However, while these might be considered part of the programming process, often the term software development is more likely used for this larger overall process – whereas the terms programming, implementation, and coding tend to be focused on the actual writing of code.  
This is interpreted into machine code.  
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.  
Techniques like Code refactoring can enhance readability.  
However, while these might be considered part of the programming process, often the term software development is more likely used for this larger overall process – whereas the terms programming, implementation, and coding tend to be focused on the actual writing of code.  
It is usually easier to code in "high-level" languages than in "low-level" ones.  
Scripting and breakpointing is also part of this process.  
Text editors were also developed that allowed changes and corrections to be made much more easily than with punched cards.  
Unreadable code often leads to bugs, inefficiencies, and duplicated code.  
By the late 1960s, data storage devices and computer terminals became inexpensive enough that programs could be created by typing directly into the computers.  
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.  
Integrated development environments (IDEs) aim to integrate all such help.  
Unreadable code often leads to bugs, inefficiencies, and duplicated code.  
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.  
Proficient programming thus usually requires expertise in several different subjects, including knowledge of the application domain, specialized algorithms, and formal logic.