One approach popular for requirements analysis is Use Case analysis. There are many approaches to the Software development process. 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. Some text editors such as Emacs allow GDB to be invoked through them, to provide a visual environment. By the late 1960s, data storage devices and computer terminals became inexpensive enough that programs could be created by typing directly into the computers. Later a control panel (plug board) added to his 1906 Type I Tabulator allowed it to be programmed for different jobs, and by the late 1940s, unit record equipment such as the IBM 602 and IBM 604, were programmed by control panels in a similar way, as were the first electronic computers. Programmable devices have existed for centuries. 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. Text editors were also developed that allowed changes and corrections to be made much more easily than with punched cards. Programs were mostly entered using punched cards or paper tape. A similar technique used for database design is Entity-Relationship Modeling (ER Modeling). After the bug is reproduced, the input of the program may need to be simplified to make it easier to debug. Some text editors such as Emacs allow GDB to be invoked through them, to provide a visual environment. The Unified Modeling Language (UML) is a notation used for both the OOAD and MDA. However, Charles Babbage had already written his first program for the Analytical Engine in 1837. Whatever the approach to development may be, the final program must satisfy some fundamental properties. 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. 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. Machine code was the language of early programs, written in the instruction set of the particular machine, often in binary notation. Integrated development environments (IDEs) aim to integrate all such help. 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. However, readability is more than just programming style. 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. Many programmers use forms of Agile software development where the various stages of formal software development are more integrated together into short cycles that take a few weeks rather than years. Techniques like Code refactoring can enhance readability.