

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. Code-breaking algorithms have also existed for centuries. Many factors, having little or nothing to do with the ability of the computer to efficiently compile and execute the code, contribute to readability. Computer programming or coding is the composition of sequences of instructions, called programs, that computers can follow to perform tasks. In the 9th century, the Arab mathematician Al-Kindi described a cryptographic algorithm for deciphering encrypted code, in A Manuscript on Deciphering Cryptographic Messages. In the 1880s, Herman Hollerith invented the concept of storing data in machine-readable form. In the 9th century, the Arab mathematician Al-Kindi described a cryptographic algorithm for deciphering encrypted code, in A Manuscript on Deciphering Cryptographic Messages. Also, specific user environment and usage history can make it difficult to reproduce the problem. The first step in most formal software development processes is requirements analysis, followed by testing to determine value modeling, implementation, and failure elimination (debugging). Following a consistent programming style often helps readability. Techniques like Code refactoring can enhance readability. 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. Readability is important because programmers spend the majority of their time reading, trying to understand, reusing and modifying existing source code, rather than writing new source code. Programming languages are essential for software development. 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. Some languages are very popular for particular kinds of applications, while some languages are regularly used to write many different kinds of applications. Implementation techniques include imperative languages (object-oriented or procedural), functional languages, and logic 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. 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). Following a consistent programming style often helps readability. After the bug is reproduced, the input of the program may need to be simplified to make it easier to debug. Some text editors such as Emacs allow GDB to be invoked through them, to provide a visual environment. 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. This can be a non-trivial task, for example as with parallel processes or some unusual software bugs.