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. However, with the concept of the stored-program computer introduced in 1949, both programs and data were stored and manipulated in the same way in computer memory. In the 1880s, Herman Hollerith invented the concept of storing data in machine-readable form. It is usually easier to code in "high-level" languages than in "low-level" ones. 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. 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. In 1801, the Jacquard loom could produce entirely different weaves by changing the "program" – a series of pasteboard cards with holes punched in them. 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. After the bug is reproduced, the input of the program may need to be simplified to make it easier to debug. Unreadable code often leads to bugs, inefficiencies, and duplicated code. Also, specific user environment and usage history can make it difficult to reproduce the problem. 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 very popular for particular kinds of applications, while some languages are regularly used to write many different kinds of applications. Different programming languages support different styles of programming (called programming paradigms). Programmable devices have existed for centuries. Computer programming or coding is the composition of sequences of instructions, called programs, that computers can follow to perform tasks. However, readability is more than just programming style. Programmable devices have existed for centuries. Ideally, the programming language best suited for the task at hand will be selected. Implementation techniques include imperative languages (object-oriented or procedural), functional languages, and logic languages. 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. 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. It is usually easier to code in "high-level" languages than in "low-level" ones. Ideally, the programming language best suited for the task at hand will be selected. Code-breaking algorithms have also existed for centuries.