There are many approaches to the Software development process. 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 exist a lot of different approaches for each of those tasks. Some of these factors include: The presentation aspects of this (such as indents, line breaks, color highlighting, and so on) are often handled by the source code editor, but the content aspects reflect the programmer's talent and skills. Machine code was the language of early programs, written in the instruction set of the particular machine, often in binary notation. 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. Also, specific user environment and usage history can make it difficult to reproduce the problem. Integrated development environments (IDEs) aim to integrate all such help. Popular modeling techniques include Object-Oriented Analysis and Design (OOAD) and Model-Driven Architecture (MDA). There are many approaches to the Software development process. 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). 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. Sometimes software development is known as software engineering, especially when it employs formal methods or follows an engineering design process. Normally the first step in debugging is to attempt to reproduce the problem. Code-breaking algorithms have also existed for centuries. However, Charles Babbage had already written his first program for the Analytical Engine in 1837. Machine code was the language of early programs, written in the instruction set of the particular machine, often in binary notation. Their jobs usually involve: Although programming has been presented in the media as a somewhat mathematical subject, some research shows that good programmers have strong skills in natural human languages, and that learning to code is similar to learning a foreign language. He gave the first description of cryptanalysis by frequency analysis, the earliest code-breaking algorithm. Various visual programming languages have also been developed with the intent to resolve readability concerns by adopting non-traditional approaches to code structure and display. 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. 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. 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. This can be a non-trivial task, for example as with parallel processes or some unusual software bugs.