

Some of these factors include: The presentation aspects of this (such as indents, line breaks, color highlighting, and so on) are often handled by the source code editor, but the content aspects reflect the programmer's talent and skills. Different programming languages support different styles of programming (called programming paradigms). Techniques like Code refactoring can enhance readability. 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. Programs were mostly entered using punched cards or paper tape. Techniques like Code refactoring can enhance readability. 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). Programs were mostly entered using punched cards or paper tape. 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. Many factors, having little or nothing to do with the ability of the computer to efficiently compile and execute the code, contribute to readability. 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. It is usually easier to code in "high-level" languages than in "low-level" ones. 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. A similar technique used for database design is Entity-Relationship Modeling (ER Modeling). 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. A study found that a few simple readability transformations made code shorter and drastically reduced the time to understand it. 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. A similar technique used for database design is Entity-Relationship Modeling (ER Modeling). 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. High-level languages made the process of developing a program simpler and more understandable, and less bound to the underlying hardware. However, Charles Babbage had already written his first program for the Analytical Engine in 1837. Compilers harnessed the power of computers to make programming easier by allowing programmers to specify calculations by entering a formula using infix notation. Integrated development environments (IDEs) aim to integrate all such help. 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. Debugging is a very important task in the software development process since having defects in a program can have significant consequences for its users.