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.  
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.  
Techniques like Code refactoring can enhance readability.  
He gave the first description of cryptanalysis by frequency analysis, the earliest code-breaking algorithm.  
Integrated development environments (IDEs) aim to integrate all such help.  
There are many approaches to the Software development process.  
It affects the aspects of quality above, including portability, usability and most importantly maintainability.  
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.  
Compilers harnessed the power of computers to make programming easier by allowing programmers to specify calculations by entering a formula using infix notation.  
Proficient programming thus usually requires expertise in several different subjects, including knowledge of the application domain, specialized algorithms, and formal logic.  
By the late 1960s, data storage devices and computer terminals became inexpensive enough that programs could be created by typing directly into the computers.  
However, Charles Babbage had already written his first program for the Analytical Engine in 1837.  
Also, specific user environment and usage history can make it difficult to reproduce the problem.  
Many applications use a mix of several languages in their construction and use.