors. Includes applications, some "Pascal notes." Contents: Set theory; combinatorics; logic; more on sets; introduction to matrix algebra; relations; functions; recursion and recurrence relations; graph theory; trees; algebraic systems; more matrix algebra; boolean algebra; monoids and automata; group theory and applications; an introduction to rings and fields.
8. Gersting, Judith. Mathematical Structures for Computer Science. San Francisco: W.H. Freeman and Company; 1982; ISBN 0-7167-1305-5.
A fine text, but emphasis on computer science applications may be too great. An accessible reference on group codes. Contents: How to speak mathematics: basic vocabulary; structures and simulations; boolean algebra and computer logic; algebraic structures; coding theory; finite-state machines; machine design and construction; computability; formal languages.
9. Grimaldi, Ralph P. Discrete and Combinatorial Mathematics. Reading, Mass.: Addison-Wesley; 1985; ISBN 0-201-12590-0.
Intended for sophomores and juniors. Contents: Fundamental principles of counting; enumeration in set theory; relations and functions; languages; finite state machines; relations: the second time around; the system of
integers; the principle of inclusion and exclusion; rings and modular arithmetic; boolean algebra and switching functions; generating functions; recurrence relations; groups, coding theory, and Pólya's method of enumeration; finite fields and combinatorial designs; an introduction to graph theory; trees; optimization and matching.
10. Hillman, Abraham P.; Alexanderson, Gerald L.; Grassl, Richard M. Discrete and Combinatorial Mathematics. New York: Dellen Publishing Company; 1986; ISBN 0-02-354580-1.
Sophomore-junior text. Contents: Sets and relations; algebraic structures; logic; induction; combinatorial principles; digraphs and graphs; groups; polynomials and rational functions; generating functions and recursions; combinatorial analysis of algorithms; introduction to coding; finite state machines and languages.
11. Johnsonbaugh, R. Discrete Mathematics, Revised Edition. New York: Macmillan; 1984; ISBN 0-02-360900-1.
Intended for a one-semester course for freshmen or sophomores. Mainly but not exclusively aimed at computer science students. Emphasizes an algorithmic approach and does a considerable amount of algorithm analysis. Contents: Introduction; counting methods and recurrence relations; graph theory; trees; network models and Petri nets; boolean algebras and combinatorial circuits; automata, grammars, and languages. Appendices on logic and matrices.
12. Kalmanson, Kenneth. An Introduction to Discrete Mathematics and Its Applications. Reading, Mass.: Addison-Wesley; 1986; ISBN 0-201-14947-8.
Intended for freshmen and sophomores. Developed in conjunction with Sloan-funded course at Montclair State College. Contents: Sets, numbers, and algorithms; sets, logic and computer arithmetic; counting; introduction to graph theory; trees and algorithms; directed graphs and networks; applied modern algebra; further topics in counting and recursion; appendix with programs in BASIC.
13. Kolman, Bernard; Busby, Robert C. Discrete Mathematical Structures for Computer Science. Englewood Cliffs, New Jersey: Prentice-Hall; 1984; ISBN 0-13-215418-8.
"There are no formal prerequisites, but the reader is encouraged to consult the Appendix as needed." Intended for a one- or two-semester course for freshmen or sophomore computer science students. Contains relatively few algorithms. Approach on the informal side, with not very many theorems or proofs. Contents: Fundamentals; relations and digraphs; functions; order relations and structures; trees and languages; semigroups and groups; finite-state machines and languages; groups and coding.
14. Korfhage, Robert R. Discrete Computational Structures, Second Edition. New York: Academic Press; 1984; ISBN 0-12-420860-6.
Contents: Entities, properties, and relations; arrays and matrices; graph theory: fundamentals; combina-
torics; trees and hierarchies; graph theory: undirected graphs; graph theory: directed graphs; discrete probability; automata and formal languages; boolean algebras; logic: propositional and predicate calculus; algorithms and programs.
15. Levy, Leon S. Discrete Structures of Computer Science. New York: John Wiley and Sons; 1980; ISBN 0-471-03208-5.
A highly-personal statement on discrete structures for computer science students. The presentation is very sketchy. For a sophomore-junior course; leaps quickly into abstraction and algorithms. Contents: An essay on discrete structures; sets, functions, and relations; directed graphs; algebraic systems; formal systems; trees; programming applications.
16. Lipschutz, Seymour. Discrete Mathematics. New York: McGraw-Hill (Schaum's Outline Series); 1976; ISBN 0-07-037981-5.

Contains an outstanding collection of (easy) worked examples and exercises. Vectors and matrices are regarded as an introductory topic. Contents: Set theory; relations; functions; vectors and matrices; graph theory; planar graphs, colorations, trees; directed graphs, finite-state machines; combinatorial analysis; algebraic systems, formal languages; posets and lattices; proposition calculus; boolean algebra.
17. Lipschutz, Seymour. Essential Computer Mathematics. New York: McGraw-Hill (Schaum's Outline Series); 1982; ISBN 0-07-037990-4.

Again there is an excellent collection of examples and exercises. Includes discussion of representation of numbers and characters, linear algebra, and probability and statistics. Suitable for technical mathematics course for data processing students; not appropriate as a text for the discrete mathematics course proposed by the Committee. Contents: Binary number system; computer codes; computer arithmetic; logic, truth tables; algorithms, flow charts, pseudocode programs; sets and relations; boolean algebra, logic gates; simplification of logic circuits; vectors, matrices, subscripted variables; linear equations; combinatorial analysis; probability; statistics: random variables; graphs, directed graphs, machines.
18. Liu, C.L. Elements of Discrete Mathematics, Second Edition. New York: McGraw-Hill; 1985; ISBN 0-07-038133-X.

A comparatively short, but well-constructed text. Intended for a one-semester course but probably more appropriate for juniors than for freshmen or sophomores, although there is no prerequisite beyond high school algebra. Induction and problem-solving are treated carly. A rather traditional mathematical approach with emphasis on combinatorics, relatively little on algorithms. Contents: Computability and formal languages; permutations, combinations, and discrete probability; relations and functions; graphs and planar graphs; trees and cutsets; finite-state machines; analysis of algorithms; discrete numeric functions and generating functions; recurrence relations and recursive algorithms; groups and rings; boolean algebras.
19. Liu, C.L. Introduction to Applied Combinatorial Mathematics. New York: McGraw-Hill; 1968; ISBN 0-07-038124-0.
This is a good source for recurrence relations, and for Pólya's theory of counting. It also contains introductions to linear and dynamic programming. Contents: Permutations and combinations; generating functions; recurrence relations; the principle of inclusion and exclusion; Pólya's theory of counting; fundamental concepts in the theory of graphs; trees, circuits, and cut-sets; planar and dual graphs; domination, independence, and chromatic numbers; transport networks; matching theory; linear programming; dynamic programming; block designs.
20. Marcus, Marvin. Discrete Mathematics: A Computational Approach Using BASIC. Rockville, Maryland: Computer Science Press; 1983; ISBN 0-914894-38-2.
Interesting approach; elementary. Complemented by a DOS 3.3 16-sector $51 / 4$ " floppy disk, DISCRETE PROGRAMS. Contents: Elementary logic; sets; relations and functions; some important functions; function optimization; induction and combinatorics; introduction to probability; introduction to matrices; solving linear equations; elementary linear programming.
21. Molluzzo, John L.; Buckley, Fred. A First Course in Discrete Mathematics. Belmont, Calif.: Wadsworth Publishing Company; 1986; ISBN 0-534-05310-6.
"...intended for non-mathematically-oriented students ...first or second-year computer science or computer information systems student." Contents: Number systems; sets and logic; combinatorics; probability; relations and functions; vectors and matrices; boolean algebra; graph theory; appendix on Pascal.
22. Mott, Joe L.; Kandel, Abraham; Baker, Theodore P. Discrete Mathematics for Computer Scientists. Reston, Virginia: Reston Publishing Company; 1983; ISBN 0-8359-1372-4.

Sophomore-junior course; programming experience desirable but not essential. For computer science audience (posets are defined on p. 17). Contents: Foundations; elementary combinatorics; recurrence relations; relations and digraphs; graphs; boolean algebras.
23. Norris, Fletcher R. Discrete Structures: An Introduction to Mathematics for Computer Scientists. Englewood Cliffs, New Jersey: Prentice-Hall; 1985; ISBN 0-13-215260-6 (Instructor's Manual; ISBN 215277).

Written for a one-semester course for freshmen and sophomores. College algebra is the prerequisite. Contents: Propositions and logic; sets; boolean algebra; the algebra of switching circuits; functions, recursion, and induction; relations and their graphs; applications of graph theory; discrete counting: an introduction to combinatorics; posets and lattices; appendices on the binary number system and matrices.
24. Pfleeger, Shari Lawrence; Straight, David W. Introduction to Discrete Structures, Revised Edition.

New York: John Wiley and Sons; 1985; ISBN 0-471-80075-9.
Aimed at computer science majors; no college-level prerequisites; theory with applications. Includes some proofs. Contents: Formal systems; functions and relations; boolean algebras; boolean algebra and logic design; lattices and their applications; cardinality and countability; graphs and their use in computing; introduction to formal languages; computability.
25. Polimeni, Albert D.; Straight, H. Joseph. Foundations of Discrete Mathematics. Monterey, Calif.: Brooks Cole Publishing Company; 1985; ISBN 0-534-03612-0.
Intended for sophomores. Prerequisite: one year of college-level mathematics, including a semester of calculus, and an introductory programming course. Pascal used throughout. Contents: Logic; set theory; number theory and mathematical induction; relations; functions; algebraic structures; graph theory.
26. Prather, Ronald P. Discrete Mathematical Structures for Computer Science. Boston: Houghton Mifflin; 1976; ISBN 0-395-20622-7 (Solutions Manual; ISBN 0-395-20623-5).
A solid coverage of all the standard material. Boolean algebras are treated as lattices. Contents: Preliminaries, algebras and algorithms, graphs and digraphs, monoids and machines, lattices and boolean algebras, groups and combinatorics, logic and languages.
27. Prather, Ronald P. Elements of Discrete Mathematics. Boston: Houghton Mifflin; 1986; ISBN 0-395-35165-0 (Solutions Manual; ISBN 0-395-35166-9).
Suitable for a one-term course. "No prior programming experience is needed because a generic pseudocode language is used to phrase algorithms." Developed under a Sloan Foundation pilot project grant. Contents: Intuitive set theory; deductive mathematical logic; discrete number systems; the notion of an algorithm; polynomial algebra; graphs and combinatorics.
28. Roman, Steven. An Introduction to Discrete Mathematics. Philadelphia: Saunders College Publishing; 1986; ISBN 0-03-064019-9.
Could be used for a one- or two-semester course for freshmen or sophomores in mathematics as well as computer science. A careful and not too hurried approach but quite traditionally mathematical with little attention to algorithms. Contents: Sets, functions, and proof techniques; logic and logic circuits; relations on sets; combinatorics-the art of counting; more on combinatorics; an introduction to graph theory.
29. Ross, Kenneth A.; Wright, Charles R.B. Discrete Mathematics. Englewood Cliffs, New Jersey: Prentice-Hall; 1985; ISBN 0-13-21.5286-X.
A large book certainly suitable for a two-semester course for freshmen or sophomores in computer science or mathematics. Considerable attention is paid to algorithms, but the approach is generally that of a mathematician rather than a computer scientist. Contents: Introduction to graphs and trees; sets; elementary logic
and induction; functions and sequences; matrices and other semigroups; counting; more logic and induction; relations; graphs; trees; boolean algebra; algebraic systems.
30. Sahni, Sartaj. Concepts in Discrete Mathematics. Fridley, Minn.: Camelot Publishing Company; 1981; ISBN 0-942450-00-0.
The author says the book is for students of computer science and engineering, with a bias towards the former, and contains needed topics not included in typical calculus and algebra courses. Algorithmic in flavor, but moderately formal. Probably a year course at the sophomore-junior level. Many interesting examples not done elsewhere. Contents: Logic; constructive proofs and mathematical induction; sets; relations; functions, recursion, and computability; analysis of algorithms; recurrence relations; combinatorics and discrete probability; graphs; modern algebra.
31. Sedlock, James T. Mathematics for Computer Studies. Belmont, Calif.: Wadsworth Publishing Company; 1985; ISBN 0-534-04326-7.
Intended as first college mathematics course for computer science majors. Unsophisticated, at the level of finite mathematics, without proofs or rigor. Contents: Introduction; computer-related arithmetic; sets, combinatorics, and probability; computer-related logic; computer-related linear mathematics; selected topics (mathematics of finance, statistics, functions, induction); introduction to advanced topics (graphs and trees, semigroups, finite-state machines, languages and grammars).
32. Skvarcius, Romualdas; Robinson, William. Discrete Mathematics with Computer Science Applications. Menlo Park, Calif.: Benjamin Cummings Publishing Company; 1986; ISBN 0-8053-7044-7.
"...intended audience is freshmen and sophomore students who are taking a concentration in computer science ...". Contents: Introduction to discrete mathematics; logic and sets; relations and functions; combinatorics; undirected graphs; directed graphs; boolean algebra; algebraic systems; machines and computations; probability.
33. Stanat, Donald F.; McAllister, David F. Discrete Mathematics in Computer Science. Englewood Cliffs, New Jersey: Prentice-Hall; 1977; ISBN 0-13-216150-8.

A sophomore-junior level course; students will need some previous exposure to college-level mathematics. The first discrete mathematics text to consider program verification in its coverage of mathematical reasoning. The text is essentially non-algorithmic, but contains a special section on analysis of searching and sorting algorithms. Contents: Mathematical models; mathematical reasoning; sets; binary relations; functions; counting and algorithm analysis; infinite sets; algebras.
34. Tremblay, Jean-Paul; Manohar, Ram. Discrete Mathematical Structures with Applications to Computer Science. New York: McGraw-Hill; 1975;

ISBN 0-07-065142-6.
An interesting feature of this text is that its first hundred pages are devoted to logic. All in all, solid coverage of the standard material. Boolean algebra is treated as a subclass of lattices. The notation for algorithms can become forbidding-see page 266 in particular. Contents: Mathematical logic; set theory; algebraic structures; lattices and boolean algebra; graph theory; introduction to computability theory.
35. Tucker, Alan C. Applied Combinatorics, Second Edition. New York: John Wiley and Sons; 1984; ISBN 0-471-86371-8.
A text suitable for a wide range of audiences, from sophomores to graduate students. Contents: Elements of graph theory; covering circuits and graph coloring; trees and selections; generating functions; recurrence relations; inclusion-exclusion; Pólya's enumeration formula; combinatorial modeling in theoretical computer science; games with graphs; appendix on set theory and logic, mathematical induction, probability, the pigeonhole principle, and Mastermind.

## References

This bibliography lists some of the materials that the planner or instructor of a lower-division discrete mathematics course might want to consult. It includes books that may be too advanced or too specialized to belong in the Bibliography: Textbooks. It also lists reports and journal articles more or less pertinent to the theme.

1. ACM/IEEE Computer Society Joint Task Force. "Computer science program requirements and accreditation." Communications of the ACM, 1984, 27 (4) 330-335.
2. ACM Curriculum Committee on Computer Science. ${ }^{\text {"Curriculum '78: Recommendations for academic }}$ programs in computer science." Communications of the ACM, 1979, 22 (3) 47-166.
3. Aho, A.; Hopcroft, J.; Ullman, J. Data Structures and Algorithms. Reading, Mass.: Addison-Wesley, 1983.

Primarily a book on data structures. Algorithms are presented in Pascal. An accessible reference for analysis of algorithms.
4. Bavel, Zamir. Math Companion for Computer Science. Reston, Virginia: Reston Publishing Company, 1982, ISBN 0-8359-4300-3 or 0-8359-4299-6 (pbk).
A manual, not a text.
5. Beidler, John; Austing, Richard H.; Cassel, Lillian N. "Computing programs in small colleges." Communications of the $A C M, 1985,28$ (6) 605-611.
Summary report of The ACM Small College Task Force; outlines resources, courses, and problems for
small colleges developing degree programs in computing. See especially "The Mathematics Component," p. 610.
6. Bellman, Richard; Cooke, Kenneth; Lockett, Jo Ann. Algorithms, Graphs, and Computers. New York: Academic Press, 1970, ISBN 0-12-084840-6.
7. Berztiss, A.T. Data Structures: Theory and Practice, Second Edition. New York: Academic Press, 1975, ISBN 0-12-093552-X.

Although dated, primarily by its dependence on FORTRAN, can be used as a reference on representation of digraphs by trees, and on critical path analysis.
8. Berztiss, Alfs T. Towards a Rigorous Curriculum for Computer Science. Technical Report 83-5, University of Pittsburgh Department of Computer Science, 1983.
9. Bogart, K.P.; Cordiero, K.; Walsh, M.L. "What is a discrete mathematics course?" SIAM News, 1985, 18(1).
10. Deo, Narsingh. Graph Theory with Applications to Engineering and Computer Science. Englewood Cliffs, New Jersey: Prentice-Hall, 1974, ISBN 0-13-363473-6.
An excellent source for applications of the theory of graphs, both undirected and directed, but somewhat dated. Contains extensive bibliographies (to 1972).
11. Dornhoff, Larry L.; Hohn, Franz E. Applied Modern Algebra. New York: Macmillan Publishing Company, 1978, ISBN 0-02-329980-0.
Deals in great detail with applications of algebra in computer engineering, but the complicated notation makes access to the application studies rather hard.
12. Gibbs, Norman E.; Tucker, Allen B. "Model curriculum for a liberal arts degree in computer science." Communications of the ACM, 1986, 29 (3).
A curriculum developed by computer scientists supported by a grant from the Sloan Foundation. The purpose was to define a rigorous undergraduate major in computer science for liberal arts colleges.
13. Gordon, Sheldon P. "A discrete approach to the calculus." Int. J. Math. Educ. Sci. Technol., 1979, 10 (1) 21-31.

A report on an experiment in using discrete calculus (finite differences and sums in place of derivatives and integrals) as an introduction to computer-based continuous calculus.
14. Gries, David. The Science of Programming. New York: Springer-Verlag, 1981, ISBN 0-387-90641-X.
This book deals with the development of correct programs. The introductory sections on logic achieve thoroughness without becoming intimidating. Apart from these sections there is little relevance to the proposed discrete mathematics course.
15. Harary, Frank. Graph Theory. Reading, Mass.:

Addison-Wesley, 1969, ISBN 0-201-02787-9.
Deals almost exclusively with undirected graphs. Extensive bibliography (to 1968). Heavy emphasis on enumeration. Most exercises require considerable mathematical maturity.
16. Hart, Eric W. "Is discrete mathematics the new math of the eighties?" Mathematics Teacher, 1985, 75 (5) 334-338.
17. Heid, M. Kathleen. "Calculus with muMath: Implications for curriculum reform." The Computing Teacher, 1983, 11 (4) 46-49.
18. Hodgson, Bernard R.; Poland, John. "Revamping the mathematics curriculum: The influence of computers." Notes of the Canadian Mathematical Society, 1983, 15 (8) n.p.

A product of a meeting of a working group of the Canadian Mathematics Education Study Group.
19. IEEE Educational Activities Board, Model Program Committee. The 1983 IEEE Computer Society Model Program in Computer Science and Engineering. New York: The Institute of Electrical and Electronics Engineers, 1983.
A section "Discrete Mathematics" appears on pp. 812. This contains a rather demanding modular outline of topics to be covered, and urges integration of the mathematical theory with computer science and engineering applications.
20. Knuth, Donald E. The Art of Computer Programming, Volume 1: Fundamental Algorithms. Reading, Mass.: Addison-Wesley, 1973, ISBN 0-201-03809-9.

Still the standard reference on the tools for analysis of algorithms.
21. Koffman, E.; Miller, P.; Wardle, C. "Recommended curriculum for CS1, 1984: A report of the ACM curriculum committee task force for CS1." Communications of the ACM, 1985, 27 (10) 998-1001.
22. Koffman, E.; Stemple, D.; Wardle, C. "Recommended curriculum for CS2, 1984-A report of the ACM curriculum committee task force for CS2." Communications of the ACM, 1985, 28 (8) 815-818.
23. Laufer, Henry. Applied Modern Algebra. Boston: Prindle, Weber, and Schmidt, 1984, ISBN 0-87150-702-1.
Sophomore-junior level; intended to follow the description for an applied algebra course in the 1981 MAACUPM recommendations (see item 25 below).
24. Lidl, Rudolf; Pilz, Gunter. Applied Abstract Algebra. New York: Springer-Verlag, 1985, ISBN 0-387-96035-X.
Describes aspects of abstract algebra applicable to discrete mathematics.
25. MAA Committee on the Undergraduate Program
in Mathematics. Recommendations for a General Mathematical Sciences Program. Washington, DC: The Mathematical Association of America, 1981.
26. Manna, Zohar; Waldinger, Richard. The Logical Basis for Computer Programming, Volume 1: Deductive Reasoning. Reading, Mass.: AddisonWesley, 1985, ISBN 0-201-18260.

This book is in two parts. The first is a thorough exposition of logic. In the second, data types are treated as theories. Although the authors see the material as ultimately replacing calculus, it probably could not do so as presented-the departure from established practice is too radical, and the style is too austere for an undergraduate text.
27. Maurer, Stephen B. "The algorithmic way of life is best." The College Mathematics Journal, 1985, 16 (1) 2-5.

This is the lead article for a "forum" in which others contribute their own very diverse points of view on the general theme. The whole exchange occupies pp. 2-21.
28. Mulder, Michael C.; Dalphin, John. "Computer science program requirements and accreditation." Computer, 1984, 17 (4) 30-35.
29. Nijenhuis, Albert; Wilf, Herbert S. Combinatorial Algorithms. New York: Academic Press, 1975, ISBN 0-12-519250-9.
A study of aspects of computers, algorithms, and mathematics, and of the relations between them, based on an examination of programs in FORTRAN.
30. Preparata, Franco P.; Yeh, Raymond T. Introduction to Discrete Structures for Computer Science and Engineering. Reading, Mass.: Addison-Wesley, 1973, ISBN 0-201-05968-1.
Junior level. Boolean algebras are arrived at via lattices. Solid coverage, but the material on graphs needs to be supplemented.
31. Ralston, A. "Computer science, mathematics, and the undergraduate curricula in both." American Mathematical Monthly, 1981, 88 (7) 472-485.
32. Ralston, A. "The really new college mathematics and its impact on the high school curriculum." In Hirsch, Christian R.; Zweng, Marilyn J. (Eds.), The Secondary School Mathematics Curriculum: 1985 Yearbook. Reston, Virginia: National Council of Teachers of Mathematics, 1985, 29-42, ISBN 0-87353-217-1.
33. Ralston, A. "The first course in computer science needs a mathematical corequisite." Communications of the $A C M, 1984,27$ (10) 1002-1005.
34. Ralston, A.; Shaw, M. "Curriculum '78-Is computer science really that unmathematical?" Com-
munications of the $A C M, 1980,23$ (2) 67-70.
35. Ralston, A.; Young, G.S. (Eds.). The Future of College Mathematics. New York: Springer-Verlag, 1983, ISBN 0-387-90813-7.
Proceedings of the 1982 "Williams Conference" that added impetus to the movement to add more discrete mathematics to the lower-division curriculum. The ideas presented in the book are far from exhausted.
36. Reingold, Edward M.; Nievergelt, Jurg; Deo, Narsingh. Combinatorial Algorithms: Theory and Practice. Englewood Cliffs, New Jersey: Prentice-Hall, 1977, ISBN 0-13-15244-7.
37. Roberts, Fred S. Applied Combinatorics. Englewood Cliffs, New Jersey: Prentice-Hall, 1984, ISBN 0-13-039313-4.
Junior-senior level; wide range of applications; algorithms.
38. Roberts, Fred S. Discrete Mathematical Models with Applications to Social, Biological, and Environmental Models. Englewood Cliffs, New Jersey: PrenticeHall, 1976, ISBN 0-13-214171-X.
39. Shaw, M. (Ed.). The Carnegie-Mellon Curriculum for Undergraduate Computer Science. New York: Springer-Verlag, 1984.
40. Smith, Douglas; Eggen, Maurice; St. Andre, Richard. A Transition to Advanced Mathematics. Monterey, Calif.: Brooks-Cole, 1983, ISBN 0-534-01249-3.
This book is intended to pave the way from calculus to more advanced mathematics. Its core chapters deal with logic, proofs, sets, relations, functions, and cardinality.
41. Solow, Daniel. How To Read and Do Proofs. New York: John Wiley and Sons, 1982, ISBN 0-471-86645-8.

For use as a supplement if the discrete mathematics course emphasizes proof.
42. Wand, Mitchell. Induction, Recursion, and Programming. New York: North Holland, 1980, ISBN 0-444-00322-3.
This text is addressed specifically to computer science students. It is based on the thesis that a program is a mathematical object. Correctness of programs is then established by reasoning about these mathematical objects in an appropriate language. Prior exposure to sets and functions is assumed.
43. Whitehead, Earl Glen, Jr. Combinatorial Algorithms. New York: Courant Institute of Mathematical Sciences, New York University, 1973.
44. Wirth, Niklaus. Algorithms and Data Structures. Englewood Cliffs, New Jersey: Prentice-Hall, 1986, ISBN 0-13-022005-1.

