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.  
A study found that a few simple readability transformations made code shorter and drastically reduced the time to understand it.  
When debugging the problem in a GUI, the programmer can try to skip some user interaction from the original problem description and check if remaining actions are sufficient for bugs to appear.  
Use of a static code analysis tool can help detect some possible problems.  
This can be a non-trivial task, for example as with parallel processes or some unusual software bugs.  
Use of a static code analysis tool can help detect some possible problems.  
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.  
To produce machine code, the source code must either be compiled or transpiled.  
 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.  
However, Charles Babbage had already written his first program for the Analytical Engine in 1837.  
 Computer programmers are those who write computer software.  
 Code-breaking algorithms have also existed for centuries.  
 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.  
The purpose of programming is to find a sequence of instructions that will automate the performance of a task (which can be as complex as an operating system) on a computer, often for solving a given problem.  
 Following a consistent programming style often helps readability.