

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. Debugging is a very important task in the software development process since having defects in a program can have significant consequences for its users. 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. Text editors were also developed that allowed changes and corrections to be made much more easily than with punched cards. Also, specific user environment and usage history can make it difficult to reproduce the problem. The first step in most formal software development processes is requirements analysis, followed by testing to determine value modeling, implementation, and failure elimination (debugging). 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. However, readability is more than just programming style. 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. Unreadable code often leads to bugs, inefficiencies, and duplicated code. Many factors, having little or nothing to do with the ability of the computer to efficiently compile and execute the code, contribute to readability. Provided the functions in a library follow the appropriate run-time conventions (e.g., method of passing arguments), then these functions may be written in any other language. Computer programmers are those who write computer software. A similar technique used for database design is Entity-Relationship Modeling (ER Modeling). By the late 1960s, data storage devices and computer terminals became inexpensive enough that programs could be created by typing directly into the computers. One approach popular for requirements analysis is Use Case analysis. High-level languages made the process of developing a program simpler and more understandable, and less bound to the underlying hardware. Sometimes software development is known as software engineering, especially when it employs formal methods or follows an engineering design process. For this purpose, algorithms are classified into orders using so-called Big O notation, which expresses resource use, such as execution time or memory consumption, in terms of the size of an input. Some languages are very popular for particular kinds of applications, while some languages are regularly used to write many different kinds of applications. A similar technique used for database design is Entity-Relationship Modeling (ER Modeling). By the late 1960s, data storage devices and computer terminals became inexpensive enough that programs could be created by typing directly into the computers. 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.