

High-level languages made the process of developing a program simpler and more understandable, and less bound to the underlying hardware. In the 1880s, Herman Hollerith invented the concept of storing data in machine-readable form. 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). Text editors were also developed that allowed changes and corrections to be made much more easily than with punched cards. Scripting and breakpointing is also part of this process. The Unified Modeling Language (UML) is a notation used for both the OOAD and MDA. Whatever the approach to development may be, the final program must satisfy some fundamental properties. The following properties are among the most important: In computer programming, readability refers to the ease with which a human reader can comprehend the purpose, control flow, and operation of 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. Machine code was the language of early programs, written in the instruction set of the particular machine, often in binary notation. Techniques like Code refactoring can enhance readability. 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 first step in most formal software development processes is requirements analysis, followed by testing to determine value modeling, implementation, and failure elimination (debugging). In 1206, the Arab engineer Al-Jazari invented a programmable drum machine where a musical mechanical automaton could be made to play different rhythms and drum patterns, via pegs and cams. Computer programming or coding is the composition of sequences of instructions, called programs, that computers can follow to perform tasks. After the bug is reproduced, the input of the program may need to be simplified to make it easier to debug. One approach popular for requirements analysis is Use Case analysis. 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. 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. Following a consistent programming style often helps readability. 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. Many applications use a mix of several languages in their construction and use. 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). It affects the aspects of quality above, including portability, usability and most importantly maintainability.