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..  
 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.  
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.  
Later a control panel (plug board) added to his 1906 Type I Tabulator allowed it to be programmed for different jobs, and by the late 1940s, unit record equipment such as the IBM 602 and IBM 604, were programmed by control panels in a similar way, as were the first electronic computers.  
Normally the first step in debugging is to attempt to reproduce the problem.  
Text editors were also developed that allowed changes and corrections to be made much more easily than with punched cards.  
They are the building blocks for all software, from the simplest applications to the most sophisticated ones.  
Many factors, having little or nothing to do with the ability of the computer to efficiently compile and execute the code, contribute to readability.  
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.  
  
The first compiler related tool, the A-0 System, was developed in 1952 by Grace Hopper, who also coined the term 'compiler'.  
  
A study found that a few simple readability transformations made code shorter and drastically reduced the time to understand it.  
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.  
The Unified Modeling Language (UML) is a notation used for both the OOAD and MDA.