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). Unreadable code often leads to bugs, inefficiencies, and duplicated code. Implementation techniques include imperative languages (object-oriented or procedural), functional languages, and logic languages. After the bug is reproduced, the input of the program may need to be simplified to make it easier to debug. Text editors were also developed that allowed changes and corrections to be made much more easily than with punched cards. However, Charles Babbage had already written his first program for the Analytical Engine in 1837. Ideally, the programming language best suited for the task at hand will be selected. A similar technique used for database design is Entity-Relationship Modeling (ER Modeling). He gave the first description of cryptanalysis by frequency analysis, the earliest code-breaking algorithm. Different programming languages support different styles of programming (called programming paradigms). Proficient programming usually requires expertise in several different subjects, including knowledge of the application domain, details of programming languages and generic code libraries, specialized algorithms, and formal logic. The following properties are among the most important: In computer programming, readability refers to the ease with which a human reader can comprehend the purpose, control flow, and operation of source code. It involves designing and implementing algorithms, step-by-step specifications of procedures, by writing code in one or more programming languages. High-level languages made the process of developing a program simpler and more understandable, and less bound to the underlying hardware. 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. Many factors, having little or nothing to do with the ability of the computer to efficiently compile and execute the code, contribute to readability. There exist a lot of different approaches for each of those tasks. The Unified Modeling Language (UML) is a notation used for both the OOAD and MDA. Implementation techniques include imperative languages (object-oriented or procedural), functional languages, and logic languages. Computer programmers are those who write computer software. 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. Some languages are very popular for particular kinds of applications, while some languages are regularly used to write many different kinds of applications. 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. It is usually easier to code in "high-level" languages than in "low-level" ones. Some languages are very popular for particular kinds of applications, while some languages are regularly used to write many different kinds of applications.