In the 1880s, Herman Hollerith invented the concept of storing data in machine-readable form. 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. It is usually easier to code in "high-level" languages than in "low-level" ones. After the bug is reproduced, the input of the program may need to be simplified to make it easier to debug. He gave the first description of cryptanalysis by frequency analysis, the earliest code-breaking algorithm. 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. 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. 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. By the late 1960s, data storage devices and computer terminals became inexpensive enough that programs could be created by typing directly into the computers. 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. However, readability is more than just programming style. However, Charles Babbage had already written his first program for the Analytical Engine in 1837. Sometimes software development is known as software engineering, especially when it employs formal methods or follows an engineering design process. 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 the 9th century, the Arab mathematician Al-Kindi described a cryptographic algorithm for deciphering encrypted code, in A Manuscript on Deciphering Cryptographic Messages. Following a consistent programming style often helps readability. 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). Machine code was the language of early programs, written in the instruction set of the particular machine, often in binary notation. It is usually easier to code in "high-level" languages than in "low-level" ones. 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. 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). However, readability is more than just programming style. 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. Also, specific user environment and usage history can make it difficult to reproduce the problem.