One approach popular for requirements analysis is Use Case analysis. 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. Text editors were also developed that allowed changes and corrections to be made much more easily than with punched cards. They are the building blocks for all software, from the simplest applications to the most sophisticated ones. 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. However, readability is more than just programming style. In the 9th century, the Arab mathematician Al-Kindi described a cryptographic algorithm for deciphering encrypted code, in A Manuscript on Deciphering Cryptographic Messages. Programmers typically use high-level programming languages that are more easily intelligible to humans than machine code, which is directly executed by the central processing unit. 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. 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. 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. However, readability is more than just programming style. Different programming languages support different styles of programming (called programming paradigms). Unreadable code often leads to bugs, inefficiencies, and duplicated code. Text editors were also developed that allowed changes and corrections to be made much more easily than with punched cards. It involves designing and implementing algorithms, step-by-step specifications of procedures, by writing code in one or more programming languages. 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. 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. Also, specific user environment and usage history can make it difficult to reproduce the problem. Programmable devices have existed for centuries. By the late 1960s, data storage devices and computer terminals became inexpensive enough that programs could be created by typing directly into the computers. Also, specific user environment and usage history can make it difficult to reproduce the problem. Programmable devices have existed for centuries. Also, specific user environment and usage history can make it difficult to reproduce the problem. Compilers harnessed the power of computers to make programming easier by allowing programmers to specify calculations by entering a formula using infix notation.