

Compilers harnessed the power of computers to make programming easier by allowing programmers to specify calculations by entering a formula using infix notation. For this purpose, algorithms are classified into orders using so-called Big O notation, which expresses resource use, such as execution time or memory consumption, in terms of the size of an input. There are many approaches to the Software development 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. In 1801, the Jacquard loom could produce entirely different weaves by changing the "program" – a series of pasteboard cards with holes punched in them. 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. It is usually easier to code in "high-level" languages than in "low-level" ones. Unreadable code often leads to bugs, inefficiencies, and duplicated code. 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. Some languages are very popular for particular kinds of applications, while some languages are regularly used to write many different kinds of applications. Machine code was the language of early programs, written in the instruction set of the particular machine, often in binary notation. Languages form an approximate spectrum from "low-level" to "high-level"; "low-level" languages are typically more machine-oriented and faster to execute, whereas "high-level" languages are more abstract and easier to use but execute less quickly. One approach popular for requirements analysis is Use Case analysis. Different programming languages support different styles of programming (called programming paradigms). Many applications use a mix of several languages in their construction and use. Computer programming or coding is the composition of sequences of instructions, called programs, that computers can follow to perform tasks. Methods of measuring programming language popularity include: counting the number of job advertisements that mention the language, the number of books sold and courses teaching the language (this overestimates the importance of newer languages), and estimates of the number of existing lines of code written in the language (this underestimates the number of users of business languages such as COBOL). Debugging is a very important task in the software development process since having defects in a program can have significant consequences for its users. Different programming languages support different styles of programming (called programming paradigms). 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. Machine code was the language of early programs, written in the instruction set of the particular machine, often in binary notation. 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. 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. 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. 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.