

Compilers harnessed the power of computers to make programming easier by allowing programmers to specify calculations by entering a formula using infix notation. High-level languages made the process of developing a program simpler and more understandable, and less bound to the underlying hardware. The following properties are among the most important: In computer programming, readability refers to the ease with which a human reader can comprehend the purpose, control flow, and operation of source code. It affects the aspects of quality above, including portability, usability and most importantly maintainability. Computer programmers are those who write computer software. Techniques like Code refactoring can enhance readability. However, Charles Babbage had already written his first program for the Analytical Engine in 1837. He gave the first description of cryptanalysis by frequency analysis, the earliest code-breaking algorithm. Programming languages are essential for software development. Later a control panel (plug board) added to his 1906 Type I Tabulator allowed it to be programmed for different jobs, and by the late 1940s, unit record equipment such as the IBM 602 and IBM 604, were programmed by control panels in a similar way, as were the first electronic computers. After the bug is reproduced, the input of the program may need to be simplified to make it easier to debug. FORTRAN, the first widely used high-level language to have a functional implementation, came out in 1957, and many other languages were soon developed—in particular, COBOL aimed at commercial data processing, and Lisp for computer research. Also, specific user environment and usage history can make it difficult to reproduce the problem. For example, when a bug in a compiler can make it crash when parsing some large source file, a simplification of the test case that results in only few lines from the original source file can be sufficient to reproduce the same crash. It is usually easier to code in "high-level" languages than in "low-level" ones. Popular modeling techniques include Object-Oriented Analysis and Design (OOAD) and Model-Driven Architecture (MDA). Whatever the approach to development may be, the final program must satisfy some fundamental properties. Sometimes software development is known as software engineering, especially when it employs formal methods or follows an engineering design process. 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. Languages form an approximate spectrum from "low-level" to "high-level"; "low-level" languages are typically more machine-oriented and faster to execute, whereas "high-level" languages are more abstract and easier to use but execute less quickly. The Unified Modeling Language (UML) is a notation used for both the OOAD and MDA. Computer programming or coding is the composition of sequences of instructions, called programs, that computers can follow to perform tasks. The academic field and the engineering practice of computer programming are both largely concerned with discovering and implementing the most efficient algorithms for a given class of problems. 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. Trade-offs from this ideal involve finding enough programmers who know the language to build a team, the availability of compilers for that language, and the efficiency with which programs written in a given language execute.