Machine code was the language of early programs, written in the instruction set of the particular machine, often in binary notation. Text editors were also developed that allowed changes and corrections to be made much more easily than with punched cards. Also, specific user environment and usage history can make it difficult to reproduce the problem. In 1801, the Jacquard loom could produce entirely different weaves by changing the "program" – a series of pasteboard cards with holes punched in them. 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. 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. Ideally, the programming language best suited for the task at hand will be selected. In the 1880s, Herman Hollerith invented the concept of storing data in machine-readable form. High-level languages made the process of developing a program simpler and more understandable, and less bound to the underlying hardware. Computer programming or coding is the composition of sequences of instructions, called programs, that computers can follow to perform tasks. 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. 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. They are the building blocks for all software, from the simplest applications to the most sophisticated ones. Normally the first step in debugging is to attempt to reproduce the problem. 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. Assembly languages were soon developed that let the programmer specify instruction in a text format (e.g., ADD X, TOTAL), with abbreviations for each operation code and meaningful names for specifying addresses. Unreadable code often leads to bugs, inefficiencies, and duplicated code. The Unified Modeling Language (UML) is a notation used for both the OOAD and MDA. Normally the first step in debugging is to attempt to reproduce the problem. A study found that a few simple readability transformations made code shorter and drastically reduced the time to understand it. 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). 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. High-level languages made the process of developing a program simpler and more understandable, and less bound to the underlying hardware. Text editors were also developed that allowed changes and corrections to be made much more easily than with punched cards.