The first compiler related tool, the A-0 System, was developed in 1952 by Grace Hopper, who also coined the term 'compiler'. 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). Many factors, having little or nothing to do with the ability of the computer to efficiently compile and execute the code, contribute to readability. 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. It involves designing and implementing algorithms, step-by-step specifications of procedures, by writing code in one or more programming languages. Some text editors such as Emacs allow GDB to be invoked through them, to provide a visual environment. The first computer program is generally dated to 1843, when mathematician Ada Lovelace published an algorithm to calculate a sequence of Bernoulli numbers, intended to be carried out by Charles Babbage's Analytical Engine. For example, when a bug in a compiler can make it crash when parsing some large source file, a simplification of the test case that results in only few lines from the original source file can be sufficient to reproduce the same crash. 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. 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. Different programming languages support different styles of programming (called programming paradigms). Unreadable code often leads to bugs, inefficiencies, and duplicated code. Provided the functions in a library follow the appropriate run-time conventions (e.g., method of passing arguments), then these functions may be written in any other language. Text editors were also developed that allowed changes and corrections to be made much more easily than with punched cards. Unreadable code often leads to bugs, inefficiencies, and duplicated code. Allen Downey, in his book How To Think Like A Computer Scientist, writes: Many computer languages provide a mechanism to call functions provided by shared libraries. 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. 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. 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. Computer programming or coding is the composition of sequences of instructions, called programs, that computers can follow to perform tasks. 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. High-level languages made the process of developing a program simpler and more understandable, and less bound to the underlying hardware. The Unified Modeling Language (UML) is a notation used for both the OOAD and MDA. There exist a lot of different approaches for each of those tasks.