

Machine code was the language of early programs, written in the instruction set of the particular machine, often in binary notation. One approach popular for requirements analysis is Use Case analysis. 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. Computer programming or coding is the composition of sequences of instructions, called programs, that computers can follow to perform tasks. Use of a static code analysis tool can help detect some possible problems. In 1801, the Jacquard loom could produce entirely different weaves by changing the "program" – a series of pasteboard cards with holes punched in them. 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. Their jobs usually involve: Although programming has been presented in the media as a somewhat mathematical subject, some research shows that good programmers have strong skills in natural human languages, and that learning to code is similar to learning a foreign language. Popular modeling techniques include Object-Oriented Analysis and Design (OOAD) and Model-Driven Architecture (MDA). 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. Integrated development environments (IDEs) aim to integrate all such help. 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. Text editors were also developed that allowed changes and corrections to be made much more easily than with punched cards. Compilers harnessed the power of computers to make programming easier by allowing programmers to specify calculations by entering a formula using infix notation. There exist a lot of different approaches for each of those tasks. Many factors, having little or nothing to do with the ability of the computer to efficiently compile and execute the code, contribute to readability. Whatever the approach to development may be, the final program must satisfy some fundamental properties. Whatever the approach to development may be, the final program must satisfy some fundamental properties. 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. It is very difficult to determine what are the most popular modern programming languages. The first compiler related tool, the A-0 System, was developed in 1952 by Grace Hopper, who also coined the term 'compiler'. They are the building blocks for all software, from the simplest applications to the most sophisticated ones. 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. 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.