Methods of measuring programming language popularity include: counting the number of job advertisements that mention the language, the number of books sold and courses teaching the language (this overestimates the importance of newer languages), and estimates of the number of existing lines of code written in the language (this underestimates the number of users of business languages such as COBOL)..  
Ideally, the programming language best suited for the task at hand will be selected.  
A study found that a few simple readability transformations made code shorter and drastically reduced the time to understand it.  
However, with the concept of the stored-program computer introduced in 1949, both programs and data were stored and manipulated in the same way in computer memory.  
For example, COBOL is still strong in corporate data centers often on large mainframe computers, Fortran in engineering applications, scripting languages in Web development, and C in embedded software.  
Assembly languages were soon developed that let the programmer specify instruction in a text format (e.g., ADD X, TOTAL), with abbreviations for each operation code and meaningful names for specifying addresses.  
It is usually easier to code in "high-level" languages than in "low-level" ones.  
 After the bug is reproduced, the input of the program may need to be simplified to make it easier to debug.  
 Implementation techniques include imperative languages (object-oriented or procedural), functional languages, and logic languages.  
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.  
 Machine code was the language of early programs, written in the instruction set of the particular machine, often in binary notation.  
 It is very difficult to determine what are the most popular modern programming languages.  
 New languages are generally designed around the syntax of a prior language with new functionality added, (for example C++ adds object-orientation to C, and Java adds memory management and bytecode to C++, but as a result, loses efficiency and the ability for low-level manipulation).  
 The first step in most formal software development processes is requirements analysis, followed by testing to determine value modeling, implementation, and failure elimination (debugging).  
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.