

Scripting and breakpointing is also part of this process. In the 9th century, the Arab mathematician Al-Kindi described a cryptographic algorithm for deciphering encrypted code, in A Manuscript on Deciphering Cryptographic Messages. He gave the first description of cryptanalysis by frequency analysis, the earliest code-breaking algorithm. There are many approaches to the Software development process. 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. Many factors, having little or nothing to do with the ability of the computer to efficiently compile and execute the code, contribute to readability. Whatever the approach to development may be, the final program must satisfy some fundamental properties. By the late 1960s, data storage devices and computer terminals became inexpensive enough that programs could be created by typing directly into the computers. A similar technique used for database design is Entity-Relationship Modeling (ER Modeling). In 1206, the Arab engineer Al-Jazari invented a programmable drum machine where a musical mechanical automaton could be made to play different rhythms and drum patterns, via pegs and cams. There exist a lot of different approaches for each of those tasks. Normally the first step in debugging is to attempt to reproduce the problem. 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. 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. When debugging the problem in a GUI, the programmer can try to skip some user interaction from the original problem description and check if remaining actions are sufficient for bugs to appear. It is usually easier to code in "high-level" languages than in "low-level" ones. Programming languages are essential for software development. 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. 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. Unreadable code often leads to bugs, inefficiencies, and duplicated code. Methods of measuring programming language popularity include: counting the number of job advertisements that mention the language, the number of books sold and courses teaching the language (this overestimates the importance of newer languages), and estimates of the number of existing lines of code written in the language (this underestimates the number of users of business languages such as COBOL). Many factors, having little or nothing to do with the ability of the computer to efficiently compile and execute the code, contribute to readability. Machine code was the language of early programs, written in the instruction set of the particular machine, often in binary notation. 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. Code-breaking algorithms have also existed for centuries.