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..  
 The first computer program is generally dated to 1843, when mathematician Ada Lovelace published an algorithm to calculate a sequence of Bernoulli numbers, intended to be carried out by Charles Babbage's Analytical Engine.  
 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.  
Techniques like Code refactoring can enhance readability.  
One approach popular for requirements analysis is Use Case analysis.  
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.  
It affects the aspects of quality above, including portability, usability and most importantly maintainability.  
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.  
 Following a consistent programming style often helps readability.  
 Code-breaking algorithms have also existed for centuries.  
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.  
 Whatever the approach to development may be, the final program must satisfy some fundamental properties.  
 After the bug is reproduced, the input of the program may need to be simplified to make it easier to debug.  
 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.  
It is usually easier to code in "high-level" languages than in "low-level" ones.