# *FORTRAN Test Programs*

The career of Frances E. (Betty) Snyder Holberton, the lead author of SP 399, *NBS FORTRAN Test Programs* [1], closely tracks the evolution of information technology (IT) for the four decades following World War II. Her work at NBS (1966-1983) and her leadership in preparing this landmark publication are best understood in the context of her earlier career.

Her first work remains perhaps the most notable: she was a member of the first team of programmers (composed of six women) to work on the ENIAC [2,7], the first operational, general purpose, electronic digital computer. "Programming" in those early days did not consist of writing code; rather, it meant setting thousands of switches and connecting dozens of cables so as to route data correctly through the machine. ENIAC was built at the Moore School of Electrical Engineering at the University of Pennsylvania. Completed in late 1945, it is widely regarded as a milestone in the history of computer technology. After five years at the Moore school, Holberton left in 1947 to work at the Eckert-Mauchly Electronic Control Company to participate in the development of the UNIVAC.

Her work on the UNIVAC included the development of the C-10 language, one of the first to use mnemonic (symbolic) instructions (e.g., "a" for add, "b" for bring). She also wrote the first sort-merge software, notable as perhaps the first case of a program whose function was to generate another program—a key idea in the notion of a programming language compiler. Compilers make it possible to express computer programs in high-level languages, such as FORTRAN, C, or Java, instead of the low-level instructions which are directly processed by the computer hardware. A line of FORTRAN, such as

```
ANGLE = ATAN (SGMENT / SQRT(X*X + Y*Y))
```

will typically generate several instructions in machine language (so-called object code); producing this machine code directly would be quite tedious and error-prone.

From the appearance of the line of FORTRAN above, one might reasonably guess that its purpose is to square two quantities, labeled X and Y, add them, take the square root of that sum, divide that quantity into SGMENT, take the arctangent of the quotient and save the result as ANGLE. All well and good, but will this line of code work equally well (or work at all) when processed on different vendors' machines? Note that vendors A and B may well have completely different underlying hardware (i.e., their low-level machine codes are not the same). Furthermore, each has very likely written its own compiler—so the translation process itself from FORTRAN (the so-called source code) to machine code may well be different. Even assuming that the A and B compiler-writers make no unintentional errors, we still need to make sure that they are working toward the same goal. The very notion of a formal language implies strict rules about the spelling and meaning of its expressions [4,6]. For instance, the square root function must be spelled "SQRT," not "SQROOT." Furthermore, even assuming that both compilers accept the standard spelling (i.e., they accept the source code and generate machine code), the problem of semantics remains. For instance, in our example, does the arctangent function return its result in degrees or radians? COBOL and FORTRAN standards committees were established [8] to ensure that just such issues were resolved in a uniform way, and Betty Holberton was an active participant on these committees. As a result, software written according to the standard could be expected to run consistently on a wide variety of machines. Given the cost of developing large software products, this is no small benefit.

By the early 1960's, industry standards for high-level languages such as FORTRAN and COBOL were becoming widely accepted. One problem remained: how to be sure that the standard was being implemented correctly by the various vendors? Put another way: how to measure the validity of the compiler software itself?

Although metrology had always been a core function of NBS, devising measurement methods for information technology nonetheless represented a novel challenge. Previous work had typically addressed the measurement of physical quantities, such as temperature, electrical charge, mechanical strength, composition, etc. But many of the artifacts of information technology are essentially logical structures. True, software is realized on some physical medium, but most often it is the logical behavior of the software that is of greatest interest, not the details of its physical implementation. (Of course, NBS has had a hand in developing tests for the physical media of computers as well.)

One way to determine the validity of a piece of software is simply for a skilled programmer to read it;

indeed, code review is an important part of the software engineering process. Nonetheless, such a subjective procedure could hardly serve as the basis for an "official" determination that a compiler did or did not conform to the language standard. Moreover, there are operational issues: how well trained are the reviewers? How long would it take them to verify a single compiler? There was a need for a largely automated and objective method for checking compiler conformance to the standard.

This was the problem addressed by Holberton and her colleague Elizabeth Parker in *NBS FORTRAN Test Programs.* Their solution was a set of test programs that methodically exercised all the features of the language to be checked. If the routines were accepted by the compiler and generated the expected output, then the compiler was basically reliable (although no finite set of tests could ever prove that a compiler is completely correct). If not, then the errors would be noted and reported back to the vendor for correction. At first, the only official use of the test sets was to certify conforming implementations, which were then eligible for purchase by Federal agencies. Of course, this provided a strong incentive for vendors to produce standard-conforming software. In time, the idea of test sets to support software standards became widely accepted. This basic strategy is used to this day to validate software development tools, such as language compilers for C and Java and so-called Application Programmer Interfaces (APIs) such as SQL (for database operations) and OpenGL (for 3-D graphics). Since the same programs are used for all vendors, the testing process is transparently fair and objective. Of course, a human operator has to run the tests and examine the output, but this is not much different than a chemist obtaining the results of an experiment from a balance or other measuring instrument.

Furthermore, the test programs were ordered, checking the most basic language features first and then progressing to the more sophisticated aspects. This gave the operator good evidence about which features were the cause of any failures detected. NBS/NIST has since participated in the design, development, and operation of several comprehensive software test sets [3]. It should be stressed that, while the operation of the tests is mostly automated, the development of the tests requires a high degree of skill and understanding of the standard being applied. Test sets often comprise tens, or even hundreds of thousands, of lines of code. Conformance testing remains an important part of the NIST mission [10].

Betty Holberton worked for Remington-Rand in the early fifties, and then began her long career in government, first at the Applied Mathematics Laboratory of the David Taylor Model Basin (1953-1966), then at the National Bureau of Standards until her retirement in 1983. Her government work was spent largely in the development and testing of standards for computer languages. In 1997, her many achievements were recognized by the Association for Women in Computing, which presented her with its highest honor, the Augusta Ada Lovelace Award [9]. Also in 1997, Women in Technology International (WITI) inducted all six of the original ENIAC programmers into their Hall of Fame [5].

*Prepared by John Cugini.*

## Bibliography


[1] Frances E. Holberton and Elizabeth G. Parker, *NBS FORTRAN Test Programs,* NBS Special Publication 399 (3 volumes), National Bureau of Standards, Washington, DC, October 1974.

[2] Rachel K. Adelson, *Programmed to Succeed: Betty Holberton* (http://www.awc-hq.org/livewire/199705.html), Association for Women in Computing.

[3] J. Cugini, *Interactive Conformance Testing for PHIGS, Eurographics '91,* September 1991 (ftp://ftp.nist.gov/pub/vvrg/pvt/pvt-design.ps). This paper discusses some recent issues in the application of conformance testing to graphics.

[4] I. D. Hill and B. L. Meek, *Programming Language Standardisation,* Ellis Horwood Ltd., Chichester, England (1980).

[5] Kathryn A. Kleiman, Speech for WITI 1997 Hall of Fame Ceremony (http://www.witi.com/center/witimuseum/halloffame/1997/eniac.shtml), Women in Technology International.

[6] Terrence W. Pratt, *Programming Languages: Design and Implementation,* 2nd ed., Prentice-Hall, Englewood Cliffs, NJ (1984).

[7] Thomas Petzinger, Jr., *The First Computers Were The Women Who Invented Programming* (http://members.aol.com/douglaseby/Page83.html). *Wall Street Journal* article on ENIAC programmers, November, 1996.

[8] American National Standard FORTRAN, ANSI X3.9-1966. This is the original FORTRAN standard, long since superseded by more recent versions.

[9] Announcement of the 1997 Augusta Ada Lovelace Award (http://www.awc-hq.org/lovelace/1997.htm), Association for Women in Computing.

[10] Homepage of the NIST Software Diagnostics and Conformance Testing Division (SDCT) (http://www.itl.nist.gov/div897/), National Institute of Standards and Technology.