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. It involves designing and implementing algorithms, step-by-step specifications of procedures, by writing code in one or more programming languages. 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. 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. 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 is usually easier to code in "high-level" languages than in "low-level" ones. 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 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. 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. 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. A similar technique used for database design is Entity-Relationship Modeling (ER Modeling). He gave the first description of cryptanalysis by frequency analysis, the earliest code-breaking algorithm. Scripting and breakpointing is also part of this process. Debugging is a very important task in the software development process since having defects in a program can have significant consequences for its users. Also, specific user environment and usage history can make it difficult to reproduce the problem. Popular modeling techniques include Object-Oriented Analysis and Design (OOAD) and Model-Driven Architecture (MDA). Integrated development environments (IDEs) aim to integrate all such help. He gave the first description of cryptanalysis by frequency analysis, the earliest code-breaking algorithm. This can be a non-trivial task, for example as with parallel processes or some unusual software bugs. Some languages are very popular for particular kinds of applications, while some languages are regularly used to write many different kinds of applications. Use of a static code analysis tool can help detect some possible problems. 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. Ideally, the programming language best suited for the task at hand will be selected. After the bug is reproduced, the input of the program may need to be simplified to make it easier to debug. Scripting and breakpointing is also part of this process.