

Some of these factors include: The presentation aspects of this (such as indents, line breaks, color highlighting, and so on) are often handled by the source code editor, but the content aspects reflect the programmer's talent and skills. The Unified Modeling Language (UML) is a notation used for both the OOAD and MDA. 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. Implementation techniques include imperative languages (object-oriented or procedural), functional languages, and logic 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 choice of language used is subject to many considerations, such as company policy, suitability to task, availability of third-party packages, or individual preference. The first compiler related tool, the A-0 System, was developed in 1952 by Grace Hopper, who also coined the term 'compiler'. 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. The choice of language used is subject to many considerations, such as company policy, suitability to task, availability of third-party packages, or individual preference. Following a consistent programming style often helps readability. There exist a lot of different approaches for each of those tasks. Debugging is often done with IDEs. Standalone debuggers like GDB are also used, and these often provide less of a visual environment, usually using a command line. 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. By the late 1960s, data storage devices and computer terminals became inexpensive enough that programs could be created by typing directly into the computers. Many factors, having little or nothing to do with the ability of the computer to efficiently compile and execute the code, contribute to readability. The academic field and the engineering practice of computer programming are both largely concerned with discovering and implementing the most efficient algorithms for a given class of problems. Proficient programming usually requires expertise in several different subjects, including knowledge of the application domain, details of programming languages and generic code libraries, specialized algorithms, and formal logic. After the bug is reproduced, the input of the program may need to be simplified to make it easier to debug. The choice of language used is subject to many considerations, such as company policy, suitability to task, availability of third-party packages, or individual preference. For example, COBOL is still strong in corporate data centers often on large mainframe computers, Fortran in engineering applications, scripting languages in Web development, and C in embedded software. 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. 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.