Compiling takes the source code from a low-level programming language and converts it 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.  
However, readability is more than just programming style.  
However, Charles Babbage had already written his first program for the Analytical Engine in 1837.  
Some text editors such as Emacs allow GDB to be invoked through them, to provide a visual environment.  
Integrated development environments (IDEs) aim to integrate all such help.  
Use of a static code analysis tool can help detect some possible problems.  
Use of a static code analysis tool can help detect some possible problems.  
Some languages are more prone to some kinds of faults because their specification does not require compilers to perform as much checking as other languages.  
The Unified Modeling Language (UML) is a notation used for both the OOAD and MDA.  
 Following a consistent programming style often helps readability.  
 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.  
Transpiling on the other hand, takes the source-code from a high-level programming language and converts it into bytecode.  
Programming languages are essential for software development.  
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.