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.  
Integrated development environments (IDEs) aim to integrate all such help.  
 Different programming languages support different styles of programming (called programming paradigms).  
 Some languages are very popular for particular kinds of applications, while some languages are regularly used to write many different kinds of applications.  
However, readability is more than just programming style.  
While these are sometimes considered programming, often the term software development is used for this larger overall process – with the terms programming, implementation, and coding reserved for the writing and editing of code per se.  
Unreadable code often leads to bugs, inefficiencies, and duplicated code.  
A study found that a few simple readability transformations made code shorter and drastically reduced the time to understand it.  
 Code-breaking algorithms have also existed for centuries.  
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.  
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.  
Many programmers use forms of Agile software development where the various stages of formal software development are more integrated together into short cycles that take a few weeks rather than years.  
 New languages are generally designed around the syntax of a prior language with new functionality added, (for example C++ adds object-orientation to C, and Java adds memory management and bytecode to C++, but as a result, loses efficiency and the ability for low-level manipulation).  
By the late 1960s, data storage devices and computer terminals became inexpensive enough that programs could be created by typing directly into the computers.  
The choice of language used is subject to many considerations, such as company policy, suitability to task, availability of third-party packages, or individual preference.