Following a consistent programming style often helps readability. 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. This can be a non-trivial task, for example as with parallel processes or some unusual software bugs. 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. 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. 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. For example, COBOL is still strong in corporate data centers often on large mainframe computers, Fortran in engineering applications, scripting languages in Web development, and C in embedded software. Following a consistent programming style often helps readability. They are the building blocks for all software, from the simplest applications to the most sophisticated ones. 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. Programmable devices have existed for centuries. It is very difficult to determine what are the most popular modern programming languages. Whatever the approach to development may be, the final program must satisfy some fundamental properties. 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. It is very difficult to determine what are the most popular modern programming languages. After the bug is reproduced, the input of the program may need to be simplified to make it easier to debug. Also, specific user environment and usage history can make it difficult to reproduce the problem. There exist a lot of different approaches for each of those tasks. 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. 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). This can be a non-trivial task, for example as with parallel processes or some unusual software bugs. 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. For example, when a bug in a compiler can make

it crash when parsing some large source file, a simplification of the test case that results in only few lines from the original source file can be sufficient to reproduce the same crash. Scripting and breakpointing is also part of this process.