In the 1880s, Herman Hollerith invented the concept of storing data in machine-readable form. 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. It is usually easier to code in "high-level" languages than in "low-level" ones. This can be a non-trivial task, for example as with parallel processes or some unusual software bugs. Provided the functions in a library follow the appropriate run-time conventions (e.g., method of passing arguments), then these functions may be written in any other language. Programmers typically use high-level programming languages that are more easily intelligible to humans than machine code, which is directly executed by the central processing unit. It is usually easier to code in "high-level" languages than in "low-level" ones. Following a consistent programming style often helps readability. 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. 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. 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. 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. Implementation techniques include imperative languages (object-oriented or procedural), functional languages, and logic languages. 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. In the 1880s, Herman Hollerith invented the concept of storing data in machine-readable form. This can be a non-trivial task, for example as with parallel processes or some unusual software bugs. Ideally, the programming language best suited for the task at hand will be selected. Integrated development environments (IDEs) aim to integrate all such help. 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. 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. Many factors, having little or nothing to do with the ability of the computer to efficiently compile and execute the code, contribute to 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. Popular modeling techniques include Object-Oriented Analysis and Design (OOAD) and Model-Driven Architecture (MDA). It affects the aspects of quality above, including portability, usability and most importantly maintainability. Code-breaking algorithms have also existed for centuries.