Use of a static code analysis tool can help detect some possible problems. 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. 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. Debugging is a very important task in the software development process since having defects in a program can have significant consequences for its users. For example, COBOL is still strong in corporate data centers often on large mainframe computers, Fortran in engineering applications, scripting languages in Web development, and C in embedded software. However, readability is more than just programming style. 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. 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. 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. Ideally, the programming language best suited for the task at hand will be selected. A similar technique used for database design is Entity-Relationship Modeling (ER Modeling). One approach popular for requirements analysis is Use Case analysis. 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. 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. Code-breaking algorithms have also existed for centuries. Auxiliary tasks accompanying and related to programming include analyzing requirements, testing, debugging (investigating and fixing problems), implementation of build systems, and management of derived artifacts, such as programs' machine code. Their jobs usually involve: Although programming has been presented in the media as a somewhat mathematical subject, some research shows that good programmers have strong skills in natural human languages, and that learning to code is similar to learning a foreign language. Unreadable code often leads to bugs, inefficiencies, and duplicated code. 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. For example, COBOL is still strong in corporate data centers often on large mainframe computers, Fortran in engineering applications, scripting languages in Web development, and C in embedded software. In the 1880s, Herman Hollerith invented the concept of storing data in machine-readable form. Implementation techniques include imperative languages (object-oriented or procedural), functional languages, and logic languages. Programs were mostly entered using punched cards or paper tape. Techniques like Code refactoring can enhance readability.