One approach popular for requirements analysis is Use Case analysis.  
Expert programmers are familiar with a variety of well-established algorithms and their respective complexities and use this knowledge to choose algorithms that are best suited to the circumstances.  
Relatedly, software engineering combines engineering techniques and principles with software development.  
Use of a static code analysis tool can help detect some possible problems.  
Programming languages are essential for software development.  
Compilers harnessed the power of computers to make programming easier by allowing programmers to specify calculations by entering a formula using infix notation.  
It affects the aspects of quality above, including portability, usability and most importantly maintainability.  
There exist a lot of different approaches for each of those tasks.  
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.  
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.  
 In the 1880s, Herman Hollerith invented the concept of storing data in machine-readable form.  
However, because an assembly language is little more than a different notation for a machine language, two machines with different instruction sets also have different assembly languages.  
Some text editors such as Emacs allow GDB to be invoked through them, to provide a visual environment.  
Many applications use a mix of several languages in their construction and use.  
Transpiling on the other hand, takes the source-code from a high-level programming language and converts it into bytecode.