DOCUMENT RESUME

ED 081 230 EM 011 409

AUTHOR	Day, Jane M.
TITLE	A Course Which Used Programming to Aid Learning Various Mathematical Concepts.
INSTITUTION	College of Notre Dame, Belmont, Calif.
PUB DATE	Jun 73
NOTE	2p.; Paper presented at the Conference on Computers in the Undergraduate Curricula (Claremont, California, June 18-20, 1973)
EDRS PRICE	MF-$0.65 HC-$3.29
DESCRIPTORS	Algebra; Calculus; *College Mathematics; *Computer Assisted Instruction; Course Descriptions; Creativity; Higher Education; Independent Study; *Mathematical Concepts; *Mathematics Instruction; Problem Solving; *Programing; Self Actualization; Undergraduate Study
IDENTIFIERS	BASIC; BASIC Manual; College of Notre Dame Belmont California; IBM 360/67; Introduction to Computing; Notes for BASIC; Stanford Computation Center

ABSTRACT

A three unit mathematics course entitled Introduction to Computing evaluated the effectiveness of programing as an aid to learning math concepts and to developing student self-reliance. Sixteen students enrolled in the course at the College of Notre Dame in Belmont, California; one terminal was available, connected to the Stanford Computation Center IBM 360/67, and the interactive mode and BASIC were used. Two texts were required--BASIC Manual and Notes for BASIC. Students solved two problems a week in algebra, calculus, or linear algebra for the first eight weeks, providing problem descriptions, flowcharts, program listings, sample outputs, conjectures, and conclusions. In the last half of the course each student completed a major individual project. The teacher lectured on programing concepts and BASIC in the early part of the course and then served mainly as a resource person and consultant for the final 12 weeks of the semester. The course was deemed successful, for the students learned mathematical concepts well, developed creative approaches to independent problem solving, and developed an esprit de corps. An expanded version of the course will be taught to large groups in future years. (PB)

ERIC: You may reproduce and distribute this paper.

Jane M Day
Sept. 5, 1973

A Course Which Used Programming to Aid Learning Various Mathematical Concepts

Jane M. Day*
College of Notre Dame
Belmont, California 94002
(415) 593-7674

This is a report on a three unit mathematics course called Introduction to Computing which was taught at the College of Notre Dame, Belmont, California, in the fall of 1972. We had one terminal, connected to the Stanford Computation Center IBM 360/67. We used the interactive mode almost entirely, and used only BASIC the first half of the course.

My initial motivation in designing the course was to evaluate the effectiveness of programming as an aid to learning various concepts, and as an activity that would cause students to become more self reliant. The students were expected to learn BASIC well and apply it to solve a variety of problems in algebra, calculus and linear algebra.

The course was organized as follows. The tests were the BASIC Manual and Notes for BASIC by Paul Goldstein, both published by the Stanford Computation Center. I prepared 16 problems in the beginning and tried to order them according to difficulty of programming. The list included, in order, problems on graphing, limits of series and functions, evaluating e and π different ways (the round-off error noticed here aroused considerable interest and led to new appreciation of series expansions and trig identities), derivatives, definite integrals, roots, zeros of functions (again, there was much interest when students who had done methodical searching saw from another's output how much more rapidly Newton's method converged), and matrix manipulations. One student observed that the square root algorithm was a special case of Newton's method and then derived a general algorithm for taking nth roots.

Two exercises were due each week, the first eight weeks. For each, students were to include a description of the problem, including identification of the concepts involved, flowchart, program listing, and sample output. Some exercises also asked for conjectures, conclusions, or comparisons based on the output obtained, and I judged this kind of problem especially fruitful. Because some students had not had calculus, I later supplied some alternate exercises on compound interest.

During the last half of the course, each student undertook a major project, selected earlier with my consultation. I encouraged them to select projects about which they would be enthusiastic, and tried to guide each student to one of appropriate difficulty for him. They turned in a description, then a flowchart, and then weekly progress reports. Each student gave a report on his project to the class during the last three weeks. There was no final exam.

I lectured on general programming concepts, BASIC, and covered the exercises for the first four weeks. After that, I lectured briefly on exercises, was available for consulting, scheduled conferences to help each student select a project, and lectured twice on matrix operations in BASIC just before they were needed. There was a lab assistant available for consulting on weekends, which was very helpful, especially in the beginning of the semester.

A student's project was supposed to require at least 60 hours and involved learning new mathematics or deepening understanding of concepts learned before. Most projects did require the expected amount of time and were well planned.

Some projects involved a lot of mathematics explicitly. Two students wrote programs to solve systems of linear equations, one to be used interactively in a tutorial way, and they also explored the literature briefly, thus becoming aware of difficulties such as ill conditioning. Additional projects included using linear algebra and developing new sets of short exercises in BASIC. One student learned FORTRAN by rewriting all the original exercises, and in some cases where round-off error had been considerable before, he compared the accuracy of single and double precision variables. Another student wrote a set of tutorial programs for vector analysis.

Other projects did not involve as much obvious mathematics but required considerable reasoning and organization, and some very creative work was displayed in these. One student used SPSS to analyze questionnaire data from a student course evaluation and three wrote lengthy games--Craps, Roulette and Battleship. A biology major learned PYLON and wrote a tutorial dialogue on kidney function; and one music/math major wrote a program to compose 12 tone music. Her report included a recital. Two of the students writing games began to learn SIMSCRIPT, but decided to use BASIC instead because it was interactive.

For his report, each student prepared multiple copies of his project and its output. In most cases this involved running programs in batch mode.

Evaluating, I am optimistic about the potential of programming assignments as an aid to learning mathematics. The students seemed to grasp a concept more firmly and be much more interested after writing a program using the principles involved. One somewhat unexpected benefit of this course was that it stimulated creativity, which was due mostly I think to each student's being encouraged to choose his own project.

Eleven of the 16 students who enrolled originally had had calculus and some linear algebra, were mathematics majors and generally good students. Seven of these also knew something about computing already but did not know BASIC. The other five had much weaker mathematical backgounds, and two of these dropped the course at mid-semester.

Of the 14 who finished, I think 12 fully achieved the goals I had set for them. They became very skillful with BASIC, clarified and strengthened their understanding of many concepts, developed noticeably in independence, and displayed increasing creativity. Also, this same majority developed a strong esprit de corps, enabling me to function more and more as a resource person. The other two did not do as well. One, who had a weak mathematics background, was extremely interested and worked hard but seemed to lack the organizational ability necessary for skillful programming. The other was a very bright mathematics major who simply reacted negatively to the whole idea of computing--was able to do the work but never became really interested in nor creative with programming.

Overall I think the course was successful. I will teach it again in the spring of 1973 and will allow 30 students this time. I plan some changes: I will prepare different and more open-ended exercises, to encourage students to explore more and to design or find more sophisticated techniques. There will also be more exercises assigned during weeks 3-8, including some to be run in batch and at least one using a plotter at Stanford. I will lecture once specifically on editing techniques, and will take the class to tour the Computation Center early in the semester. The students need that orientation since they may go there later to pick up output, visit Consulting, or use a public terminal.

This particular course is probably of short term value, for I think that computing problems should and will be integrated into mathematics courses wherever appropriate. However, teaching this course has given me a chance to explore just what some of the appropriate places are, and how this college can begin the integration. As an aside, I have recently given assignments in abstract algebra to write a program to compute the Cayley table for the symmetric group on four objects, or compute its subgroups, or test a subgroup for normality. These were alternate exercises, and one student did select and do one such project.

NOTE

* This work was supported in large part by the National Science Foundation, GJ-28765. The author is a participating faculty member from the College of Notre Dame in the Northern California Regional Computing Network, directed by the Stanford Computation Center. She is appreciative of the support received from these sources.