Assembly languages were soon developed that let the programmer specify instruction in a text format (e..g., ADD X, TOTAL), with abbreviations for each operation code and meaningful names for specifying addresses.  
 Machine code was the language of early programs, written in the instruction set of the particular machine, often in binary notation.  
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.  
Languages form an approximate spectrum from "low-level" to "high-level"; "low-level" languages are typically more machine-oriented and faster to execute, whereas "high-level" languages are more abstract and easier to use but execute less quickly.  
 Debugging is a very important task in the software development process since having defects in a program can have significant consequences for its users.  
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.  
 Different programming languages support different styles of programming (called programming paradigms).  
 Code-breaking algorithms have also existed for centuries.  
However, readability is more than just programming style.  
This can be a non-trivial task, for example as with parallel processes or some unusual software bugs.  
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.  
It affects the aspects of quality above, including portability, usability and most importantly maintainability.  
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.  
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.