There exist a lot of different approaches for each of those tasks. One approach popular for requirements analysis is Use Case analysis. Scripting and breakpointing is also part of this process. The Unified Modeling Language (UML) is a notation used for both the OOAD and MDA. 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. 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. Some text editors such as Emacs allow GDB to be invoked through them, to provide a visual environment. In 1801, the Jacquard loom could produce entirely different weaves by changing the "program" – a series of pasteboard cards with holes punched in them. They are the building blocks for all software, from the simplest applications to the most sophisticated ones. Compilers harnessed the power of computers to make programming easier by allowing programmers to specify calculations by entering a formula using infix notation. 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. Ideally, the programming language best suited for the task at hand will be selected. Trade-offs from this ideal involve finding enough programmers who know the language to build a team, the availability of compilers for that language, and the efficiency with which programs written in a given language execute. Expert programmers are familiar with a variety of well-established algorithms and their respective complexities and use this knowledge to choose algorithms that are best suited to the circumstances. 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. A similar technique used for database design is Entity-Relationship Modeling (ER Modeling). 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. Implementation techniques include imperative languages (object-oriented or procedural), functional languages, and logic languages. High-level languages made the process of developing a program simpler and more understandable, and less bound to the underlying hardware. Computer programming or coding is the composition of sequences of instructions, called programs, that computers can follow to perform tasks. Programmable devices have existed for centuries. A study found that a few simple readability transformations made code shorter and drastically reduced the time to understand it. Many programmers use forms of Agile software development where the various stages of formal software development are more integrated together into short cycles that take a few weeks rather than years.