These compiled languages allow the programmer to write programs in terms that are syntactically richer, and more capable of abstracting the code, making it easy to target varying machine instruction sets via compilation declarations and heuristics. However, Charles Babbage had already written his first program for the Analytical Engine in 1837. 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. Compilers harnessed the power of computers to make programming easier by allowing programmers to specify calculations by entering a formula using infix notation. 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. Implementation techniques include imperative languages (object-oriented or procedural), functional languages, and logic languages. Use of a static code analysis tool can help detect some possible problems. This can be a non-trivial task, for example as with parallel processes or some unusual software bugs. Some text editors such as Emacs allow GDB to be invoked through them, to provide a visual environment. Whatever the approach to development may be, the final program must satisfy some fundamental properties. It affects the aspects of quality above, including portability, usability and most importantly maintainability. 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. This can be a non-trivial task, for example as with parallel processes or some unusual software bugs. Also, specific user environment and usage history can make it difficult to reproduce the problem. Also, specific user environment and usage history can make it difficult to reproduce the problem. There are many approaches to the Software development process. Implementation techniques include imperative languages (object-oriented or procedural), functional languages, and logic languages. 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. There are many approaches to the Software development process. Some text editors such as Emacs allow GDB to be invoked through them, to provide a visual environment. 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. Use of a static code analysis tool can help detect some possible problems. 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. 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. 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).