After the bug is reproduced, the input of the program may need to be simplified to make it easier to debug..  
Normally the first step in debugging is to attempt to reproduce the problem.  
  
One approach popular for requirements analysis is Use Case analysis.  
 Programmable devices have existed for centuries.  
 Different programming languages support different styles of programming (called programming paradigms).  
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.  
 Various visual programming languages have also been developed with the intent to resolve readability concerns by adopting non-traditional approaches to code structure and display.  
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.  
In the 9th century, the Arab mathematician Al-Kindi described a cryptographic algorithm for deciphering encrypted code, in A Manuscript on Deciphering Cryptographic Messages.  
 These compiled languages allow the programmer to write programs in terms that are syntactically richer, and more capable of abstracting the code, making it easy to target varying machine instruction sets via compilation declarations and heuristics.  
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.  
He gave the first description of cryptanalysis by frequency analysis, the earliest code-breaking algorithm.  
Many factors, having little or nothing to do with the ability of the computer to efficiently compile and execute the code, contribute to readability.  
Unreadable code often leads to bugs, inefficiencies, and duplicated code.