2-1989

The New Generation of Computer Literacy

J. Paul Myers Jr.
Trinity University, pmyers@trinity.edu

The New Generation of Computer Literacy

J. Paul Myers, Jr.

Department of Computer Science
Trinity University
San Antonio, Texas 78284

ABSTRACT

A tremendous mismatch is developing between two of the most critical components of any computer literacy course: the textbooks and the students. We are encountering a "new generation" of students (literally as well as figuratively!) who are much better acquainted with computer usage than their earlier counterparts. Yet many textbooks with increasing emphasis in those same computer tools continue to appear. There are signs of a coming change in that a few authors and publishers apparently are becoming aware of the need for innovations in texts for non-scientists. These textbooks open the door for a new orientation to principles in the teaching of computer literacy.

1. INTRODUCTION

Two of the most common terms used to describe events and circumstances in computer science are "new generation" and "crisis." Possibly in a new field nothing is seen as neutral or dispassioned; instead we proclaim the doom and urgency for change of "crisis" or the salvation of "new generation." This latter is applied primarily to developments in technology (hardware and certain aspects of software); and the former most often alerts us to problems in the management of the technologies and in recruitment at all levels. While, for reasons to be explained, our choice to describe a recent development in curriculum and pedagogy as heralding a "new generation," the title could as well have been "The Crisis in Computer Literacy." This aspect of our curriculum is a mess!

Present students are increasingly sophisticated regarding computer technology. Obviously media, the general cultural attention to computers, and growing-up in the information age have contributed to a familiarity with at least some aspects of computing. And, of course, high schools (even grade schools) are increasing their offerings in computer-related courses. Almost eighty percent of the states officially encourage schools to provide students with some exposure to computers [5]. Indeed, Gilbert and Green report that in 1985 over sixty percent of incoming freshmen had at least a half-year of computer instruction [6].

These trends in pre-college schooling are accelerating. Recent legislation in California, for example, mandates, as of July 1988, that to receive full credentials all teachers must take a fifth year of the program that includes computer education. The bill justifies this requirement on the basis that "... public school pupils need quality instruction and support in the areas of computer education ... for entry into an increasingly technological society" [3]. As an additional measure of this acceleration, An *Electronic Learning* survey conducted in September 1986 found that over fifty percent of the nation's largest education schools required a course in computer literacy for graduation. A similar survey five years earlier showed only five percent [4]! And of course many of the other schools strongly recommend such a course.

2. THE "CRISIS"

A significant impact of these requirements and "strong recommendations" that new high school teachers receive instruction in computer education is that college computer science departments experience steady demand for courses. But another impact is that the students of those very teachers are increasingly well-versed in computer usage. Thus we (in computer science departments) are in the middle -- trying to impart new and meaningful information -- of a feedback loop, potentially leading to instability as each new generation of students is more sophisticated than the previous.

A problem arises in that traditionally computer literacy and the available textbooks focus almost exclusively on applications and technology: an orientation to the use of computers. Courses have consisted mostly of instruction in BASIC and word-processing, some mention of the history of computers and how they work, lots of jargon, and discussion of their impact on society ("computer awareness"); but the primary thrust has been instruction in functional skills [13]. The necessity to teach ever new material to students already trained in the use of computers should have forced a change in this functional orientation to computer literacy.

But it has not. The recent appearance of software packages in word-processing, spreadsheets, database, expert systems, statistics, and design has perpetuated the tool-focused approach to textbook writing, publishing, and teaching for computer literacy courses. Many of these packages have evolved to a state of user-friendliness that most non-technical students can learn them even though they may have no future use for them. And, unfortunately, since many textbooks present a smorgasbord of various packages at a very superficial level, computer literacy is in danger of settling into a course of virtually zero intellectual content and attracting the ridicule of other academic departments (which may, paradoxically, require some of these skills of their students).

Such textbooks and approaches may have been useful at an earlier time; but in the present era of a substantial maturity in the discipline of computer science and of "a nation at risk" from a diminishing emphasis on principles and sound education, such texts are a step in the wrong direction. It is precisely this sort of superficial approach that continues the misunderstanding that other disciplines have toward computer science: a non-discipline with no intellectual content save the design and use of mere tools. To perpetuate this sort of "literacy" or understanding of the field is a major disservice, especially when the rewards of the approach are so few.

And it's hard to imagine that serious computer scientists will be content for long (if ever) to have the discipline reduced to dazzling students with the riches of software packages in the name of "computer literacy." In fact, we may be approaching a time when these skills are seen as remedial in a college environment; in just this spirit Allegheny College has already stopped offering courses in computer literacy [13]. Skills are often taught in the settings (summer jobs or other courses) in which they're used. And software tools have become so easily learned that a student's peers often provide the instruction in a relaxing, non-threatening, informal, and playful atmosphere -- a far better model for education anyway! Many students are already too busy programming (in the "high-level languages" of software packages) to take the time for instruction in those very skills [13].

So to continue the present trend in which the teaching of computer literacy for non-technical students stabilizes into a catalogue of skills, features, and packages would be disastrous. Imagine the boredom on the part of students required to learn yet another software package that accomplishes tasks of no interest to them (otherwise they'd have already learned it) to justify the notion that we're teaching something new. And what faculty member could maintain interest and enthusiasm for such curricular content?

Recently Van Dyke has discouraged such an emphasis on functional competence. She mentions that relatively few jobs require computer competency. And of those that do, the skills are both too specific and too diverse to be anticipated by a single computer course; they are best taught on the job. Moreover, vocational preparation at the college level is inappropriate; liberal education

has always intended not to prepare students for vocations [14]. Even in the customary use of the word, "literacy" can mean merely reading and writing, but also carries the sense of being well-educated.

3. THE "NEW GENERATION"

What then is appropriate and of enduring value in a course taught by computer scientists for non-technical, non-business, well-educated students? Why not a course presenting the principles and intellectual depth of our discipline?

In its scientific, non-data-processing aspects, computer science draws on rich traditions in physics, engineering, and mathematics (primarily the last for our purposes here). Moreover, its mathematical content has enjoyed considerable broad appeal. Relevant concepts have been popularized often: in early books such as Waismann's *Introduction to Mathematical Thinking*, Nagel and Newman's *Gödel's Proof*, and Newman's *The World of Mathematics* to the more recent Pulitzer Prize-winning *Gödel, Escher, Bach* by Hofstadter (a computer scientist).

In areas of logic, foundations, automata capabilities and limitations, and the like, is to be found the soul of computer science. In fact, certain of these topics (algorithms, logic, constructivism, ...) and names (Kleene, Church, Gödel, Turing, ...), that have been of but token interest in most mathematics departments for sixty years are now common parlance in computer science departments. Here is the soul of computing and here is genuine computer literacy.

And the public agrees, in that popularizations of these topics occur frequently and enjoy a significant appeal and readership: the books that we have cited remain in print for edition after edition. Now with the emergence of the "fifth generation" and artificial intelligence, substantial integration is possible for these more "theoretical" concepts with the day-to-day impact of computers. An avenue for this integration, for example, occurs in the similarly popular field of cognitive science. Philosophy is another likely bridge between the substance of computer science and the students' interests and lives. Significant interdisciplinary studies are thus available, which is not the case in the tools-based approach to computer literacy.

Many things of interest and permanence are possible when we focus on principles rather than never-ending litanies of equipment, applications, and types of software packages. We can expand somewhat on a recent sentiment of Harrington:

> ... while students may know how to write code, they have little understanding of the principles or structure of a well-written program. Perhaps, more important, students seem unaware that small knowledge of BASIC programming isn't the same as general computer literacy [9].

We would emphasize principles of computation in the first sentence and would include applications, in general, in the second.

While more and more textbooks appear with emphases on software packages, superficial comparisons of those packages, slick paper, multiple-color illustrations, and gimmicks, there has been a recent appearance of principles-oriented texts. Four notable entries into this field whose authors indicate their possible use in a literacy or introductory course for non-majors are *Principles of Computer Science* by Cullen Schaffer; *Computer Science -- An Overview* by J. Glenn Brookshear; *Algorithmics: The Spirit of Computing* by David Harel; and *Computer Science -- A Modern Introduction* by Les Goldschlager and Andrew Lister. While none of these is perhaps perfect or even fully adequate for our particular purpose of a course for humanities students, we applaud their direction. For example, Harel's subtitle is indeed apt. And Schaffer's prefatory observation is worth citing:

> ... the first goal of most texts is to convey *practical* information, much of which is rather less than earthshaking. Most people appreciate the utility of a keyboard; few care to read about it.
>
> The topics treated here are of practical value, but they have been chosen primarily on grounds of intellectual significance. I have asked myself what ideas we

> computer scientists have reason to be proud of and then attempted to present these at an introductory level [12].

Often the choice of topics may simply be a watered-down version of texts for our own majors (data structures, searching/sorting, etc.) or the text may be just too difficult (but for this, Harel's might be the perfect choice). But the authors' hearts are in the right place: "intellectual significance."

So, as in other areas to which the term is applied, the "new generation" of computer literacy texts promises increase in power, increase in elegance, broader usefulness, and reduction in size.

Regarding this last, anecdotally, while querying publishers for suitable texts at the 1988 ACM/SIGCSE Conference, one particular conversation stands out. A representative wished me luck in identifying this new trend in literacy texts because publishers are incurring ever greater expenses in outdoing their competitors in such areas as length, software rights, paper quality, multiple colors, instructor transparencies, and other eye-catching devices. Now we certainly have no obligation to make things easier for the publishing world; but this remark is indicative of how low our tolerance has become in this enterprise so that content is not even a major point of emphasis among competing publishers! None of the new textbooks listed above use color, nor do they even include more than a couple photographs among them.

A principles-oriented course following one of this new generation of textbooks would provide another benefit to the computer science field: a possible arena for the recruitment of majors. Maybe this shouldn't be a significant priority in designing a computer literacy course; but on the other hand, maybe it should. In any event, students are recruited into fields that interest them; and I am yet to hear of someone majoring in computer science on the basis of a particularly good experience with word-processing or spreadsheet software.

In my previous involvement with the Department of Mathematics and Computer Science at the University of Denver, I taught a course, "The World of Mathematics," that emphasized the cultural aspects of mathematics. With biases toward principles rather than usefulness (of course, one could argue -- I did -- that at this late stage in their education, if students didn't know how to use math yet, to teach them useful math was impossible, so let's have fun instead), I used such texts as Péter's *Playing With Infinity* and Ore's *Graphs and Their Uses*. These were successful; and during those years the courses resulted in some students' changing major to our Department.

Another benefit of the new generation literacy courses, but one whose mention is easily misunderstood, is the improvement of the reputation of computer science among other academic departments. The old style of course is referred to by many students as "a blow-off," "an easy A," "you don't have to attend class," etc. Even worse is that the majority of students rated their text as being of only easy to moderate high-school level of sophistication [11]. These remarks are not lost on other academicians who may resent the vocational level of these courses. Again, our goal is not (necessarily) to please our colleagues or to secure their respect; but their view of such instruction is yet another indicator of the intellectual paucity of the content. Though not (yet) replacing in-class instruction, the non-academic Trinity University Computing Center offers a wide variety of workshops varying from one or two hours to several daily sessions in a large selection of software packages. This seems to be a much more suitable forum for instruction in these tools.

An appropriate course, then, would utilize a reasonable text of the kind beginning to appear. But acknowledging realities such as the apprehension that some students still feel regarding the use of technology [10], there might well be a hands-on portion. This could include a bit of programming for the pleasure of actually implementing some algorithmic thinking; and/or it could include some word-processing (the one application of universal appeal), possibly in a laboratory setting.

Also, the course might include term-papers or panel discussions based on library assignments on the uses of computers, their impact on

society, ethics, etc. since these topics are important in "computer literacy." But the absence of such discussions from a text should not be seen as a critical omission since library research in almost any aspect of these topics is so easy; there is plenty of coverage in magazines and papers on current issues involving computers.

Of computer literacy, Barger has written that as educators have not agreed on its proper content or method of instruction, it "serves as a kind of Rorschach test onto which individuals project their own experiences and values" [1]. Indeed the above is a biased account of what needs to be done given the "crisis" that is so easy to document in computer literacy at this time. But this writer is encouraged to be in a position to not simply rail about present inadequacies and promote a fantasy of a better world, but to be able to acknowledge that some authors and publishers are beginning a new trend according to principles that seem to have considerable interest and permanence. The biases are shared!

REFERENCES

[1] Barger, R.N., "Computer literacy: toward a clearer definition," *T.H.E.J.*, October 1983, 108-112.

[2] Brookshear, J.G., *Computer Science -- An Overview* (2nd ed.), Benjamin/Cummings Publ. Co., Inc., Menlo Park, CA, 1988.

[3] Bruder, Isabelle, "California teachers need extra courses in computer literacy for credential," *Electronic Learning*, April 1988, 16-17.

[4] Bruder, Isabelle, "Ed schools: literacy requirements stagnant, but more offer degrees," *Electronic Learning*, April 1988, 18-19.

[5] *Electronic Learning*'s Seventh Annual Survey of the States, *Electronic Learning*, October 1987, 39-44.

[6] Gilbert, S.W. and Green, K.C., "New computing in higher education," *Change*, May/June 1986, 33-50.

[7] Goldschlager, L. and Lister, A., *Computer Science -- A Modern Introduction*, 2nd ed., Prentice-Hall, Inc., Englewood Cliffs, NJ, 1988.

[8] Harel, D., *Algorithmics: The Spirit of Computing*, Addison-Wesley Publ. Co., Reading, MA, 1987.

[9] Harrington, J.L., "The computer background of incoming freshmen: looking for emerging trends," *SIGCSE Bull.*, 20,1, February 1988, 210-214.

[10] Martin, J.B. and Martin, K.E., "A profile of today's computer literacy students: an update," *SIGCSE Bull.*, 20,1, February 1988, 235-239.

[11] Myers, J.P., Jr. and Buentello, R., "Computer literacy: student preparation and attitudes," in preparation.

[12] Schaffer, C., *Principles of Computer Science*, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1988.

[13] Turner, J.A., "Familiarity with new technology breeds changes in computer-literacy courses," *T.C.E.H.*, July 22, 1987, 9.

[14] Van Dyke, C., "Taking 'computer literacy' literally," *Comm. ACM*, 30,5, May 1987, 366-374.