

It is usually easier to code in "high-level" languages than in "low-level" ones. While these are sometimes considered programming, often the term software development is used for this larger overall process – with the terms programming, implementation, and coding reserved for the writing and editing of code per se. Trial-and-error/divide-and-conquer is needed: the programmer will try to remove some parts of the original test case and check if the problem still exists. Sometimes software development is known as software engineering, especially when it employs formal methods or follows an engineering design process. Following a consistent programming style often helps readability. It involves designing and implementing algorithms, step-by-step specifications of procedures, by writing code in one or more programming languages. Use of a static code analysis tool can help detect some possible problems. They are the building blocks for all software, from the simplest applications to the most sophisticated ones. 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. 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. 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. 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. High-level languages made the process of developing a program simpler and more understandable, and less bound to the underlying hardware. Compilers harnessed the power of computers to make programming easier by allowing programmers to specify calculations by entering a formula using infix notation. It is very difficult to determine what are the most popular modern programming languages. Unreadable code often leads to bugs, inefficiencies, and duplicated code. In 1206, the Arab engineer Al-Jazari invented a programmable drum machine where a musical mechanical automaton could be made to play different rhythms and drum patterns, via pegs and cams. Integrated development environments (IDEs) aim to integrate all such help. He gave the first description of cryptanalysis by frequency analysis, the earliest code-breaking algorithm. New languages are generally designed around the syntax of a prior language with new functionality added, (for example C++ adds object-orientation to C, and Java adds memory management and bytecode to C++, but as a result, loses efficiency and the ability for low-level manipulation). 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. Code-breaking algorithms have also existed for centuries. In the 1880s, Herman Hollerith invented the concept of storing data in machine-readable form. Many applications use a mix of several languages in their construction and use. Expert programmers are familiar with a variety of well-established algorithms and their respective complexities and use this knowledge to choose algorithms that are best suited to the circumstances.