

It affects the aspects of quality above, including portability, usability and most importantly maintainability. Also, specific user environment and usage history can make it difficult to reproduce the problem. Many factors, having little or nothing to do with the ability of the computer to efficiently compile and execute the code, contribute to readability. Use of a static code analysis tool can help detect some possible problems. 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. Programming languages are essential for software development. In 1801, the Jacquard loom could produce entirely different weaves by changing the "program" – a series of pasteboard cards with holes punched in them. Use of a static code analysis tool can help detect some possible problems. 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. 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. There exist a lot of different approaches for each of those tasks. Unreadable code often leads to bugs, inefficiencies, and duplicated code. Many applications use a mix of several languages in their construction and use. In the 1880s, Herman Hollerith invented the concept of storing data in machine-readable form. 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. Computer programmers are those who write computer software. Programmable devices have existed for centuries. Compilers harnessed the power of computers to make programming easier by allowing programmers to specify calculations by entering a formula using infix notation. The first step in most formal software development processes is requirements analysis, followed by testing to determine value modeling, implementation, and failure elimination (debugging). The choice of language used is subject to many considerations, such as company policy, suitability to task, availability of third-party packages, or individual preference. 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. 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. This can be a non-trivial task, for example as with parallel processes or some unusual software bugs. The first step in most formal software development processes is requirements analysis, followed by testing to determine value modeling, implementation, and failure elimination (debugging).