After the bug is reproduced, the input of the program may need to be simplified to make it easier to debug. There exist a lot of different approaches for each of those tasks. After the bug is reproduced, the input of the program may need to be simplified to make it easier to debug. Use of a static code analysis tool can help detect some possible problems. However, Charles Babbage had already written his first program for the Analytical Engine in 1837. 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. It is very difficult to determine what are the most popular modern programming languages. 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. Different programming languages support different styles of programming (called programming paradigms). 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. One approach popular for requirements analysis is Use Case analysis. The first compiler related tool, the A-0 System, was developed in 1952 by Grace Hopper, who also coined the term 'compiler'. Later a control panel (plug board) added to his 1906 Type I Tabulator allowed it to be programmed for different jobs, and by the late 1940s, unit record equipment such as the IBM 602 and IBM 604, were programmed by control panels in a similar way, as were the first electronic computers. 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. Programs were mostly entered using punched cards or paper tape. However, readability is more than just programming style. Integrated development environments (IDEs) aim to integrate all such help. 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. Expert programmers are familiar with a variety of well-established algorithms and their respective complexities and use this knowledge to choose algorithms that are best suited to the circumstances. 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. They are the building blocks for all software, from the simplest applications to the most sophisticated ones. While these are sometimes considered programming, often the term software development is used for this larger overall process – with the terms programming, implementation, and coding reserved for the writing and editing of code per se. Computer programming or coding is the composition of sequences of instructions, called programs, that computers can follow to perform tasks. Later a control panel (plug board) added to his 1906 Type I Tabulator allowed it to be programmed for different jobs, and by the late 1940s, unit record equipment such as the IBM 602 and IBM 604, were programmed by control panels in a similar way, as were the first electronic computers. Ideally, the programming language best suited for the task at hand will be selected.