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. This can be a non-trivial task, for example as with parallel processes or some unusual software bugs. It is very difficult to determine what are the most popular modern programming languages. 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. Also, specific user environment and usage history can make it difficult to reproduce the problem. Integrated development environments (IDEs) aim to integrate all such help. In 1206, the Arab engineer Al-Jazari invented a programmable drum machine where a musical mechanical automaton could be made to play different rhythms and drum patterns, via pegs and cams. Programmable devices have existed for centuries. However, Charles Babbage had already written his first program for the Analytical Engine in 1837. 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. Methods of measuring programming language popularity include: counting the number of job advertisements that mention the language, the number of books sold and courses teaching the language (this overestimates the importance of newer languages), and estimates of the number of existing lines of code written in the language (this underestimates the number of users of business languages such as COBOL). 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. Scripting and breakpointing is also part of this process. Different programming languages support different styles of programming (called programming paradigms). 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. A similar technique used for database design is Entity-Relationship Modeling (ER Modeling). Trade-offs from this ideal involve finding enough programmers who know the language to build a team, the availability of compilers for that language, and the efficiency with which programs written in a given language execute. Different programming languages support different styles of programming (called programming paradigms). Provided the functions in a library follow the appropriate run-time conventions (e.g., method of passing arguments), then these functions may be written in any other language. After the bug is reproduced, the input of the program may need to be simplified to make it easier to debug. Many applications use a mix of several languages in their construction and use. Programming languages are essential for software development. 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. It involves designing and implementing algorithms, step-by-step specifications of procedures, by writing code in one or more programming languages. 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.