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..  
Text editors were also developed that allowed changes and corrections to be made much more easily than with punched cards.  
 Implementation techniques include imperative languages (object-oriented or procedural), functional languages, and logic languages.  
 Various visual programming languages have also been developed with the intent to resolve readability concerns by adopting non-traditional approaches to code structure and display.  
 A similar technique used for database design is Entity-Relationship Modeling (ER Modeling).  
 In the 1880s, Herman Hollerith invented the concept of storing data in machine-readable form.  
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.  
 Code-breaking algorithms have also existed for centuries.  
 Following a consistent programming style often helps readability.  
Many applications use a mix of several languages in their construction and use.  
Use of a static code analysis tool can help detect some possible problems.  
 Different programming languages support different styles of programming (called programming paradigms).  
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.  
Programmers typically use high-level programming languages that are more easily intelligible to humans than machine code, which is directly executed by the central processing unit.  
The Unified Modeling Language (UML) is a notation used for both the OOAD and MDA.