

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. Programmable devices have existed for centuries. Programs were mostly entered using punched cards or paper tape. In the 1880s, Herman Hollerith invented the concept of storing data in machine-readable form. 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. 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. It involves designing and implementing algorithms, step-by-step specifications of procedures, by writing code in one or more programming languages. Some languages are very popular for particular kinds of applications, while some languages are regularly used to write many different kinds of applications. Techniques like Code refactoring can enhance readability. 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. 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. Following a consistent programming style often helps readability. As early as the 9th century, a programmable music sequencer was invented by the Persian Banu Musa brothers, who described an automated mechanical flute player in the Book of Ingenious Devices. In the 1880s, Herman Hollerith invented the concept of storing data in machine-readable form. Programming languages are essential for software development. 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). Different programming languages support different styles of programming (called programming paradigms). 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. 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. After the bug is reproduced, the input of the program may need to be simplified to make it easier to debug. 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. Use of a static code analysis tool can help detect some possible problems. 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. They are the building blocks for all software, from the simplest applications to the most sophisticated ones. The Unified Modeling Language (UML) is a notation used for both the OOAD and MDA.