

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. 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. It affects the aspects of quality above, including portability, usability and most importantly maintainability. 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. Various visual programming languages have also been developed with the intent to resolve readability concerns by adopting non-traditional approaches to code structure and display. Techniques like Code refactoring can enhance readability. After the bug is reproduced, the input of the program may need to be simplified to make it easier to debug. However, with the concept of the stored-program computer introduced in 1949, both programs and data were stored and manipulated in the same way in computer memory. One approach popular for requirements analysis is Use Case analysis. Readability is important because programmers spend the majority of their time reading, trying to understand, reusing and modifying existing source code, rather than writing new source code. However, because an assembly language is little more than a different notation for a machine language, two machines with different instruction sets also have different assembly languages. Ideally, the programming language best suited for the task at hand will be selected. Also, specific user environment and usage history can make it difficult to reproduce the problem. 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. 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. Programmable devices have existed for centuries. Whatever the approach to development may be, the final program must satisfy some fundamental properties. 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. He gave the first description of cryptanalysis by frequency analysis, the earliest code-breaking algorithm. It is very difficult to determine what are the most popular modern programming languages. He gave the first description of cryptanalysis by frequency analysis, the earliest code-breaking algorithm. He gave the first description of cryptanalysis by frequency analysis, the earliest code-breaking algorithm. 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.