

Unreadable code often leads to bugs, inefficiencies, and duplicated code. 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. Integrated development environments (IDEs) aim to integrate all such help. 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. After the bug is reproduced, the input of the program may need to be simplified to make it easier to debug. Auxiliary tasks accompanying and related to programming include analyzing requirements, testing, debugging (investigating and fixing problems), implementation of build systems, and management of derived artifacts, such as programs' machine code. Following a consistent programming style often helps readability. The first computer program is generally dated to 1843, when mathematician Ada Lovelace published an algorithm to calculate a sequence of Bernoulli numbers, intended to be carried out by Charles Babbage's Analytical Engine. 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. Whatever the approach to development may be, the final program must satisfy some fundamental properties. Many applications use a mix of several languages in their construction and use. 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. Whatever the approach to development may be, the final program must satisfy some fundamental properties. Unreadable code often leads to bugs, inefficiencies, and duplicated code. Machine code was the language of early programs, written in the instruction set of the particular machine, often in binary notation. In 1801, the Jacquard loom could produce entirely different weaves by changing the "program" – a series of pasteboard cards with holes punched in them. Code-breaking algorithms have also existed for centuries. However, Charles Babbage had already written his first program for the Analytical Engine in 1837. Code-breaking algorithms have also existed for centuries. 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. 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. 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. Some text editors such as Emacs allow GDB to be invoked through them, to provide a visual environment. Some languages are very popular for particular kinds of applications, while some languages are regularly used to write many different kinds of applications. 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.