Implementation techniques include imperative languages (object-oriented or procedural), functional languages, and logic languages..  
They are the building blocks for all software, from the simplest applications to the most sophisticated ones.  
Use of a static code analysis tool can help detect some possible problems.  
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.  
Unreadable code often leads to bugs, inefficiencies, and duplicated code.  
By the late 1960s, data storage devices and computer terminals became inexpensive enough that programs could be created by typing directly into the computers.  
 Programmable devices have existed for centuries.  
The Unified Modeling Language (UML) is a notation used for both the OOAD and MDA.  
 Different programming languages support different styles of programming (called programming paradigms).  
 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.  
  
While these are sometimes considered programming, often the term software development is used for this larger overall process – with the terms programming, implementation, and coding reserved for the writing and editing of code per se.  
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.  
 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.