A study found that a few simple readability transformations made code shorter and drastically reduced the time to understand it.  
The following properties are among the most important:  
  
 In computer programming, readability refers to the ease with which a human reader can comprehend the purpose, control flow, and operation of source code.  
This is interpreted into machine code.  
It is usually easier to code in "high-level" languages than in "low-level" ones.  
There are many approaches to the Software development process.  
Relatedly, software engineering combines engineering techniques and principles with software development.  
Unreadable code often leads to bugs, inefficiencies, and duplicated code.  
Unreadable code often leads to bugs, inefficiencies, and duplicated 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.  
Some text editors such as Emacs allow GDB to be invoked through them, to provide a visual environment.  
By the late 1960s, data storage devices and computer terminals became inexpensive enough that programs could be created by typing directly into the computers.  
Many applications use a mix of several languages in their construction and use.  
Proficient programming thus usually requires expertise in several different subjects, including knowledge of the application domain, specialized algorithms, and formal logic.  
Also, those involved with software development may at times engage in reverse engineering, which is the practice of seeking to understand an existing program so as to re-implement its function in some way.  
Trial-and-error/divide-and-conquer is needed: the programmer will try to remove some parts of the original test case and check if the problem still exists.