

He gave the first description of cryptanalysis by frequency analysis, the earliest code-breaking algorithm. Scripting and breakpointing is also part of this process. 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. However, readability is more than just programming style. Many factors, having little or nothing to do with the ability of the computer to efficiently compile and execute the code, contribute to readability. 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. By the late 1960s, data storage devices and computer terminals became inexpensive enough that programs could be created by typing directly into the computers. In 1801, the Jacquard loom could produce entirely different weaves by changing the "program" – a series of pasteboard cards with holes punched in them. However, Charles Babbage had already written his first program for the Analytical Engine in 1837. However, with the concept of the stored-program computer introduced in 1949, both programs and data were stored and manipulated in the same way in computer memory. 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. 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. Code-breaking algorithms have also existed for centuries. 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. Following a consistent programming style often helps readability. Some of these factors include: The presentation aspects of this (such as indents, line breaks, color highlighting, and so on) are often handled by the source code editor, but the content aspects reflect the programmer's talent and skills. 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. Implementation techniques include imperative languages (object-oriented or procedural), functional languages, and logic languages. Proficient programming usually requires expertise in several different subjects, including knowledge of the application domain, details of programming languages and generic code libraries, specialized algorithms, and formal logic. The first step in most formal software development processes is requirements analysis, followed by testing to determine value modeling, implementation, and failure elimination (debugging). Machine code was the language of early programs, written in the instruction set of the particular machine, often in binary notation. It affects the aspects of quality above, including portability, usability and most importantly maintainability. FORTRAN, the first widely used high-level language to have a functional implementation, came out in 1957, and many other languages were soon developed—in particular, COBOL aimed at commercial data processing, and Lisp for computer research.