Sometimes software development is known as software engineering, especially when it employs formal methods or follows an engineering design process. 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. 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. 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. Ideally, the programming language best suited for the task at hand will be selected. 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). Programming languages are essential for software development. Programmable devices have existed for centuries. A study found that a few simple readability transformations made code shorter and drastically reduced the time to understand it. Integrated development environments (IDEs) aim to integrate all such help. 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. Allen Downey, in his book How To Think Like A Computer Scientist, writes: Many computer languages provide a mechanism to call functions provided by shared libraries. Implementation techniques include imperative languages (object-oriented or procedural), functional languages, and logic languages. Techniques like Code refactoring can enhance readability. After the bug is reproduced, the input of the program may need to be simplified to make it easier to debug. It is usually easier to code in "high-level" languages than in "low-level" ones. Programs were mostly entered using punched cards or paper tape. 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. One approach popular for requirements analysis is Use Case analysis. 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 Unified Modeling Language (UML) is a notation used for both the OOAD and MDA. 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. One approach popular for requirements analysis is Use Case analysis. One approach popular for requirements analysis is Use Case analysis. This can be a non-trivial task, for example as with parallel processes or some unusual software bugs.