Programs were mostly entered using punched cards or paper tape..  
Text editors were also developed that allowed changes and corrections to be made much more easily than with punched cards.  
 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.  
 Following a consistent programming style often helps readability.  
The Unified Modeling Language (UML) is a notation used for both the OOAD and MDA.  
He gave the first description of cryptanalysis by frequency analysis, the earliest code-breaking algorithm.  
 Readability is important because programmers spend the majority of their time reading, trying to understand, reusing and modifying existing source code, rather than writing new source code.  
However, because an assembly language is little more than a different notation for a machine language, two machines with different instruction sets also have different assembly languages.  
Scripting and breakpointing is also part of this process.  
Trade-offs from this ideal involve finding enough programmers who know the language to build a team, the availability of compilers for that language, and the efficiency with which programs written in a given language execute.  
Many factors, having little or nothing to do with the ability of the computer to efficiently compile and execute the code, contribute to readability.  
 High-level languages made the process of developing a program simpler and more understandable, and less bound to the underlying hardware.  
 In the 1880s, Herman Hollerith invented the concept of storing data in machine-readable form.  
It affects the aspects of quality above, including portability, usability and most importantly maintainability.  
 Auxiliary tasks accompanying and related to programming include analyzing requirements, testing, debugging (investigating and fixing problems), implementation of build systems, and management of derived artifacts, such as programs' machine code.