CURRICULUM PROPOSAL FORM
University-Wide Undergraduate Curriculum Committee

UWUCC USE ONLY

Number 25 / A
Action
Date

I. TITLE/AUTHOR OF CHANGE

COURSE/PROGRAM TITLE CO 105, Fundamentals of Computer Science

DEPARTMENT Computer Science

CONTACT PERSON William Oblitey, X4491 (X2524)

II. APPROVALS

John A. Cross
Department Curriculum Committee

Thomas P. Cunningham
Department Chairperson

Richard D. Roberts
College Curriculum Committee

W J Cole
College Dean *

(Not Applicable)
Director of Liberal Studies
(where applicable)

Provost
(where applicable)

* COLLEGE DEAN MUST CONSULT WITH PROVOST BEFORE APPROVING CURRICULUM CHANGES. APPROVAL BY COLLEGE DEAN INDICATES THAT THE PROPOSED CHANGE IS CONSISTENT WITH LONG RANGE PLANNING DOCUMENTS, THAT ALL REQUESTS FOR RESOURCES, MADE AS PART OF THE PROPOSAL, CAN BE MET, AND THAT THE PROPOSAL HAS THE SUPPORT OF THE UNIVERSITY ADMINISTRATION.

III. TIMETABLE

Date Submitted to UWUCC

Semester/Year to be Implemented

Date to be published in Catalog Fall 1990

IV. DESCRIPTION OF CURRICULUM CHANGE

IV. DESCRIPTION OF CURRICULUM CHANGE

1. Catalog Description

CO 105 Fundamentals of Computer Science 3c-01-3sh

Prerequisites: None

This is the first course for Computer Science majors. It is required of all Computer Science freshmen students, and is appropriate for other Natural Sciences and Mathematics students. Topics include the fundamental concepts of computer architecture, algorithm development and analysis, programming languages, software engineering, data organization and representation, and systems software. Hands-on introduction to computer usage with an emphasis on terminology and the underlying connections within the discipline.

COURSE SYLLABUS

I. CATALOG DESCRIPTION

CO 105 Fundamentals of Computer Science 3c-01-3sh

Prerequisites: None

This is the first course for Computer Science majors. It is required of all Computer Science transfer students, and is appropriate for other Natural Sciences and Mathematics students. Topics include the fundamental concepts of computer architecture, algorithm development and analysis, programming languages, software engineering, data organization and representation, and systems software. Hands-on introduction to computer usage with an emphasis on terminology and the underlying connections within the discipline.

II. COURSE OBJECTIVES

1. Students will understand what the fundamental concepts of Computer Science are.

2. Students will develop a foundation from which they can understand the relevance and inter-relationships of Computer Science courses.

3. Students will realize and be motivated to cope with the scope and dynamics of the Computer Science field.

4. Students will develop their abilities to think of problems in terms of the abstraction and refinement process, thus increasing their ability to develop solutions to complex problems by structuring them as intellectually manageable subproblems.

III. COURSE OUTLINE

The following outline of the material to be covered in the course is not the only sequence in which the material may be taught.

A. History of Computing (2 hrs)
   1. Contributions of Pascal, Babbage, Hollerith and others
   2. Computer Generations
   3. Computer Sizes
   4. Significant Events of recent computing history

B. Use of the Computer (5 hrs)
1. Using the terminal
2. Editing text files
3. Introduction to microcomputers
4. System commands
5. Introduction to minicomputers and mainframes
6. Introduction to supercomputers

C. Data Representation (3 hrs)
1. Binary numbers
2. Integer representation (positive and negative)
3. Fractions - binary, hexadecimal
4. Exponents - binary, hexadecimal, excess notation
5. Packed and floating decimal
6. Characters - ASCII, EBCDIC

D. Computer Architecture (5 hrs)
1. CPU organization - registers, op codes
2. Memory organization - addresses, base & bound, paging, segmentation
3. Device operations - interrupts, polling, parity, commands, physical actions
4. Microcoding and machine arithmetic
5. Modern architectures - pipelining, vector and array processors
6. Future architectures - data driven, demand driven and inference machines

E. Problem Solving Skills and Algorithm Development (5 hrs)
1. Top-down approach
2. Common tasks - searching, sorting, merging
3. Techniques - repetition, recursion, divide-and-conquer

F. Programming Languages (5 hrs)
1. Language levels
2. Choosing the language for the task
3. Data types
4. Operators
5. Control statements and conditions
6. Input and output
7. Specifications
8. Programming errors and debugging
9. Translation techniques and theories

G. Software Engineering (4 hrs)
1. Modularization
2. Documentation
3. Design methodologies
4. Software life cycle
5. Programming tools and environments

H. Systems Software (2 hrs)
   1. Operating systems
   2. Compilers, interpreters and assemblers
   3. Linkers and loaders

I. Data Organization (5 hrs)
   1. Data Structures
      a. Consecutive storage - arrays, stacks, queues, trees
      b. Non-consecutive storage - lists, stacks, queues, trees
   2. Files - access and organization
   3. Data base approaches - relational, hierarchical and network

J. Algorithm Analysis (2 hrs)
   1. Measuring time and space
   2. Effects of data structures
   3. Computability

K. Artificial Intelligence (2 hrs)
   1. Heuristics
   2. Applications - robotics, image analysis, spoken language recognition
   3. Search methods

IV. EVALUATION METHODS

Students will be given two intra-term examinations and a final examination. [redacted]

Several projects involving editing and formatting files, algorithm development and programming language understanding will also be given.

Suggested Assignments:

Three assignments involving creating files and formatting text: These files may contain reviews of articles related to topics in the course.

Three assignments involving general problem solving skills: Students will be given problems to solve and will have to use a text/word processor to explain <u>how</u> the problem was solved.

One assignment that requires students to use several operating system commands and includes the compilation and execution of a given program.

One assignment that requires students to develop an algorithm for some task/problem: The algorithm may be expressed in some form of pseudocode.

One assignment involving the understanding of an algorithm expressed in some programming language: Students may be asked to determine the results of the program based on a given set of data.

One assignment that requires students to compare two algorithms that solve the same problem but use different data structures: Students may be asked to evaluate the algorithms for speed, storage use, understandability and/or the appropriate programming language.

One assignment that involves computer game playing and program learning: Students may be asked to devise alternative strategies for a computerized game in playing human opponents.

Collectively, these assignments will count 33% of the student's grade.

V. SUGGESTIONS FOR REQUIRED TEXTBOOKS, SUPPLEMENTAL BOOKS AND READINGS

Behforooz, A. & O.P. Sharma, An Introduction to Computer Science,
Prentice-Hall, 1986.

Brookshear, J.G., Computer Science, An Overview, Benjamin Cummings, 1985.

VI. SPECIAL RESOURCE REQUIREMENTS

Computer hands-on space for microcomputing and mainframe computing, in addition to what is currently available in Stright 107, Mack Data Center, and Johnson Data Center, is currently being resolved with the Director for Academic Computing.

VII. BIBLIOGRAPHY

Behforooz, A. & O.P. Sharma, An Introduction to Computer Science, Prentice-Hall, 1986.

Brookshear, J.G., Computer Science, An Overview, Benjamin Cummings, 1985.

Denning, P. J., Report of the ACM Task Force on the Core of Computer Science, ACM Press, 1988.

COURSE ANALYSIS QUESTIONNAIRE

Section A: Details of the Course

A1. This course will serve the foundational needs of entry level computer science majors. It is intended to be the first course for Computer Science majors, and it is designed to introduce all areas of Computer Science to the students. The course can also meet the needs of Math Ed. students who expect to teach computer science in high schools. For these Math Ed students, an understanding of the major areas of Computer Science will make their high school classes more worthwhile. The course is not proposed for inclusion in the Liberal Studies course list.

A2. Separate sections of CO 110 will be scheduled for majors and non-majors beginning in Fall 1990. Also, in the interest of course title clarity, a separate proposal has been submitted requesting a change of title for CO 110 to Problem Solving and Structured Programming.

A3. CO 105 is taught in the traditional format used in the Computer Science Department - lectures with written assignments. Some assignments will involve the use of the computer.

A4. CO 105 has been taught for three semesters as CO 481 - Foundations of Computer Science.

A5. CO 105 is not a dual-level course.

A6. CO 105 may not be taken for variable credit.

A7. The idea of a computer fundamentals course is relatively new to computer science; most computer science curricula begin with a course that emphasizes programming only. However, many institutions currently offer courses similar to CO 105, usually calling the course "Introduction to Computer Science". Among the leading institutions in promoting a fundamentals course are Marquette University, SUNY at Stony Brook, Moorhead State University and the University of Kansas.

A8. In summer 1989, the NSF sponsored a faculty workshop for teaching this course. Dr. William Oblitey represented the IUP Computer Science Department at this workshop, and our proposal reflects their current thinking. We are following the most recent ACM recommendations for this course as published in Denning, et al., "Report of the ACM Task Force on the Core of Computer Science,", 1988.

Section B: Interdisciplinary Implications

B1. CO 105 will be taught by one instructor.

B2. No additional or corollary courses are needed with CO 105.

B3. The content of CO 105 is not related to any course offered by any other department. However, because a number of other programs require CO 110 as a first course in programming and this proposal causes CO 110 to become a second course in computing for Computer Science majors, separate sections of CO 110 will be scheduled for Computer Science majors and non-majors beginning in Fall, 1990.

B4. Continuing Education students who have little or no Computer Science background should benefit from taking CO 105; however, for some Continuing Education students, CO 101 may be a more appropriate choice. The School of Continuing Education has expressed concerns that CO 105 would have to be given in late afternoon or evening (after 4:00 P.M.) for their students to attend. At least one section of CO 105 can be scheduled to accommodate this. There was also a concern that hands-on experience be part of the class. At least half of the suggested assignments described in the syllabus involve computer usage.

Section C: Implementation

C1. The resources needed to teach CO 105 are currently available.

a. Faculty - All members of the Computer Science faculty are capable of teaching CO 105. In particular, Dr. William Oblitey and Mr. Jim Wolfe have attended workshops for teaching this course.

b. Space - Conventional classroom space, with the addition of projection capability for computer displays, is adequate for offering this course. We currently achieve this capability with "carts" which we take into lectures. Computer hands-on space, which has been a problem, is being resolved with the Director for Academic Computing.

c. Equipment - The hardware and software needs of the course will be met through existing microcomputer, minicomputer, and mainframe equipment that are available.

d. Laboratory Supplies and Other Consumable Goods - These will be handled by the Department.

e. Library Materials - Students may be asked to read current computing periodicals in connection with some assignments. The library currently receives all needed periodicals.

f. Travel Funds - Additional travel funds are not needed.

C2. No resource for CO 105 is funded by a grant.

C3. CO 105 will be offered during every regular semester and during one or more summer school sessions.

C4. Three sections in the Fall semester, two sections in the Spring semester and one or more sections in summer school.

C5. Up to 30 students per section may be accommodated.

C6. No professional society recommends enrollment limits or parameters for CO 105 or courses related to CO 105.

C7. This course will require an additional three credits of Computer Science majors. The Department is requesting a three credit reduction in free electives for Computer Science majors so that there will be no increase in the 124-credit program for the students.

D. Miscellaneous - none included.