It would be very nice if I could illustrate the various techniques with small demonstration programs and could conclude with “... and when faced with a program a thousand times as large, you compose it in the same way.” This common educational device, however, would be self-defeating as one of my central themes will be that any two things that differ in some respect by a factor of already a hundred or more, are utterly incomparable.”

Edsger Dijkstra, 1972
“It is characteristic in software engineering that the problems to be solved by advanced practitioners require sustained efforts over months or years from many people, often in the tens or hundreds. This kind of mass problem-solving effort requires a radically different kind of precision and scope in techniques than are required for individual problem solvers.”

Harlan Mills, 1980
“We hear desperate cries for a silver bullet, something to make software costs drop as rapidly as computer hardware costs do. But, as we look to the horizon of a decade hence, we see no silver bullet. There is no single development, in either technology or management technique, which by itself promises even one order of magnitude improvement in productivity, in reliability, in simplicity. ... Not only are there no silver bullets now in view, the very nature of software makes it unlikely that there will be any.”

Fred Brooks, 1986
Structured programming
Formal testing
Object-oriented programming
Design patterns
Unit tests
Test-Driven Development
Agile

Functional programming
Dev ops
“Every Turing machine is reducible into, or in a determined sense is equivalent to, a program written in a language which admits as formation rules only composition and iteration.”

Corrado Böhm and Giuseppi Jacopini, 1966
“Program testing can be used to show the presence of bugs, but never to show their absence.”

Edsger Dijkstra, 1969

“It is well known that you cannot test reliability into a software system.”

Harlan Mills, 1976
Commenting style - Ben Shneiderman, 1980

Variable names – Larry Weissman, 1974

Indenting – Tom Love and Ben Shneiderman, 1977

GOTO - Max Sime, Thomas Green, and John Guest, 1973; Henry Lucas and Robert Kaplan, 1976
“In preparing my retrospective and update of *The Mythical Man-Month*, I was struck by how few of the propositions asserted in it have been critiqued, proven, or disproven by ongoing software engineering research and experience.”

Fred Brooks, 1995
“Coding style wars are a waste of valuable resources, although the confusion caused by Hungarian probably wastes more time.”

Unidentified Windows NT 3.1 developer, ~1992
“Since there was no *mathematical rigor* to inhibit these discussions, some became quite *vehement*.”

Harlan Mills, 1988
So what happened?
“More interesting, however, was the coincidence that all of them had learned to program before they studied programming formally in school. That’s a major change brought about by the personal computer. In my day, I had not even seen a computer before I went to work for IBM in 1956.”

Gerald Weinberg, 1998
“It is practically impossible to teach good programming to students that have had a prior exposure to BASIC: as potential programmers they are mentally mutilated beyond hope of regeneration.”

Edsger Dijkstra, 1975
“Our present programming courses are patterned along those of a “course in French Dictionary.” In such a course we study the dictionary and learn what the meanings of French words are in English (that corresponds to learning what PL/I or Fortran statements do to data). At the completion of such a course in French dictionary we then invite and exhort the graduates to go forth and write French poetry. Of course, the result is that some people can write French poetry and some not, but the skills critical to writing poetry were not learned in the course they just took in French dictionary.”

Harlan Mills, 1972
“If the *precision* and *scope* are not gained in *university education*, it is difficult to acquire them later, no matter how well *motivated* or *adept* a person might be at *individual, intuitive approaches* to problem solving.”

Harlan Mills, 1980
Things you may not know as much about as you would like

- What **programming language** to use
- What to look for in **code reviews**
- What to look for when **hiring**
- How **reliable** your software is
- Is your software about to **fail**
- Is your software **obsolete**
PROGRAMMERS ARE CONTENT USING THEIR OWN EXPERIENCE AS A GUIDE

CHANGE MY MIND
“Most of these attack the central argument that there is no magical solution, and my clear opinion that there cannot be one. Most agree with the arguments in ‘No Silver Bullet,’ but then go on to assert that there is indeed a silver bullet for the software beast, which the author has invented.”

Fred Brooks, 1995
I thought I was the **hardest-working person** on the planet. I thought we were the **hardest-working industry**. That’s what we tell ourselves. It’s all **malarkey**.

I’ve had this front-row seat over the last three years to **greatness**. It’s a **humbling experience**...seeing just what it takes to **actually be that great**.”

Alexis Ohanian, 2018
“Another essential personality factor in programming is at least a small dose of humility. Without humility, a programmer is doomed to the classic pattern of Greek drama: success leading to overconfidence (hubris) leading to blind self-destruction. Sophocles himself could not have invented a better plot (to reveal the inadequacy of our powers) than that of the programmer learning a few simple techniques, feeling that he is an expert, and then being crushed by the irresistible power of the computer.”

Gerald Weinberg, 1971
“In the past twenty-five years a **whole new data processing industry** has exploded into a critical role in business and government. Had this hardware development been spaced out over **125 years**, rather than just **25 years**, a **different history** would have resulted. For example, just imagine the **opportunity** for orderly industrial development with **five human generations** of university curriculum development, education, feedback for the **expansion of useful methodologies** and **pruning of less useful topics**.”

Harlan Mills, 1976
A
Program

A
Programming System
(Interfaces System Integration)

A
Programming Product
(Generalization, Testing, Documentation, Maintenance)

A
Programming Systems Product
“There is no doubt that the **grail of efficiency** leads to **abuse**. Programmers waste enormous amounts of time thinking about, or worrying about, the speed of **noncritical parts** of their programs, and these attempts at efficiency actually have a **strong negative impact** when **debugging** and **maintenance** are considered. We should forget about **small efficiencies**, say about 97% of the time: **premature optimization** is the **root** of all evil.”

Donald Knuth, 1974
Radio Shack® TRS-80 Computer System...

The first complete, low-priced, fully assembled microcomputer!

Radio Shack® TRS-80. Here's the answer for the thousands of people who need a complete, inexpensive, ready-to-use computer without the delay and problems of building one! Designed in the U.S.A. by Radio Shack, the TRS-80 is 100% wired and tested. It's ideal for finances, accounting, education, laboratory, or even games. Use your imagination, or use Radio Shack's expanding line of prepared cassette programs. And Tandy will provide full support in software and accessories. The advanced Z-80 based CPU includes 4K bytes RAM, Radio Shack floating-point Level-1 BASIC stored in 4K ROM, integrated 52-character ASCII keyboard, computer-controlled interface with expansion port, regulated power supply, 12" video monitor that displays 16 lines of 54 characters, battery/AC data recorder, 300-page instructions/programming manual, and a 2-game cassette. Computer, 16xix8x3½". Video display, 16x13xix12x12. For 120VAC. UL listed. Complete system, Shpg. wt., 35 lbs. $599.95

599.95

Save $49.90...Buy the Complete TRS-80 System!

Computer with Power Supply and Built-in Keyboard, 26-1091
12" Video Display, 26-1091
Realistic® CTR-41 Data Recorder, 14-041
Separate Components Price

$545.85

26-1191. 16K RAM Memory Expansion Module...
“Hell is other people.”

Jean-Paul Sartre, 1944
“More recently I discovered why the use of the goto statement has such disastrous effects.

We should do our utmost to shorten the conceptual gap between the static program and the dynamic process, to make the correspondence between the program (spread out in text space) and the process (spread out in time) as trivial as possible.”

Edsger Dijkstra, 1968
“Optimization is the root of all evil.”

Adam Barr, 2019
I think I'll add a cache—

[Moves review slider left]
Microsoft Azure
The cloud for modern business.
“Short cuts makes long delays.”

Peregrin Took, 3018 T.A.
“The next generation of programmers will be much more competent than the first ones. They will have to be. Just as it was easier to get into college in the ‘good old days,’ it was also easier to get by as a programmer in the ‘good old days.’ For this new generation, a programmer will need to be capable of a level of precision and productivity never dreamed of before.”

Harlan Mills, 1978
Thank you for listening.

Any questions?

Thank you for your attention.

Any questions?

Thank you.

Any questions?

Thank you for your attention.

Any questions?

Please feel free to ask any questions.

@AdamDavidBarr