Also, specific user environment and usage history can make it difficult to reproduce the problem. 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. The following properties are among the most important: In computer programming, readability refers to the ease with which a human reader can comprehend the purpose, control flow, and operation of source code. A study found that a few simple readability transformations made code shorter and drastically reduced the time to understand it. However, readability is more than just programming style. 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. Programmable devices have existed for centuries. 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. Use of a static code analysis tool can help detect some possible problems. It is very difficult to determine what are the most popular modern programming languages. Unreadable code often leads to bugs, inefficiencies, and duplicated code. Trial-and-error/divide-and-conquer is needed: the programmer will try to remove some parts of the original test case and check if the problem still exists. One approach popular for requirements analysis is Use Case analysis. Programs were mostly entered using punched cards or paper tape. 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. Machine code was the language of early programs, written in the instruction set of the particular machine, often in binary notation. 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. By the late 1960s, data storage devices and computer terminals became inexpensive enough that programs could be created by typing directly into the computers. Trial-and-error/divide-and-conquer is needed: the programmer will try to remove some parts of the original test case and check if the problem still exists. However, Charles Babbage had already written his first program for the Analytical Engine in 1837. In 1801, the Jacquard loom could produce entirely different weaves by changing the "program" – a series of pasteboard cards with holes punched in them. Ideally, the programming language best suited for the task at hand will be selected. Integrated development environments (IDEs) aim to integrate all such help. 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. In the 1880s, Herman Hollerith invented the concept of storing data in machine-readable form.