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. Programming languages are essential for software development. A similar technique used for database design is Entity-Relationship Modeling (ER Modeling). 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. High-level languages made the process of developing a program simpler and more understandable, and less bound to the underlying hardware. Machine code was the language of early programs, written in the instruction set of the particular machine, often in binary notation. Whatever the approach to development may be, the final program must satisfy some fundamental properties. It is usually easier to code in "high-level" languages than in "low-level" ones. Programmable devices have existed for centuries. Some languages are more prone to some kinds of faults because their specification does not require compilers to perform as much checking as other languages. Different programming languages support different styles of programming (called programming paradigms). The Unified Modeling Language (UML) is a notation used for both the OOAD and MDA. Some text editors such as Emacs allow GDB to be invoked through them, to provide a visual environment. 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. A study found that a few simple readability transformations made code shorter and drastically reduced the time to understand it. Implementation techniques include imperative languages (object-oriented or procedural), functional languages, and logic languages. 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. Some text editors such as Emacs allow GDB to be invoked through them, to provide a visual environment. One approach popular for requirements analysis is Use Case analysis. 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. Integrated development environments (IDEs) aim to integrate all such help. As early as the 9th century, a programmable music sequencer was invented by the Persian Banu Musa brothers, who described an automated mechanical flute player in the Book of Ingenious Devices. He gave the first description of cryptanalysis by frequency analysis, the earliest code-breaking algorithm. Many factors, having little or nothing to do with the ability of the computer to efficiently compile and execute the code, contribute to readability. Integrated development environments (IDEs) aim to integrate all such help.