FORTRAN, the first widely used high-level language to have a functional implementation, came out in 1957, and many other languages were soon developed—in particular, COBOL aimed at commercial data processing, and Lisp for computer research. Programming languages are essential for software development. 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. 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. Allen Downey, in his book How To Think Like A Computer Scientist, writes: Many computer languages provide a mechanism to call functions provided by shared libraries. Compilers harnessed the power of computers to make programming easier by allowing programmers to specify calculations by entering a formula using infix notation. Whatever the approach to development may be, the final program must satisfy some fundamental properties. Whatever the approach to development may be, the final program must satisfy some fundamental properties. Computer programmers are those who write computer software. Text editors were also developed that allowed changes and corrections to be made much more easily than with punched cards. One approach popular for requirements analysis is Use Case analysis. As early as the 9th century, a programmable music sequencer was invented by the Persian Banu Musa brothers, who described an automated mechanical flute player in the Book of Ingenious Devices. Whatever the approach to development may be, the final program must satisfy some fundamental properties. It affects the aspects of quality above, including portability, usability and most importantly maintainability. Programming languages are essential for software development. Programmable devices have existed for centuries. 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. It is usually easier to code in "high-level" languages than in "low-level" ones. Normally the first step in debugging is to attempt to reproduce the problem. Sometimes software development is known as software engineering, especially when it employs formal methods or follows an engineering design process. 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. Code-breaking algorithms have also existed for centuries. Compilers harnessed the power of computers to make programming easier by allowing programmers to specify calculations by entering a formula using infix notation. After the bug is reproduced, the input of the program may need to be simplified to make it easier to debug. In the 1880s, Herman Hollerith invented the concept of storing data in machine-readable form.