Ideally, the programming language best suited for the task at hand will be selected.  
Use of a static code analysis tool can help detect some possible problems.  
One approach popular for requirements analysis is Use Case analysis.  
Compiling takes the source code from a low-level programming language and converts it into machine code.  
One approach popular for requirements analysis is Use Case analysis.  
By the late 1960s, data storage devices and computer terminals became inexpensive enough that programs could be created by typing directly into the computers.  
One approach popular for requirements analysis is Use Case analysis.  
Some text editors such as Emacs allow GDB to be invoked through them, to provide a visual environment.  
 Tasks accompanying and related to programming include testing, debugging, source code maintenance, implementation of build systems, and management of derived artifacts, such as the machine code of computer programs.  
 After the bug is reproduced, the input of the program may need to be simplified to make it easier to debug.  
Also, specific user environment and usage history can make it difficult to reproduce the problem.  
Proficient programming thus usually requires expertise in several different subjects, including knowledge of the application domain, specialized algorithms, and formal logic.  
  
The first compiler related tool, the A-0 System, was developed in 1952 by Grace Hopper, who also coined the term 'compiler'.  
Many applications use a mix of several languages in their construction and use.  
 Popular modeling techniques include Object-Oriented Analysis and Design (OOAD) and Model-Driven Architecture (MDA).