economic, and ethical context of the design problem. Thus, the design experience should give a student occasion to deliberate on these aspects of a problem in addition to studying the trade-offs in the technical aspects of a problem.

ABET, a federation of 26 engineering societies, is responsible for the accreditation of engineering programs in the USA. Many of the specific educational guidelines identified in this article are taken from ABET guidelines for accreditation. ABET has recently introduced new accreditation guidelines, Criteria 2000. They do not change the fundamental requirements of an engineering education, but focus on assessing the outcomes of engineering programs more than on specifications of content. In addition, the IEEE Computer Society (q.v.) and the ACM (Association for Computing Machinery—q.v.) have initiated a Task Force to develop a proposal for both computer engineering and computer science curricula for the year 2001.

Bibliography

Yale Patt

EDUCATION IN COMPUTER SCIENCE
For articles on related subjects see COMPUTER LITERACY; COMPUTER SCIENCE; COMPUTER SCIENCE—PH.D. STATISTICS; and EDUCATION IN COMPUTER ENGINEERING.
See also APPENDIX IV

UNITED STATES
Higher Education Programs: History
Academic programs in computing at institutions of higher education began in the mid-1950s under pressure from early users of computing equipment, or from computing center staff deluged with questions about the use of these new devices. Initially, the “educational program” might have consisted only of a short non-credit course given by the computing center staff. Such a course mainly emphasized hardware characteristics, binary arithmetic, and how to program a problem for computer solution (usually in machine or assembly language—see MACHINE AND ASSEMBLY LANGUAGE PROGRAMMING). At times, some of the instructional material was absorbed into an existing course in mathematics or engineering, generally in three or four lectures. However, with the rapid growth of broadly-based university computing installations during the 1960-1965 period, and with the growth of an organized body of knowledge, it became necessary to establish more formal educational programs in computing.

One of the most influential early efforts took place at the University of Michigan, and subsequently at the University of Houston, during the period 1959-1962. These efforts, conducted jointly by the Computing Center and the College of Engineering, were aimed less at establishing computer science as a distinct academic discipline than at the “Use of Computers in Engineering Education” (University of Michigan Study, 1960 and 1961, and University of Houston Study, 1962). At approximately the same time, Stanford University, through the joint efforts of its computing center and department of mathematics, was establishing the discipline of computer science as an optional field of study in the department of mathematics.

These early efforts were capped by the creation of separate departments of computer science. In 1962, Stanford University established a Department of Computer Science in the School of Humanities and Sciences; in the same year, Purdue University created a Department of Computer Science in the Division of Mathematical Sciences. In each case, the bond between the service and academic functions of computing was evident from the fact that one person was both director of the computing center and chairman of the department (George Forsythe at Stanford, Sam Conte at Purdue); this pattern was followed subsequently by other universities. Another pattern established by Stanford and Purdue was that of initially offering only graduate programs in computer science at the master’s and doctoral levels. The thinking at the time was that there could be no well-defined undergraduate program in computer science, and that specialization in computing should start only at the graduate level. (It also reflected the fact that few professors were qualified to teach computing at the time.)

By the mid-1960s, developments in computer science education were proceeding apace. Governmental and quasi-governmental reports made recommendations that spurred the growth of computer science academic programs. Two were of particular importance. The National Academy of Sciences report on “Digital Computer Needs in Universities and Colleges” (Rosser et al., 1966) recommended, among other things, that campuses should “increase as rapidly as possible the number of specialists trained annually as computer specialists and the support of pioneering research into computer systems, computer languages, and specialized equipment.” The President’s Science Advisory Committee report on “Computers in Higher Education” (Pierce et al., 1967) recommended that “the Federal Government expand its support of both
research and education in computer sciences.” These reports helped obtain government and university support for the new discipline.

During the same period, university-sponsored conferences produced reports and books, such as “University Education in Computing Science” (Finerman, 1968), indicating that computer science was truly emerging as an academic discipline and not a short-lived curiosity. Indeed, the “intellectual respectability” of computer science was a controversial issue in the 1960s. Many educators argued that the computer was just a tool, and that a body of study based upon a tool was not a proper academic discipline; others took the position that computer science was not a coherent discipline but rather a collection of bits and pieces from other disciplines; still others felt that computers were not that important and were not proper objects of academic interest. By and large, however, this skepticism was short-lived.

At the same time, computing, mathematics, and engineering professional societies sponsored studies of the curricular effects of the new discipline. Reports of the Mathematical Association of America (Committee on the Undergraduate Program in Mathematics) and the Commission on Engineering Education (Cosine Committee) recommended changes in existing academic programs to insure that students in mathematics and engineering received adequate preparation in computing. This preparation was necessitated by the fact that a growing number of mathematics and engineering majors found themselves working in the computing field soon after graduation. The studies of the Association for Computing Machinery (ACM—q.v.) had the most widespread effect. ACM chartered a Curriculum Committee on Computer Science to recommend necessary academic programs. The subsequent influential report of the Committee, “Curriculum 68” (Atchison et al., 1968), defined for the first time the scope and content of a recommended undergraduate program in computer science. Subsequently, the Committee considerably revised and updated the recommended undergraduate program in its report, “Curriculum 78” (Austing et al., 1979). ACM also chartered a Curriculum Committee on Computer Education for Management. This Committee issued two principal reports on undergraduate and on graduate programs in information systems.


The two societies joined to publish accreditation guidelines in 1983 (Mulder and Dalphin, 1984) and to form a Computer Science Accreditation Commission (CSAC) of the Computing Sciences Accreditation Board (CSAB) in 1984. CSAC was later absorbed by the Accreditation Board for Engineering and Technology (ABET). As of late 1999, 155 programs in computer science had been accredited; for more information, contact CSAC at 111 Market Place, Suite 1050, Baltimore, MD 21202-4012 (email: csac@abet.org).

ACM also chartered Curriculum Committees for master’s level programs in computer science (Magel et al., 1981), undergraduate and graduate degree programs in information systems (Nunamaker et al., 1982; Davis et al., 1997), and related computer science programs in vocational-technical schools, community and junior colleges, and health computing (ACM, 1983). ACM has also published curricular recommendations for secondary school programs in computer science and for teacher certification (ACM, 1985; ACM, 1997) (see also http://www.acm.org/education/curricula.html).

The effect of all these studies, conferences, and reports was a proliferation of academic programs in computer science and engineering. From the early graduate programs have come myriad graduate and undergraduate programs at two-year colleges (associate’s degree), four-year colleges (bachelor’s), five-year colleges (bachelor’s and master’s), and universities (bachelor’s, master’s, and doctoral); these programs are in addition to the numerous computing service courses available to students majoring in other disciplines. Furthermore, there are a multitude of vocational courses given by technical schools. More recently, computing courses have been introduced into the educational programs of most secondary schools.

University Educational Programs

Higher education programs in computing go by different names, such as computer science, computer engineering, computer science and engineering, information science, data processing, and information systems. Each name has also come to denote a particular emphasis and origin. Thus, computer science usually indicates a mathematical and scientific emphasis generally found at universities; information systems usually indicates computing applied to organizational systems generally related to the business administration programs at universities; and data processing
Table 1. Earned degrees in computer and information sciences\(^1\) conferred by institutions of higher education, by level of degree and gender of student: 1970–1971 to 1996–1997.

<table>
<thead>
<tr>
<th>Year</th>
<th>Bachelor's degrees</th>
<th>Master's degrees</th>
<th>Doctoral degrees</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Total</td>
<td>Men</td>
<td>Women</td>
</tr>
<tr>
<td>1970–1971</td>
<td>2,388</td>
<td>2,064</td>
<td>324</td>
</tr>
<tr>
<td>1971–1972</td>
<td>3,402</td>
<td>2,941</td>
<td>461</td>
</tr>
<tr>
<td>1972–1973</td>
<td>4,304</td>
<td>3,664</td>
<td>640</td>
</tr>
<tr>
<td>1973–1974</td>
<td>4,756</td>
<td>3,976</td>
<td>780</td>
</tr>
<tr>
<td>1974–1975</td>
<td>5,033</td>
<td>4,080</td>
<td>953</td>
</tr>
<tr>
<td>1975–1976</td>
<td>5,652</td>
<td>4,534</td>
<td>1,118</td>
</tr>
<tr>
<td>1976–1977</td>
<td>6,407</td>
<td>4,876</td>
<td>1,531</td>
</tr>
<tr>
<td>1977–1978</td>
<td>7,201</td>
<td>5,349</td>
<td>1,852</td>
</tr>
<tr>
<td>1978–1979</td>
<td>8,719</td>
<td>6,272</td>
<td>2,447</td>
</tr>
<tr>
<td>1981–1982</td>
<td>20,267</td>
<td>13,218</td>
<td>7,049</td>
</tr>
<tr>
<td>1982–1983</td>
<td>24,501</td>
<td>15,606</td>
<td>8,904</td>
</tr>
<tr>
<td>1983–1984</td>
<td>32,172</td>
<td>20,246</td>
<td>11,926</td>
</tr>
<tr>
<td>1985–1986</td>
<td>41,889</td>
<td>26,923</td>
<td>14,666</td>
</tr>
<tr>
<td>1986–1987</td>
<td>39,664</td>
<td>25,929</td>
<td>13,735</td>
</tr>
<tr>
<td>1987–1988</td>
<td>34,348</td>
<td>23,347</td>
<td>11,201</td>
</tr>
<tr>
<td>1988–1989</td>
<td>30,637</td>
<td>21,221</td>
<td>9,416</td>
</tr>
<tr>
<td>1989–1990</td>
<td>27,257</td>
<td>19,117</td>
<td>8,140</td>
</tr>
<tr>
<td>1990–1991</td>
<td>25,083</td>
<td>17,726</td>
<td>7,357</td>
</tr>
<tr>
<td>1991–1992</td>
<td>24,557</td>
<td>17,510</td>
<td>7,047</td>
</tr>
<tr>
<td>1992–1993</td>
<td>24,400</td>
<td>17,403</td>
<td>6,977</td>
</tr>
<tr>
<td>1994–1995</td>
<td>24,404</td>
<td>17,463</td>
<td>6,941</td>
</tr>
<tr>
<td>1995–1996</td>
<td>24,098</td>
<td>17,468</td>
<td>6,630</td>
</tr>
<tr>
<td>1996–1997</td>
<td>26,768</td>
<td>18,037</td>
<td>8,731</td>
</tr>
</tbody>
</table>

1 Includes degrees in computer and information sciences, general; information sciences and systems; data processing; computer programming; systems analysis; and other information sciences. The data on Ph.D.s differs somewhat from that in the article COMPUTER SCIENCE–PH.D. STATISTICS because the data in that article comes from a different source (see Kozen and Morris, 1999) than the data above.


usually indicates computing applied to administrative and commercial applications generally taught at two-year colleges. The programs may be housed in a department of computer science, computer engineering, computer science and engineering, computer and information science, or data processing, or given as an option in mathematics, engineering, or business administration.

Table 1 shows the number of higher-level degrees (bachelor’s and beyond) awarded in C&IS from 1970 to 1997. Although the number of master’s and doctor’s degrees has continually increased during this period, the number of bachelor’s degrees peaked in 1985–1986 at almost 42,000 and then declined rapidly to just over 24,000 in 1992–1996 but increased to almost 27,000 in 1997. Although precise data after 1997 is not yet available, it is clear from reports from colleges and universities that the number of bachelor’s degrees is rapidly increasing once again.

The growth of computer science students at institutions of higher education generally parallels corresponding growth in demand in industry. The demand for computer science graduates at all levels declined from the late 1980s to the early 1990s but in the late 1990s demand surged to an all-time high, largely spurred by the growth of the Internet (q.v.).

The demand for two-year college graduates is more limited and there is even less demand for vocational school and high school graduates to fill professional positions, although many technician positions are available, especially for graduates with associate’s degrees. The bachelor’s (and, increasingly, the master’s) degree has rapidly become the entry-level degree for suitable professional positions in industry. In recent years, new Ph.D. graduates have been actively recruited by universities, research organizations, and manufacturers of computer equipment.

Non-University Educational Programs

Computer science educational programs originated at universities and spread downward, from graduate to undergraduate to two-year colleges and then to high
schools. Although subsequent sections of this article deal almost exclusively with university and college programs (undergraduate and graduate), in this section, we briefly discuss other educational programs in computing, specifically those offered by private technical schools or institutes and by two-year colleges. The latter are in some ways similar to those at technical schools and in other ways different, since they may offer a preparation for four-year undergraduate work.

**Technical School Programs**

Private schools for training technicians have been operating for many years. In many fields, they serve a worthwhile function by preparing people for jobs as secretaries, dental technicians, TV repairers, and the like. When the computing industry started expanding rapidly, a large number of private schools began offering educational programs in computing. There are many jobs in industry for which training as a technician is worthwhile, and the technical school graduate should be qualified to assume such jobs.

Unfortunately, some computing institutes intimate that their training will prepare students for well-paying professional jobs in the computing industry, but their graduates often discover too late that most such positions are filled by college graduates. The professional career path in computing, as in most other fields, requires a college education. Nevertheless, because of the severe shortages of the late 1990s, many college graduates from disciplines other than computer science are finding that they can get good entry-level jobs after intensive training at a technical institute.

**Community College Programs**

Two-year community (or junior) colleges have grown rapidly in recent years, both in quantity and in scope of offerings. Forty years ago, the community college was rather rare, usually specializing in such areas as agriculture, forestry, and mining. Today, the community college has become as broadly based and diversified as its university cousin.

The community college serves a twofold purpose. One is to train the student for a position as a technician. For these graduates, the two-year associate's degree is proof of better standards than those usually maintained by a technical school; the degree is also proof of a more well-rounded education. The second purpose of the community college is to serve as a bridge between high school and the four-year college or university course, especially for those students uncertain of their desire or ability to continue with higher education. For these students, the associate's degree may be an intermediate step on the way to a bachelor's degree.

Students terminating after two years and entering industry often suffer the same identity problem as do technical school graduates. Indeed, they are more than technicians, but not the same as college graduates. More often than not, the career paths open to them are technician-oriented. On the other hand, graduates wishing to continue toward a bachelor's degree sometimes find the transition quite difficult. Community college standards are not always the same as university standards; community college courses are not always identical or even similar to corresponding courses at the university.

Some of these difficulties are being addressed; for example, community colleges and universities have been cooperating in facilitating the transfer process by making courses more compatible. Transfer still remains a problem, however, as does the technician versus professional issue. Increasingly, as the "computer profession" evolves and becomes better defined, the broader educational scope of a bachelor's degree becomes a prerequisite for a professional career.

We will not separately detail the usual curricula at two-year colleges. In some cases, these are similar to freshman and sophomore level computing courses at universities. In other cases, the differences are more visible. By and large, university programs are more theoretically oriented, emphasizing both the theoretical underpinnings of computing and scientific or engineering applications. Two-year college programs tend to emphasize the practical aspects and the business applications of computing. The four-year university program allows more time to take courses unrelated to computing, mathematics, and associated technical disciplines. Because of their shorter time span, community college programs are more intensely oriented to courses in programming, business mathematics, accounting, and other technical areas. Accordingly, graduates of community colleges do not possess the broader educational background of graduates of four-year programs.

**Secondary School Programs**

Although ACM has published a complete curriculum for secondary schools (ACM, 1997), most secondary schools do not have the staff or facilities to offer the complete curriculum. However, even before high school some schools offer education in the use of word processing (q.v.), the Internet (q.v.), and spreadsheets (q.v.), as well as courses in Logo (q.v.) and Basic (q.v.). Most high schools offer courses in one or more programming languages such as Basic, Visual Basic, Pascal (q.v.), C (q.v.), and C++ (q.v.). Some high schools offer an advanced placement course using C++ based on the curriculum approved by the
College Board Advanced Placement Committee (see http://www.collegeboard.org).

The Undergraduate Curriculum

The undergraduate program varies from university to university, depending upon such factors as the resources available, the amount of specialization deemed useful, and the interests of the faculty. Even the content of specific courses is, in some cases, quite variable. As noted earlier, the most comprehensive attempts made to date in defining the scope and content of an undergraduate program in computer science have been the works of the ACM Curriculum Committee, Curriculum 68, Curriculum 78, and Computing Curricula—that is, Report of the Joint (ACM/IEEE) Curriculum Task Force. In particular, the 1968 report had a profound effect on shaping the direction of education in the then still emerging discipline of computer science.

The program prescribed in Curriculum 68 reflected the viewpoint of those advocating a strong specialization in computing at the undergraduate level; as such, it follows the traditional pattern of most scientific and engineering undergraduate programs. The large component of computer and mathematics courses recommended (between one-half and two-thirds of the total undergraduate course load) plus technical electives in computer-related disciplines, leaves little room for non-technical subjects in the humanities and the social sciences within the normal four-year program.

Curriculum 78 revised the recommendations for the undergraduate program, reflecting the significant developments that had occurred within computer science education during the intervening decade. Curriculum 78 provides somewhat greater flexibility than Curriculum 68 in the content of courses, emphasizing the objectives of the undergraduate program and the subject matter to be covered. Aside from the proposed curriculum itself, the report discusses such topics as service courses, continuing education, computing facilities, and staff.

Curriculum 78 proposed the following requirements for computer science majors:

- a core of eight computer courses which would be taken by all majors
- four elective courses chosen from a group of 10 advanced courses described in the report
- five mathematics courses (calculus, mathematical analysis 1 and 2, linear algebra, discrete structures, probability and statistics).

Curriculum 78 has been criticized because of its reduced number of mathematics courses and the fact that those mathematics courses that are required are not prerequisite to the computer courses—and therefore are not as integral a part of the prerequisite structure as in Curriculum 68 (see, for example, Ralston and Shaw, 1980).

In 1984, an ACM Task Force revised the first two courses in the curriculum, CS1 and CS2, providing an increased emphasis on problem solving, structured design, and software engineering (q.v.) (Koffman et al., 1984, 1985). Subsequently, a model curriculum for a liberal arts bachelor's degree in computer science was published (Gibbs and Tucker, 1986) that consists of three introductory courses (CS1, CS2, and discrete mathematics) followed by four core courses in computer science.

JOINT CURRICULUM TASK FORCE

In the spring of 1988, ACM and the IEEE Computer Society formed a joint curriculum task force whose charter was to present recommendations for the design and implementation of undergraduate curricula in the discipline of computing. A motivation for this effort was the recognition that, despite strong and fundamental differences among institutions that house the departments offering undergraduate programs, these departments share a substantial curriculum in common. Any curriculum recommendations that attempt to speak for the entire discipline must not only identify the shared subject matter, but also suggest ways in which it can serve as the basis for building undergraduate programs in different kinds of institutions.

The task force proceeded in two stages. The first stage report (Denning et al., 1989) focused on defining the field of computer science, proposing a teaching paradigm for computer science that conforms to traditional scientific standards, and giving an example of a three-semester introductory course sequence based on this model and the definition of the field. The report outlines nine fundamental areas of computer science (since expanded to 12 processes (see COMPUTER SCIENCE) and the three basic processes. The nine areas were:

- Algorithms and data structures
- Architecture
- Artificial intelligence and robotics
- Database and information retrieval
- Human–computer communication
- Numerical and symbolic computation
- Operating systems
Programming languages
Software methodology and engineering

The three basic processes and their elements (in parentheses) are:

- Theory (definitions and axioms, theorems, proofs, interpretation of results)
- Abstraction (data collection and hypothesis formation, modeling and prediction, design of experiments, analysis of results)
- Design (requirements, specification, design and implementation, testing and analysis)

The second stage report (Tucker et al., 1991) discusses how to develop a curriculum based on the model of computer science developed in the first stage. It contains the following parts:

- A collection of 55 subject matter modules called knowledge units that comprise the common requirements for all undergraduate programs in the field of computing, thereby insuring breadth of study. Each knowledge unit contains a list of lecture topics, relations to other knowledge units, recommended hours of coverage, and suggested laboratories.
- A collection of advanced and supplementary curriculum material that provides depth of study in several of the subjects.
- A list of 12 recurring concepts that occur throughout the discipline.

Besides the computing requirements, the report discusses requirements in science and mathematics. The mathematics requirements are a minimum of the equivalent of one-half academic year of mathematics courses including discrete mathematics, calculus, and at least one of the following subjects: probability, linear algebra, advanced discrete mathematics, and mathematical logic.

Rather than provide a single, definitive curriculum for all programs in computing, the report discusses how to develop curricula which incorporate all the components above and how to map the knowledge units into courses. It also describes the role of laboratories in the curriculum. The appendix to the report describes eight sample curricula which differ in their emphasis and assumed institutional constraints.

A major contribution of this report was to show that there are different approaches to computer science education and that there are many good ways of packaging topics. Also it showed that there are many topics, such as social and professional issues, that are essential components of a computing education.

**INFORMATION SYSTEMS CURRICULA**
The ACM also chartered a curriculum committee on computer education for management that has developed curricula for undergraduate and graduate programs in information systems (q.v.). Students in these programs learn how to apply computer technology to meet the information needs of an organization. The first graduate report was published in 1972 (Ashenhurst, 1972), and the first undergraduate report was published in 1973 (Couger, 1973).

This committee was superseded by a curriculum committee on information systems that published its recommendations in the report *Information Systems Curriculum Recommendations for the 80s: Undergraduate and Graduate Programs* (Nunamaker et al., 1982). The committee updated the curriculum and its requirements, stressing the inclusion of the American Assembly of Collegiate Schools of Business common body of knowledge as a major component of the program and introducing a management information systems (MIS—q.v.) policy course as a capstone to the program. There are eight required information systems courses for the undergraduate student, and 10 for the graduate student.

### Table 2. Presentation areas and codes for IS '97.

<table>
<thead>
<tr>
<th>Levels</th>
<th>Presentation areas</th>
<th>Courses</th>
</tr>
</thead>
<tbody>
<tr>
<td>1—General IS Fundamentals</td>
<td>Information Systems Theory and Practice</td>
<td>- Fundamentals of IS</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2—Major and Minor Information Technology</td>
<td>Information Systems Theory and Practice</td>
<td>- Information Systems Theory and Practice</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3—Major Information Systems Development</td>
<td>Information System Development and Management Processes</td>
<td>- Physical Design and Implementation with a DBMS</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The most recently published information systems curriculum is IS '97 (Davis et al., 1997). The architecture of the information systems curriculum at the highest level consists of five curriculum presentation areas: IS fundamentals; information systems theory and practice; information technology (q.v.); information systems development; and information systems deployment and management processes. These five presentation areas consist of 10 courses (Table 2). The curriculum gives course descriptions and resource recommendations for the IS degree program. The details in the appendices provide the basis for customizing courses while maintaining the coverage defined by the curriculum.

**Courses for Non-Majors**

Many undergraduate courses in computer science attract not only the majors in computer science (or information systems) but also students majoring in other disciplines who complete a minor in computer science. For those who do less, the introductory course in programming is still a popular option, especially for students majoring in mathematics, science, or engineering, but the liberal arts or business student will often instead take a course in computer literacy (q.v.), computers and society (see SOCIETY, COMPUTERS IN), or management information systems (q.v.) that emphasizes the development of computer literacy through the use of microcomputer packages rather than through actual computer programming.

**Graduate Curricula in Computer Science**

Graduate programs in computer science preceded the introduction of undergraduate programs, the earliest programs appearing in the early 1960s. Although concentrating on undergraduate computer science, *Curriculum '68* also provided recommendations for master's programs.

In 1981, the ACM Curriculum Committee on Computer Science published recommendations for master's programs (Magel et al., 1981). This report recognizes the emergence of two kinds of programs with different goals: academic programs designed to prepare students for Ph.D. study, and professional programs designed to prepare students for business and industry. However, the committee rejected the idea of a purely terminal program and believes that all programs should make it possible for students to study beyond the master's level.

Although early master's programs in computer science did not require a bachelor's degree in computer science or even substantial prior study in the field, students entering a master's program now should have a B.S. in computer science or at least the equivalent of the material included in CS1 through CS8 of *Curriculum '78*, and mathematics through calculus, linear algebra, discrete structures, and one course in probability and statistics. Maturity in both abstract reasoning and the use of models, as well as one or more years of practical experience in computer science, are desirable.

According to this report, the master's program should provide both breadth in several areas and depth in a few. In addition, it should allow a degree of flexibility to address individual needs. The typical program will consist of 30 to 36 semester hours of courses in programming languages; operating systems and computer architecture; theoretical computer science; and data and file structures. The report lists 30 courses in these four areas, as well as in other areas, with brief descriptions.

Doctoral programs in computer science are intended for students with theoretical or research interests, and most such programs reflect the research interests of the faculty members. In general, courses are similar to those in the master's degree programs. Of course, the doctoral dissertation lies at the heart of the doctoral program. It is the means by which the student demonstrates the capability for making an original contribution to knowledge. This demonstrated capability is the fundamental requirement for the doctorate.

**Summary**

Formal education in computer science and technology dates back only to the early to mid-1960s. Educational programs originated at universities, resulting from the increasing use of computers by students, faculty, and administrators. Today, most colleges and universities offer academic programs in computing, either as a separate discipline or as an option in a related discipline. As can be expected in such a new field, the educational program still has fuzzy edges; at times, it overlaps applied mathematics, electrical engineering, business administration, and other disciplines. Yet, in just a few decades, it has become a visible and influential area of study. Computer science undergraduate programs also provide a service function by offering courses to the student majoring in other disciplines. Usually, these students require some computer courses so that they can better apply computing methods to their fields. Often, however, these students become computing practitioners after graduation.

In earlier days, entry into the computer field was always through some other discipline; there simply were no academic programs in computing. People learned by doing—by using computers, by programming, and by absorbing knowledge in some informal manner. Today, many enter with a degree in computer science, information systems, or related programs.
Furthermore, in earlier days, a university or college degree was not required for many professional positions in the computing organization (especially administrative data processing). Increasingly, prospective employers today require at least a bachelor's degree (in computing or some other field with concentration in computing) to qualify for a professional position. In many cases, a master's degree in computing is preferable. The graduate with a doctorate in computer science experienced difficulty in this shortage, however, ended temporarily in 1991-1992 at universities and at industrial research organizations; finding positions. However, this situation was rapidly reversed in the late 1990s; the demand for doctoral degree holders now far outstrips the supply.

There is now an increasing awareness that the use of the computer stimulates and modifies intellectual processes, and as a result makes it possible for people to expand their intellectual capabilities. This added dimension—the extension of human intellect—must be part of any program in computer science or information systems.

**Bibliography**

Early efforts to bring computing methods into engineering education are described in three related volumes:


There were two principal government-sponsored studies on computing in universities during the mid-1960s. Both gave background information on the use of computers in universities and recommended government financial support for computer education:


Undergraduate and graduate programs


For a bibliography on the subject, see "A Survey of the Literature in Computer Science Education Since Curriculum 68" by Austing, Barnes, and Engel, *Comm. of the ACM,* 20, 13-21 (January 1977). In addition, the quarterly SIGCSE Bulletin of the ACM Special Interest Group on Computer Science Education contains articles of interest on a continuing basis.

The case for less specialized undergraduate programs in computer science is presented in:


The mathematical background of the undergraduate student in computer science is examined in:


The ACM has compiled curriculum recommendations for a variety of educational programs in computer science. The first item below is a paper. The remaining items are booklets that may be ordered from the ACM Order Department, P.O. Box 64145, Baltimore, MD 21264.

There have been several national surveys on computers in higher education and computer manpower conducted by John Hamblen. These report on computing facilities and related expenditures, and computer science and related degree programs. Two of these (which referenced earlier publications) are listed below:


From 1970–1984, Orrin E. Taulbee prepared annual reports giving data on Ph.D. academic programs, which were continued after his death. The latest report is listed below.


Website

Elliott B. Koffman and Aaron Finerman

EUROPE

The teaching of computer science and technology has developed in Europe along more or less the same lines as in the USA, and for the same reasons.

Some of the first computers in Europe were installed or built in universities: Cambridge and Manchester in the UK; Göttingen, Munich, and Darmstadt in Germany; Zurich in Switzerland; and Paris, Grenoble, and Toulouse in France. They were used mainly for research purposes in departments of applied mathematics and sometimes in electrical engineering, but these research projects led to the development of academic programs.

By the mid-1950s, optional courses had started at the universities that had their own computers or could afford to rent a computer mainly for students in mathematics or physics. At that time, a curriculum in computer science was usually divided into three parts—numerical analysis, hardware, and programming.

In 1965, there was one university in England offering a B.Sc. degree in computer science, but in Germany, there were no similar degrees before 1970, despite an extensive teaching program at a number of Hochschulen (schools of engineering). In France, degrees in computer science were given by the Institut de Programmation starting in 1964, although the teaching of computer science started much earlier at the Universities of Grenoble (1956), Toulouse (1957), and Paris (1957). It was also in France that computer science and technology acquired the status of an autonomous scientific discipline very early because of the definition of the word “informatique” by the Académie Française in 1966. Except in English-speaking countries where “computer science” is still the normal designation, informatique or its variants in other languages is the standard name for the discipline in Europe.

During the 1960s, the main disciplines taught were programming, with advanced courses in compilers (q.v.) and operating systems (q.v.), theoretical computer science and numerical analysis (q.v.), often part of the mathematics department. Europe played an important part in the definition of the language Algol 60 (q.v.), its implementation, and its use in teaching. Hardware development generally stayed with electrical engineering departments. There was no attempt to define a “European curriculum” and in each country the national “Computer Science Society” set up a specialized group to discuss the curriculum problem. The discussions in all countries were based on what was known about the developments in US universities, and therefore the ACM “Curriculum 68” had a tremendous influence on European curricula, as did subsequent ACM curricula.

More theoretically minded, Europeans such as Dijkstra and Hoare, had a great influence on the evolution of the disciplines of programming and of software engineering by promoting structured programming (q.v.), clean data structures, and stepwise refinement. Wirth’s Pascal (q.v.) language was widely adopted in computer science teaching.

In 1999, the teaching of informatics in Europe was not very different from computer science instruction in the USA. The titles and contents of the courses are more or less the same, and the differences come mainly from the differences in the administrative organization of education in each country. There was a tendency for computer science studies to take longer in northern European countries than in southern ones, the 1988 European Union Council Directive on a general system for the recognition of higher education diplomas awarded on completion of professional education and training of at least three years duration has made curricula more uniform, in order to be “eurocompatible.”

Most universities offer traditional degrees in computer science (undergraduate, diploma or master’s, and Ph.D.) with differences according to the different
department titles, including engineering, computer science, information systems, communications, and even multimedia (q.v.). Computing, complementary to computer science proper, irrigates all disciplines: a number of degrees in mathematics, chemistry, physics, biology, social sciences, and the humanities have components that include elements of computer science: algorithms, numerical imaging, object-oriented concepts (q.v.) and languages, artificial intelligence (q.v.), neural networks (q.v.), robotics (q.v.), systems, and networking.

An interesting development is the emergence of new disciplines in which computer science plays a strong part: bio-informatics, information and communication systems, mathematics and informatics, etc. In view of the lifelong learning paradigm now in favor with the European Commission, versatility appears to be opening more doors than does the discipline of computer science alone.

Bernard Levrat

ASIA

This article covers computer science education in Southeast Asia (Brunei, Burma, Cambodia, Indonesia, Laos, Malaysia, the Philippines, Singapore, Thailand, and Vietnam), India and Japan. Since developments in India and Japan have tended to be like those in the USA and Europe, this article will focus mainly on Southeast Asia.

Universities in Japan and India started installing computers in the 1950s; in Thailand, Singapore, and Malaysia in the 1960s; and in the remaining countries later than this. Indeed, in 1964, only two computers were installed in Thailand, one in Singapore, and none in all the other countries in Southeast Asia. By the 1990s, all universities in all the countries in Southeast Asia had significant numbers of computers as well as access to the Internet (q.v.).

Early Southeast Asian educators in computer-related areas were educated mainly in the USA, the UK, Canada, and, later, in Japan before, still later, being educated in their own countries. As in the USA, most of the early computer science programs featured numerical analysis (q.v.), both theoretical and applied.

The discipline of computer science, which was established in the USA in the 1960s, had its beginnings in Asia in the 1970s. The first degrees in computer-related subjects were granted in Japan and India in the 1970s, and then later in Thailand, Singapore, Malaysia, the Philippines, and Indonesia.

By the late 1990s, most universities in Japan, India, Indonesia, Malaysia, the Philippines, and Thailand offered degree programs in computer-related areas. A few computer science curricula were available in Vietnam, Brunei, and Burma.


Up-to-date as well as background information on some computer curricula in Asia may be found on Websites such as those listed in Table 1.

ACM Curriculum '68 (Atchison et al., 1968) was the model for most early undergraduate programs at universities in Asia. Mathematically-oriented students prospered in this curriculum, but non-mathematically-oriented students did not. The latter claimed that they needed more applied courses to be able to find jobs. Thus many universities in Southeast Asia modified Curriculum '68 by adding business administration courses and applied courses such as Computer Applications in Banking or Computer Applications in Hotel Administration. Such programs were often then renamed Business Computing or something similar. Most of the Business Computing curricula also required practical training in business establishments with computer or data processing departments. Also, whereas faculties of science or mathematical science in Southeast Asia tended to adopt Curriculum '68, faculties of engineering tended to follow the IEEE Curriculum (Cain, 1977).

Some Southeast Asian universities placed particular emphasis on computer-related curricula by grouping them together into a faculty or a school. For example, a Faculty of Information Technology might include a Department of Computer Hardware, a Department of Computer Software, a Department of Artificial Intelligence and Robotics, a Department of Network Engineering, etc.

Master's and Doctoral level computer science curricula in Southeast Asia are generally patterned after those in the USA, UK, or Japan. Some Japanese-like programs concentrate mainly on research in laboratories where a variety of equipment is readily available. Some curricula modeled on the UK may be more theoretically oriented, while programs modeled on the USA usually include both coursework and research. Several Ph.D. programs in Southeast Asia require the student to publish in a refereed journal in the USA or the UK before
Table 1. Websities for selected Asian universities.

<table>
<thead>
<tr>
<th>Department</th>
<th>University</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computer Section</td>
<td>Maktab Kejuruteraan Jefri, Bolkiah, Brunei</td>
<td><a href="http://www.brunet.bn/php/chongrms/general.htm">http://www.brunet.bn/php/chongrms/general.htm</a></td>
</tr>
<tr>
<td>Computer Science</td>
<td>University of Pune, India</td>
<td><a href="http://cs.unipune.ernet.in">http://cs.unipune.ernet.in</a></td>
</tr>
<tr>
<td>Computer Science</td>
<td>University of Indonesia</td>
<td><a href="http://www.cs.ui.ac.id">http://www.cs.ui.ac.id</a></td>
</tr>
<tr>
<td>Information Science</td>
<td>Japan Advanced Institute of Science and Technology</td>
<td><a href="http://www.jaist.ac.jp/~kouhou/General_ito/organization-e/JOHO-e.html">http://www.jaist.ac.jp/~kouhou/General_ito/organization-e/JOHO-e.html</a></td>
</tr>
<tr>
<td>Informatics and Mathematical Science</td>
<td>Osaka University, Japan</td>
<td><a href="http://www.ics.es.osaka-u.ac.jp/index-e.html">http://www.ics.es.osaka-u.ac.jp/index-e.html</a></td>
</tr>
<tr>
<td>Computer Science and Information Systems</td>
<td>Universiti Teknologi, Malaysia</td>
<td><a href="http://www.fsksm.utm.my">http://www.fsksm.utm.my</a></td>
</tr>
<tr>
<td>Computer Management and Information Technology</td>
<td>Polytechnic University of the Philippines</td>
<td><a href="http://www.geocities.com/CollegePark/Classroom/3110/">http://www.geocities.com/CollegePark/Classroom/3110/</a></td>
</tr>
<tr>
<td>Computer Science</td>
<td>University of the Philippines at Dili man</td>
<td><a href="http://www.engg.upd.edu.ph/cs/">http://www.engg.upd.edu.ph/cs/</a></td>
</tr>
<tr>
<td>Computer Science</td>
<td>Nanyang Technological University, Singapore</td>
<td><a href="http://www.geocities.com/CollegePark/Campus/4293/comp.htm">http://www.geocities.com/CollegePark/Campus/4293/comp.htm</a></td>
</tr>
<tr>
<td>Computational Science</td>
<td>National University of Singapore</td>
<td><a href="http://www.cz3.nus.edu.sg">http://www.cz3.nus.edu.sg</a></td>
</tr>
<tr>
<td>Computer Science</td>
<td>Assumption University, Thailand</td>
<td><a href="http://www.s-t.au.ac.th">http://www.s-t.au.ac.th</a></td>
</tr>
<tr>
<td>Computer Engineering</td>
<td>Chulalongkorn University, Thailand</td>
<td><a href="http://www.cp.eng.chula.ac.th">http://www.cp.eng.chula.ac.th</a></td>
</tr>
<tr>
<td>Computer Sciences</td>
<td>Ho Chi Minh City University, Vietnam</td>
<td><a href="http://www.vnn.vn/huflit/">http://www.vnn.vn/huflit/</a></td>
</tr>
</tbody>
</table>

graduation. This is in contrast to those in the USA and the UK, where publications are not normally required prior to graduation but are expected thereafter.

The graduates of computer curricula in Southeast Asia are employed in a broad range of organizations in such areas as airlines, banking, consulting, government, finance, hotels, insurance, manufacturing, and universities.

Bibliography


Srisakdi Charmonman

EDVAC

The EDVAC (Electronic Discrete Variable Automatic Computer), see Fig. 1, the first stored program computer to be designed, was a direct outgrowth of work on the ENIAC. During the design and construction of the ENIAC in 1944 and 1945, the need for more storage than its 20 10-decimal digit numbers was realized. The experience with acoustic delay lines for radar range measurement led to the concept of recirculating storage of digital information. The group at the Moore School of Electrical Engineering at the University of Pennsylvania started development work on mercury delay lines for such storage, and initiated the design of the EDVAC.

As the first stored program computer, EDVAC instructions that controlled the computational process were stored in the same way that its data was stored. The basic logical ideas are described by von Neumann (1945), and computers based on such designs have come to be known as von Neumann machines (q.v.), even though most historians question whether von Neumann deserves such exclusive credit for the stored program concept. In the spring of 1945, J. Presper Eckert described the mercury delay line (see ULTRASONIC MEMORY) to the author. In answer to the question of how to control the operations, he replied that the instructions would be stored in the delay lines just like numbers. Once he said it, the solution was obvious. There is no doubt that Eckert deserves credit for the delay line memory, and though there is no proof that he first thought of putting instructions in the delay lines, it seems probable that he or John Mauchly thought of it before von Neumann came on