Code-breaking algorithms have also existed for centuries. However, Charles Babbage had already written his first program for the Analytical Engine in 1837. Techniques like Code refactoring can enhance readability. Different programming languages support different styles of programming (called programming paradigms). Some languages are very popular for particular kinds of applications, while some languages are regularly used to write many different kinds of applications. 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. 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 1880s, Herman Hollerith invented the concept of storing data in machine-readable form. Compilers harnessed the power of computers to make programming easier by allowing programmers to specify calculations by entering a formula using infix notation. It is very difficult to determine what are the most popular modern programming languages. High-level languages made the process of developing a program simpler and more understandable, and less bound to the underlying hardware. Many factors, having little or nothing to do with the ability of the computer to efficiently compile and execute the code, contribute to readability. However, readability is more than just programming style. 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. 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. The choice of language used is subject to many considerations, such as company policy, suitability to task, availability of third-party packages, or individual preference. Auxiliary tasks accompanying and related to programming include analyzing requirements, testing, debugging (investigating and fixing problems), implementation of build systems, and management of derived artifacts, such as programs' machine code. The first compiler related tool, the A-0 System, was developed in 1952 by Grace Hopper, who also coined the term 'compiler'. In 1801, the Jacquard loom could produce entirely different weaves by changing the "program" – a series of pasteboard cards with holes punched in them. In 1801, the Jacquard loom could produce entirely different weaves by changing the "program" – a series of pasteboard cards with holes punched in them. Following a consistent programming style often helps readability. Use of a static code analysis tool can help detect some possible problems. Compilers harnessed the power of computers to make programming easier by allowing programmers to specify calculations by entering a formula using infix notation. In the 9th century, the Arab mathematician Al-Kindi described a cryptographic algorithm for deciphering encrypted code, in A Manuscript on Deciphering Cryptographic Messages. 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.