Implementation techniques include imperative languages (object-oriented or procedural), functional languages, and logic languages..  
  
 In the 1880s, Herman Hollerith invented the concept of storing data in machine-readable form.  
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.  
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.  
Unreadable code often leads to bugs, inefficiencies, and duplicated code.  
 Programmable devices have existed for centuries.  
The Unified Modeling Language (UML) is a notation used for both the OOAD and MDA.  
Compilers harnessed the power of computers to make programming easier by allowing programmers to specify calculations by entering a formula using infix notation.  
Many programmers use forms of Agile software development where the various stages of formal software development are more integrated together into short cycles that take a few weeks rather than years.  
 These compiled languages allow the programmer to write programs in terms that are syntactically richer, and more capable of abstracting the code, making it easy to target varying machine instruction sets via compilation declarations and heuristics.  
 Code-breaking algorithms have also existed for centuries.  
It involves designing and implementing algorithms, step-by-step specifications of procedures, by writing code in one or more programming languages.  
 Computer programmers are those who write computer software.  
FORTRAN, the first widely used high-level language to have a functional implementation, came out in 1957, and many other languages were soon developed—in particular, COBOL aimed at commercial data processing, and Lisp for computer research.