

A Brief History of the Computing Curriculum at Grinnell College

Notes by Henry M. Walker

Grinnell College first included a computing or computer science course in its curriculum in 1971, when a 1-credit FORTRAN course was offered with Calculus II as a co-requisite. Since that time, there has been a continuing evolution of the computing curriculum. Several significant influences on this development have been the increased use of computing throughout the college, the expansion of available computing equipment, the changing nature of the needs of students, and the maturing of the discipline of computer science itself.

The history of the computing and computer science curricula may be organized into the following periods:

- ▷ Introductory Programming In Context: 1971-1978
- ▷ Computer Studies: 1978-1990
- ▷ The Computer Science Track in the Math Major: 1983-1990
- ▷ Emergence of the Computer Science Major: 1990-1995
- ▷ Redefinition of Computer Science: 1995-Present

As with most historical developments, the motivations, ideas, and experiences at each of these stages strongly influenced the next.

With the number of courses and people involved with this curriculum over this span of years, any thorough description of these periods could extend for many pages and is beyond the scope of this article. Rather, what follows is a brief review of some major themes and directions, in the hopes that such a selected summary will clarify some of the principles and realities of how the computer science curriculum has evolved to its current stage. This history also may provide additional insight for some current discussions.

1. Introductory Programming in Context: 1971-78

The first computing course at the college emphasized programming and applications and appeared in 1971, the same year as the institution of the Freshman Tutorial. Specifically, Math 134 was offered as a 1-credit supplement to Math 133.

Math 134, *Computing Programming Workshop* (1 credit)
An introduction to computer programming, with special emphasis on material related to calculus. Prerequisite: concurrent registration in Math 133 (Calculus II).

This FORTRAN-based course emphasized numerical methods and utilized a remote card-reader and printer connected to a computer at the University of Iowa. In 1974, a more extensive introductory course was offered to a more general audience.

Math 100, *Introduction to Computer Science* (4 credits)
An introduction, first, to computers and their uses, and second, to a programming language, either BASIC or FORTRAN. Includes a study of algorithms and data structures. Instruction in writing programs and using “prepackaged” programs to solve numerical and nonnumerical problems. Prerequisite: none.

Both Math 100 and 134 stressed programming to solve problems, with the Math 100 open to a wider audience and addressing a somewhat broader range of applications.

During this time, Math 134 served as the primary programming course for students taking mathematics courses. In 1975, the audience for Math 134 was broadened considerably, when the co-requisite was expanded to allow Math 131 (Calculus I), Math 128 (Calculus and Probability II), or Math 133 (Calculus II).

In 1976, Math 134 was replaced by Math 125, which allowed a still broader range of mathematics courses to serve as appropriate co-requisites.

Math 125, *Computing Programming Workshop* (1 credit)

An introduction to BASIC programming, with examples drawn from the mathematics course in which the student is concurrently enrolled. Prerequisite: concurrent registration in any mathematics course numbered from 121 through 234 (precalculus through differential equations or mathematical statistics).

Lectures in this course focused on language syntax, examples, and a few standard algorithms (e.g., searching and sorting), while problem sets were tied to specific mathematics courses. The shift to BASIC reflected the College's acquisition of its first computer, a PDP 11/45.

By 1977, the faculty realized a need to clarify the audience, goals, and scope of Math 100 and Math 125. Both courses covered many similar elements of programming. Applications in Math 125 had expanded to many areas of mathematics, well beyond algorithms specifically related to calculus. Since most Grinnell students took mathematics during their undergraduate program, the difference in target audience between Math 100 and 125 was much less clear than several years earlier.

With no prerequisite, Math 100 proceeded at a slower pace than Math 125, and Math 100 discussed some software packages. Thus, Math 100 was offered for 4 credits, while Math 125 was offered for only 1 credit. In practice, many students selected the course on the basis of credits offered rather than educational goals or interests.

To clarify course content and to highlight different perspectives in computing, the computing curriculum was redesigned in 1977 to separate programming and social issues into two courses. First, the prerequisite for introductory programming was dropped completely in 1977, when Math 125 was replaced by Math 101.

Math 101, *Introduction to Computer Programming* (1 or 2 credits)

An introduction to BASIC programming. Problem sets are tailored to individual students' interests and background, with problem sets associated to courses in several departments being available. This course may be repeated once for 1 credit if taken for 1 credit initially. Prerequisite: none.

At the same time, a course on the social issues of computing was introduced to offer a more general perspective.

Math 102, *Computers in Society* (2 credits)

A systematic study of the impact of computers on society using assigned readings and lectures. Discussion topics will be chosen from: individual privacy and technology, human obsolescence, artificial intelligence, effects on education, and computer control of systems such as monetary flow and national defense. Prerequisite or co-requisite: Math 101.

Together, Math 101 and 102 covered the content of Math 100, while covering mathematical algorithms and techniques for students with appropriate technical background. Math 100 and 125, therefore, were dropped.

As the appeal of programming increased, enrollments in Math 101 soared for several years, as shown in the following table which gives the number of students receiving credit for the course:

Year	Number of Students Completing Course		
	Fall	Spring	Total
1977-78	43	55	98
1978-79	52	43	95
1979-80	57	125	182
1980-81	72	80	152
1981-82	66	127	193

With this interest, Math 101 was one of the few courses on campus in which the enrollment for a semester exceeded the course number.

Computer Studies: 1978-1990

By 1978, interest on campus in computing had increased in several ways. Faculty in several departments were using the computer to store and analyze data in various ways. Courses in such fields as economics, psychology, and sociology, introduced students to techniques involving the computer. Some students had an increasing interest in computer languages, algorithms, and other aspects of computer science. In addition, Computer Services actively promoted the increased use of computing across many areas of the college. Such factors motivated interest in coordinating the use of computing within the curriculum as well as a modest expansion in the study of specific topics within computer science.

As a parallel development, during the previous year, the College had instituted a program of Interdisciplinary Concentrations to encourage students to engage in a breadth of study, following a coherent schedule of courses. Thus, the 1977-1978 College Catalog states:

Each recognized concentration includes an organized cluster of courses drawn from several disciplines and related to a common focus of interest. Thus each provides a structured introduction to a broad area of study while including sufficient flexibility to adapt each program to a student's particular focus of interest. Each culminates in an interdisciplinary senior seminar in which students and faculty draw upon their work in the several disciplines. [p. 140]

These themes and interests were combined in 1978, with the establishment of an Interdisciplinary Concentration in Computer Studies, which is described in the 1978-1979 College Catalog as follows:

This program introduces the student to some of the concepts fundamental to computer science and to some of the major areas of computer applications. The computer serves as a common thread for the required courses in computer science, mathematics, physics, and at least one application area. The senior seminar draws on ideas generated in those fields to study an important topic in computer science. [p. 103]

In addition to taking advantage of the existing computing courses and an electronics course, offered by the Physics Department, the following computer science course was introduced:

Math 201, *Programming Language Concepts and Data Structures* (4 credits)
An introduction to the design and manipulation of data structures, such as tags, linked lists, stacks, and hash tables. A study of the structure and characteristics of programming languages through a comparison of BASIC with one of more other languages, such as FORTRAN, SNOBOL, PASCAL, or Assembly Language. Prerequisite: Math 133 or 128 and two credits of Math 101.

In retrospect, Math 201 began a trend that would be repeated frequently, in which a single course was severely overloaded in the attempt to provide students with some understanding of much of computer science (after Math 101) within the constraint of a single semester. With some luck and much determination, an instructor could select and restrict topics adequately to create a coherent, manageable course.

As a second precedent, the new Computer Studies Concentration also was established assuming that the Mathematics faculty would agree to staff the Senior Seminar as an overload as necessary. While there was some hope that other staffing might be found from time to time, in practice overloads by Mathematics faculty were mandatory in order to offer the concentration at all and to increase the computer science courses available.

Viewed from a historical perspective, Math 201 was successful in providing some expansion of the computing curriculum and in introducing students to some important topics. With the wealth of material appropriate for that course, however, it was only a matter of time before Math 201 was identified as being hopelessly overloaded.

For the most part, the Computer Studies Concentration accomplished its intended purposes of allowing a partial expansion of the computing curriculum and encouraging the coordination of computing courses at the College. In addition, external funding for interdisciplinary concentrations allowed a significant expansion of the library holdings in computing. For the next several years, small adjustments were made in the counting of specific applications courses within the Computer Studies Concentration.

The main structure of the computer science courses remained unchanged until 1983. During that year, the Mathematics Department redesigned its lowest-level courses to emphasize problem-solving. Specifically, the precalculus course was changed from *Algebra and Trigonometry* to *Problem Solving and Precalculus*. Similarly, as the difficulties students had in beginning programming often were due to problem-solving abilities, the following introductory course was designed:

Math 103, *Problem Solving and Computing* (4 credits)
An introduction to the nature of problem solving. Topics will include readings about problem-solving techniques, an introduction to a programming language, and – most importantly – analyzing and solving problems. The problem-solving techniques will be of a general nature and will apply to problems in a wide variety of disciplines. Both the power and the limitations of the computer as a problem-solving tool will be discussed. Prerequisite: None.

At the same time, the faculty observed that computing was being used naturally in many courses throughout the curriculum, as part of the natural evolution of research in those areas. This reduced the need for a separate course to examine computer applications, and Math 102 was dropped as no longer necessary.

Also at that time, many students were entering college with some computing background, and it seemed that these students could start at a level somewhat higher than Math 101. Thus, also in 1983, the Math 101 course was reduced to 1-credit only, and a more substantial computer science course became the more natural starting place for students interested in computing or the Computer Studies Concentration.

Math 203, *Introduction to Computer Science* (4 credits)
An introduction to fundamental computer-science topics. Includes programming in Pascal, machine organization, elementary data structures, and some classical algorithms for sorting and searching data. Prerequisites: Math 101 or 103 (or equivalent); a course in quantitative problem-solving [e.g., math, physics, or chemistry]; sophomore standing.

To further expand the computer science content of the Computer Studies Concentration, a second course was added.

Math 204, *Algorithms and Software Design* (4 credits)
Study of data structures and algorithms with an emphasis on structures that can be applied. Consideration of large application programs with attention to program development and human interface. Prerequisite: Math 203.

With these additions, the computer science requirement for the Computer Studies Concentration now specified Math 203, Math 204, and Physics 220 (Electronics).

The Computer Science Track in the Math Major: 1983-1990

The discussions that led to refining the Computer Studies Concentration in 1983 also prompted the Mathematics Department to expand its upper level offerings in computer science substantially. The specific proposal was introduced by the following rationale:

In recent years, Computer Science has similarly emerged as a rigorous, intellectually stimulating discipline with a firm foundation in theory and methodology. Computer Science education need not be a vocationally oriented, "how to" series of courses; instead, Computer Science can stress concepts, theory, and abstractions which traditionally have been important in a liberal arts curriculum. Therefore the Mathematics Department believes that now is an appropriate time to continue broadening its offerings to include topics in Computer Science in the [mathematics] major. ... [February 11, 1983 proposal, pp. 8-9]

Three courses were introduced at this time:

Math 301, *Data Structures* (4 credits)
Study of structures used to organize data (lists, stacks, queues, trees, graphs) and of the algorithms used to manipulate these structures. Assignments to implement data structures and to use them in computer science and other applications programs. Emphasis is on mathematical principles behind the data structures. Prerequisites: Math 203 and 218 (Combinatorics).

Math 302, *Programming Language Concepts* (4 credits)
A careful study of the concepts underlying programming languages. Examples from Pascal and other high-level languages will be used to illustrate the general concepts. Prerequisite: Math 301.

Math 341, *Automata, Formal Languages, and Computational Complexity* (4 credits)
A formal study of computation devices, their related languages, and the possibility and difficulty of computations. Examples are pushdown automata and Turing machines, context-free languages and recursively enumerable sets, and the halting problem and NP-completeness. Prerequisites: Math 203 and 215 (Linear Algebra and Differential Equations).

At the same time, the Mathematics Major was made more flexible, so that these courses could be counted as part of an [undesigned] track through that major. Since the Computer Studies Concentration was designed as being interdisciplinary, however, these upper level courses were not accepted for that program.

In addition to adding significant computer-science content to the curriculum, it may be worth noting that these specific additions also reflect various national discussions of curriculum that were taking place during the mid-1980s. In particular, Math 301-302 reflect many elements of courses CS7-8 in ACM's Curriculum '78, and all three courses were strongly influenced by informal discussions with a group of faculty that would become the Liberal Arts Computer Science Consortium (LACS).

By 1986, several problems had surfaced regarding Math 203.

The principal weakness of the current Mathematics 203 course is that it tries to cover far too much ground. The universal experience at other colleges and universities is that making students really proficient in Pascal requires a semester (and in many places Pascal programming is a two-semester sequence); we try to do it in one and simultaneously to develop as much as possible of the mathematical background presupposed in more advanced courses in computer science. The result is that neither job is done as well as it might be. [Departmental Memo to the Science Division, March 5, 1986, p. 2]

In effect, then, our proposal is to redistribute the subject matter of several of the courses currently in our curriculum.

Specifically, Math 203 was split into two courses:

Math 151, *Computer Programming in Pascal* (4 credits)

An introduction to computer programming using the high-level, block-structured programming language, Pascal. Topics include algorithm design, coding, testing, debugging, producing documentation, procedural abstraction, and specific examples of data structures. Prerequisite: completion of a Level I course in Problem Solving and Technology Studies [e.g., math, physics, chemistry].

Math 206, *Fundamentals of Computer Science* (4 credits)

An introduction to many of the fundamentals concepts in computer science. Builds upon the programming knowledge from Mathematics 151 to study the design, analysis, and verification of algorithms. Includes a discussion of data abstraction and data structures. Also provides an overview of the field of computer science. Prerequisite: Math 151.

While this change in introductory computer science retained Pascal as the first programming language (in Math 151), the 1986 proposal for Math 206 included a course syllabus based on Modula 2.

With this change in Math 203, the computer science requirement in the Computer Studies Concentration also was modified, so that Math 151 and 204 were required, and students could choose from either Math 206 or Physics 220 *Electronics*. Also, with this change, Math 301 was shifted to “a slightly higher level, more in keeping with its prerequisite chain. The emphasis of the course also would broaden beyond data structures somewhat to include an expanded coverage of the analysis of algorithms.” [Ibid.]

This redesign of the introductory sequence and Math 301 was consistent with many national curricular discussions, and the Rationale of the proposal quoted heavily from a draft “Model Curriculum” by LACS. The Rationale also reiterated the role of Math 103 for preparing students with relatively weak backgrounds for the regular programming course (originally Math 203, now Math 151).

Emergence of the Computer Science Major: 1990-1995

During the 1989-1990 academic year, the Mathematics Department had extensive discussions concerning the position of computer science within the department and the needs of computer science students. Some basic conclusions are outlined in a March 15, 1990, memorandum from the Department to the Dean:

... Computer Science has emerged over the past few years as a separate discipline within the liberal arts and at the College. While both Mathematics and Computer Science may be viewed as part of a broad area sometimes called the Mathematical Sciences, the two fields focus on rather different types of problems, and the two disciplines approach problems in distinct ways.

Further, in talking to people outside the College (including prospective students, potential employers, and publishers), there currently is some confusion about both the mathematics and the computer science curriculum at Grinnell. For example, the course listings for the two disciplines are intermixed. Prospective students interested in mathematics sometimes are put off by the integration of computing courses throughout the mathematics listings; those interested in computer science find it difficult to locate relevant computing courses. The solution to this identification problem is to list mathematics courses separately from those in computer science, all within the same overall departmental listing. [p. 3]

More specifically, these discussions led to several significant changes. One course was added to the curriculum.

Math/CS 207, *Computer Architecture and Operating Systems* (4 credits)
Study of a simple computer's architecture and organization and the principal components of a typical operating system. Topics from architecture: levels of organization, digital logic, instruction execution, information storage, addressing, machine and assembly language. From operating systems: storage management, scheduling, concurrent processing, synchronization, and data protection. Prerequisite: Math/CS 206.

With this new course, the computer science curriculum now covered the full range of core undergraduate computer science courses, as recommended by LACS.

In addition, *Algorithms and Software Design* was upgraded, by changing its prerequisite from Math/CS 151 to Math/CS 206, and the course number was increased correspondingly from 204 to Math/CS 223. This project-based course became the only non-core course in computer science open to majors, a circumstance that continues to the present.

The addition of Math/CS 207 and the upgrading of Math/CS 223 provided a thin, but adequate, base for a designated major in computer science. While this major contained somewhat less computer science than might be desirable, it also required four semesters of mathematics, including Calculus I and II, Linear Algebra and Differential Equations, and Combinatorics. This mathematical requirement allowed some upper level courses, such as Math/CS 301 and 302, to be offered at a more rigorous level than was possible at many other schools. As an aside, it may be of interest to note that the new Computer Science Major was the first new major to be approved at Grinnell College in approximately 25 years.

In addition, the name of the department was changed to the Department of Mathematics and Computer Science to further clarify the position of computer science within the college. Similarly, College Catalog materials were updated to reflect the separate nature of the two disciplines, with mathematics courses being listed separately from those in computer science.

Also, since the term "Computer Studies" was often confused with "Computer Science", since national expectations for computing expertise had risen considerably in recent years, and since computing applications had been integrated into the curriculum in many ways, the College dropped the Computer Studies Concentration.

Following the adoption of the Computer Science Major, work focused upon developing and refining existing courses, and few formal changes were proposed for several years. There were, however, several important experiments in a few courses, most notably in CS 206. There, instructors tried a variety of languages, including Modula 2, Scheme, and Pascal with Units (e.g., Sun Pascal or HP Pascal). Also, a Plus-2 option based on C has sometimes been offered in conjunction with CS 206.

Redefinition of Computer Science: 1995-Present

By 1995, the faculty teaching computer science courses had gained considerable experience with the new major. New perspectives and insights came from observing student difficulties, reading published reports, and talking to colleagues at national meetings.

The most pressing difficulty centered upon a technical restriction at Grinnell College that no more than 48 credits of work in a single department could be counted toward graduation. While this limitation was designed to force students to take a range of courses as part of their liberal arts education, the rule did not anticipate the possibility that two disciplines might be housed in the same department. Thus, after more than a year of discussions among the department, the administration, the Curriculum Committee, and the faculty, the following change was approved, largely resolving this problem.

Resolved, that in applying the College's limit of 48 credits within one department that students may count toward graduation, up to 12 credits of mathematics be exempted for students in computer science. Double majors in the two disciplines would not be allowed.

In anticipation of future discussions and changes, the faculty also changed the names of CS 151 and CS 206 to *Fundamentals of Computer Science I and II*, so that alternative languages could be introduced in these courses without the need to further change the College Catalog.

During the past year, the same faculty have engaged in extensive discussions on many aspects of the computer science curriculum, including appropriate goals, topics, target audiences, and emphases. While a wide range of perspectives are represented, the group seems reasonably close to a consensus on the following points.

General Issues

- The current upper-level, computer-science curriculum appears close to being optimal, given current constraints in staffing. On an absolute scale, however, the major has lost significant ground in the past several years, as the discipline has moved ahead.
- The computer science major currently is much too inflexible, as every course (except CS 103) is required for the major and as every course after CS 206 is offered only in alternate years.
- The current computer science curriculum is deficient in several areas.
 - § Concurrent processes and parallel algorithms should be discussed in considerably greater depth. Some basic issues for concurrent processes are introduced briefly in the current CS 207. There is barely enough time in this course, however, to discuss the concept of deadlock, much less to consider alternative approaches for detection or prevention of deadlock. Parallel algorithms may be mentioned briefly in CS 301, but that course cannot present many such algorithms or approaches without omitting other vital subjects.
 - § There is no natural course in which to cover program verification. At times, this has been taught within CS 302, but such treatment then eliminates other vital language-related material.
 - § CS 223 is the only non-core course offered, and this is offered at the sophomore level. While a Special Topics course was offered once and several group independent projects have been offered from time to time, NO electives in computer science are offered regularly.

Introductory Courses

- The first two semesters of computer science (e.g., CS 151 and CS 206) should introduce students to two different problem-solving paradigms, as recommended recently in several national forums, such as the “Revised Model Curriculum” by LACS. Additional details of the deliberations will be discussed shortly.

Core Courses

- The goals and content of CS 301, CS 302, and CS 341 seem appropriate, both for the current course offerings and for any modified curriculum that might be introduced within the next few years. Of course, goals and content of any computer science course must be updated continually, but the current courses seem conceptually sound.
- While CS 207 may be as good as possible under the current staffing constraints, the course is extremely overloaded and should be divided into two courses. The first course might follow the computer organization course outlined recently in the “Revised Model Curriculum”. The second course might combine a more thorough treatment of operating systems with a greatly expanded study of parallel algorithms.

Electives and Upper Level Courses

- A sophomore- or junior-level course focusing on program verification and analysis should be added.
- Students should be able to choose a few upper-level electives to provide greater depth in some areas of interest, to obtain better background for various career paths, and to prepare themselves more appropriately for graduate school.
- Students should have an experience in an upper-level project as part of the major. This might be an independent project or part of an upper-level version of CS 223. Again, such work is consistent with the recommendations of the “Revised Model Curriculum.”

Beyond these areas of substantial agreement, the computer science faculty has had extensive conversations on the nature of the first courses in computer science for majors. The most attention has been paid to the content and languages for CS 151 and CS 206. For example, for over a year, the faculty have discussed the advisability of changing CS 151 to a Scheme-based course, with CS 206 combining Scheme with either C or C++. Recently, sample class schedules for such courses have been developed and discussed for these approaches. Such discussions are continuing at this time.

In addition to this main thrust for introductory classes, various proposals have been suggested for students with varying levels of programming experience from high school. Discussions of these supplemental courses are still at a preliminary stage.

Due to constraints of time, the faculty has not yet had the chance to discuss at length possible service courses, such as CS 103.

Finally, there has been some preliminary discussion of appropriate staffing levels for the computer science curriculum. One such proposal, based on offering courses over a 2-year cycle, follows:

Desirable Target Schedule for CS Course Offerings

	<i>Year 1</i>		<i>Year 2</i>	
<i>Fall</i>	<i>Spring</i>	<i>Fall</i>	<i>Spring</i>	
Tutorial	103	Tutorial	103	
151	151	151	151	
	151 or variant		151 or variant	
206	206	206	206	
211	243	213	341	
301	302	301	302	
Elective	Elective	Elective	Elective	
	323		323	

This schedule uses the following course numbering for existing or new courses:

- 103: Problem Solving and Computing (or other general literacy course)
- 151: Fundamentals of Computer Science I
- 206: Fundamentals of Computer Science II
- 211: Computer Organization and Architecture (from old 207)
(As in the “Revised Model Curriculum” – with memory allocation and alg.)
- 213: Operating Systems and Parallel Algorithms (from old 207)
- 243: Program Verification and Analysis of Algorithms (new, alternating with 341)
- 301: Algorithms and Abstract Data Types (offered every year)
- 302: Programming Language Concepts (offered every year)
- 341: Theory of Computation
- 323: Algorithms and Software Development (formerly 223)

Such staffing would require 14 sections of computer science to be taught each year. This would involve 2.8 FTE. Current staffing levels are 9.5 sections per year or 1.9 FTE.