

Unreadable code often leads to bugs, inefficiencies, and duplicated code. Also, specific user environment and usage history can make it difficult to reproduce the problem. 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. Sometimes software development is known as software engineering, especially when it employs formal methods or follows an engineering design process. There exist a lot of different approaches for each of those tasks. 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. When debugging the problem in a GUI, the programmer can try to skip some user interaction from the original problem description and check if remaining actions are sufficient for bugs to appear. Some languages are more prone to some kinds of faults because their specification does not require compilers to perform as much checking as other languages. It involves designing and implementing algorithms, step-by-step specifications of procedures, by writing code in one or more programming languages. 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. 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. 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). Scripting and breakpointing is also part of this process. Assembly languages were soon developed that let the programmer specify instruction in a text format (e.g., ADD X, TOTAL), with abbreviations for each operation code and meaningful names for specifying addresses. Following a consistent programming style often helps readability. Different programming languages support different styles of programming (called programming paradigms). By the late 1960s, data storage devices and computer terminals became inexpensive enough that programs could be created by typing directly into the computers. When debugging the problem in a GUI, the programmer can try to skip some user interaction from the original problem description and check if remaining actions are sufficient for bugs to appear. Integrated development environments (IDEs) aim to integrate all such help. 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. Text editors were also developed that allowed changes and corrections to be made much more easily than with punched cards. Some languages are very popular for particular kinds of applications, while some languages are regularly used to write many different kinds of applications. 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. 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.