

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. 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 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. Implementation techniques include imperative languages (object-oriented or procedural), functional languages, and logic languages. Unreadable code often leads to bugs, inefficiencies, and duplicated code. In the 1880s, Herman Hollerith invented the concept of storing data in machine-readable form. Some text editors such as Emacs allow GDB to be invoked through them, to provide a visual environment. For example, when a bug in a compiler can make it crash when parsing some large source file, a simplification of the test case that results in only few lines from the original source file can be sufficient to reproduce the same crash. 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. 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. This can be a non-trivial task, for example as with parallel processes or some unusual software bugs. Some languages are more prone to some kinds of faults because their specification does not require compilers to perform as much checking as other languages. Compilers harnessed the power of computers to make programming easier by allowing programmers to specify calculations by entering a formula using infix notation. Some languages are more prone to some kinds of faults because their specification does not require compilers to perform as much checking as other languages. In the 9th century, the Arab mathematician Al-Kindi described a cryptographic algorithm for deciphering encrypted code, in A Manuscript on Deciphering Cryptographic Messages. Text editors were also developed that allowed changes and corrections to be made much more easily than with punched cards. High-level languages made the process of developing a program simpler and more understandable, and less bound to the underlying hardware. Normally the first step in debugging is to attempt to reproduce the problem. It is very difficult to determine what are the most popular modern programming languages. Also, specific user environment and usage history can make it difficult to reproduce the problem. Popular modeling techniques include Object-Oriented Analysis and Design (OOAD) and Model-Driven Architecture (MDA). 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. 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.