

Programmers typically use high-level programming languages that are more easily intelligible to humans than machine code, which is directly executed by the central processing unit. Proficient programming usually requires expertise in several different subjects, including knowledge of the application domain, details of programming languages and generic code libraries, specialized algorithms, and formal logic. Text editors were also developed that allowed changes and corrections to be made much more easily than with punched cards. Many factors, having little or nothing to do with the ability of the computer to efficiently compile and execute the code, contribute to readability. Debugging is often done with IDEs. Standalone debuggers like GDB are also used, and these often provide less of a visual environment, usually using a command line. Some languages are more prone to some kinds of faults because their specification does not require compilers to perform as much checking as other languages. There are many approaches to the Software development process. In 1801, the Jacquard loom could produce entirely different weaves by changing the "program" – a series of pasteboard cards with holes punched in them. In the 1880s, Herman Hollerith invented the concept of storing data in machine-readable form. Scripting and breakpointing is also part of this process. Debugging is a very important task in the software development process since having defects in a program can have significant consequences for its users. Assembly languages were soon developed that let the programmer specify instruction in a text format (e.g., ADD X, TOTAL), with abbreviations for each operation code and meaningful names for specifying addresses. Some of these factors include: The presentation aspects of this (such as indents, line breaks, color highlighting, and so on) are often handled by the source code editor, but the content aspects reflect the programmer's talent and skills. They are the building blocks for all software, from the simplest applications to the most sophisticated ones. Text editors were also developed that allowed changes and corrections to be made much more easily than with punched cards. A similar technique used for database design is Entity-Relationship Modeling (ER Modeling). Allen Downey, in his book *How To Think Like A Computer Scientist*, writes: Many computer languages provide a mechanism to call functions provided by shared libraries. The first computer program is generally dated to 1843, when mathematician Ada Lovelace published an algorithm to calculate a sequence of Bernoulli numbers, intended to be carried out by Charles Babbage's Analytical Engine. Compilers harnessed the power of computers to make programming easier by allowing programmers to specify calculations by entering a formula using infix notation. Many factors, having little or nothing to do with the ability of the computer to efficiently compile and execute the code, contribute to readability. Their jobs usually involve: Although programming has been presented in the media as a somewhat mathematical subject, some research shows that good programmers have strong skills in natural human languages, and that learning to code is similar to learning a foreign language. Some languages are more prone to some kinds of faults because their specification does not require compilers to perform as much checking as other languages. High-level languages made the process of developing a program simpler and more understandable, and less bound to the underlying hardware. It is very difficult to determine what are the most popular modern programming languages.