

It is usually easier to code in "high-level" languages than in "low-level" ones. Ideally, the programming language best suited for the task at hand will be selected. 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. 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. In the 1880s, Herman Hollerith invented the concept of storing data in machine-readable form. Also, specific user environment and usage history can make it difficult to reproduce the problem. 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). 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. 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. Many factors, having little or nothing to do with the ability of the computer to efficiently compile and execute the code, contribute to readability. It is usually easier to code in "high-level" languages than in "low-level" ones. 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. It is very difficult to determine what are the most popular modern programming languages. There exist a lot of different approaches for each of those tasks. Expert programmers are familiar with a variety of well-established algorithms and their respective complexities and use this knowledge to choose algorithms that are best suited to the circumstances. 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. Popular modeling techniques include Object-Oriented Analysis and Design (OOAD) and Model-Driven Architecture (MDA). They are the building blocks for all software, from the simplest applications to the most sophisticated ones. Integrated development environments (IDEs) aim to integrate all such help. Later a control panel (plug board) added to his 1906 Type I Tabulator allowed it to be programmed for different jobs, and by the late 1940s, unit record equipment such as the IBM 602 and IBM 604, were programmed by control panels in a similar way, as were the first electronic computers. There are many approaches to the Software development process. By the late 1960s, data storage devices and computer terminals became inexpensive enough that programs could be created by typing directly into the computers. Different programming languages support different styles of programming (called programming paradigms). Different programming languages support different styles of programming (called programming paradigms).