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.  
However, readability is more than just programming style.  
Text editors were also developed that allowed changes and corrections to be made much more easily than with punched cards.  
Relatedly, software engineering combines engineering techniques and principles with software development.  
One approach popular for requirements analysis is Use Case analysis.  
Also, specific user environment and usage history can make it difficult to reproduce the problem.  
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.  
Compiling takes the source code from a low-level programming language and converts it into machine code.  
The source code of a program is written in one or more languages that are intelligible to programmers, rather than machine code, which is directly executed by the central processing unit.  
Compiling takes the source code from a low-level programming language and converts it into machine code.  
  
The first compiler related tool, the A-0 System, was developed in 1952 by Grace Hopper, who also coined the term 'compiler'.  
 Some languages are very popular for particular kinds of applications, while some languages are regularly used to write many different kinds of applications.  
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.  
 In the 1880s, Herman Hollerith invented the concept of storing data in machine-readable form.  
Techniques like Code refactoring can enhance readability.