

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. 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. Many applications use a mix of several languages in their construction and use. By the late 1960s, data storage devices and computer terminals became inexpensive enough that programs could be created by typing directly into the computers. Sometimes software development is known as software engineering, especially when it employs formal methods or follows an engineering design process. Later a control panel (plug board) added to his 1906 Type I Tabulator allowed it to be programmed for different jobs, and by the late 1940s, unit record equipment such as the IBM 602 and IBM 604, were programmed by control panels in a similar way, as were the first electronic computers. 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. Text editors were also developed that allowed changes and corrections to be made much more easily than with punched cards. Languages form an approximate spectrum from "low-level" to "high-level"; "low-level" languages are typically more machine-oriented and faster to execute, whereas "high-level" languages are more abstract and easier to use but execute less quickly. 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. Different programming languages support different styles of programming (called programming paradigms). There exist a lot of different approaches for each of those tasks. A similar technique used for database design is Entity-Relationship Modeling (ER Modeling). Code-breaking algorithms have also existed for centuries. Integrated development environments (IDEs) aim to integrate all such help. Normally the first step in debugging is to attempt to reproduce the problem. 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. Computer programming or coding is the composition of sequences of instructions, called programs, that computers can follow to perform tasks. The first step in most formal software development processes is requirements analysis, followed by testing to determine value modeling, implementation, and failure elimination (debugging). 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. 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. 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. 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. In the 9th century, the Arab mathematician Al-Kindi described a cryptographic algorithm for deciphering encrypted code, in A Manuscript on Deciphering Cryptographic Messages.