Popular modeling techniques include Object-Oriented Analysis and Design (OOAD) and Model-Driven Architecture (MDA). This can be a non-trivial task, for example as with parallel processes or some unusual software bugs. Some text editors such as Emacs allow GDB to be invoked through them, to provide a visual environment. 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. Sometimes software development is known as software engineering, especially when it employs formal methods or follows an engineering design process. Some languages are very popular for particular kinds of applications, while some languages are regularly used to write many different kinds of applications. 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. The Unified Modeling Language (UML) is a notation used for both the OOAD and MDA. Use of a static code analysis tool can help detect some possible problems. The first compiler related tool, the A-0 System, was developed in 1952 by Grace Hopper, who also coined the term 'compiler'. The academic field and the engineering practice of computer programming are both largely concerned with discovering and implementing the most efficient algorithms for a given class of problems. Also, specific user environment and usage history can make it difficult to reproduce the problem. 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, 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. It is usually easier to code in "high-level" languages than in "low-level" ones. Scripting and breakpointing is also part of this process. 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. 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. Programs were mostly entered using punched cards or paper tape. 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. They are the building blocks for all software, from the simplest applications to the most sophisticated ones. Their jobs usually involve: Although programming has been presented in the media as a somewhat mathematical subject, some research shows that good programmers have strong skills in natural human languages, and that learning to code is similar to learning a foreign language. 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. 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.