Many factors, having little or nothing to do with the ability of the computer to efficiently compile and execute the code, contribute to readability. Trial-and-error/divide-and-conquer is needed: the programmer will try to remove some parts of the original test case and check if the problem still exists. 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. 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. A similar technique used for database design is Entity-Relationship Modeling (ER Modeling). It involves designing and implementing algorithms, step-by-step specifications of procedures, by writing code in one or more programming languages. Also, specific user environment and usage history can make it difficult to reproduce the problem. 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. Scripting and breakpointing is also part of this process. 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. 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. Some languages are very popular for particular kinds of applications, while some languages are regularly used to write many different kinds of applications. Unreadable code often leads to bugs, inefficiencies, and duplicated code. Whatever the approach to development may be, the final program must satisfy some fundamental properties. The first step in most formal software development processes is requirements analysis, followed by testing to determine value modeling, implementation, and failure elimination (debugging). 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. He gave the first description of cryptanalysis by frequency analysis, the earliest code-breaking algorithm. 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. 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. 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. There are many approaches to the Software development process. It is very difficult to determine what are the most popular modern programming languages. 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. FORTRAN, the first widely used high-level language to have a functional implementation, came out in 1957, and many other languages were soon developed—in particular, COBOL aimed at commercial data processing, and Lisp for computer research.