There are many approaches to the Software development process. 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. 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. Some languages are very popular for particular kinds of applications, while some languages are regularly used to write many different kinds of applications. 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. Debugging is a very important task in the software development process since having defects in a program can have significant consequences for its users. A similar technique used for database design is Entity-Relationship Modeling (ER Modeling). 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. These compiled languages allow the programmer to write programs in terms that are syntactically richer, and more capable of abstracting the code, making it easy to target varying machine instruction sets via compilation declarations and heuristics. Some text editors such as Emacs allow GDB to be invoked through them, to provide a visual environment. After the bug is reproduced, the input of the program may need to be simplified to make it easier to debug. 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 factors, having little or nothing to do with the ability of the computer to efficiently compile and execute the code, contribute to readability. Code-breaking algorithms have also existed for centuries. After the bug is reproduced, the input of the program may need to be simplified to make it easier to debug. The first step in most formal software development processes is requirements analysis, followed by testing to determine value modeling, implementation, and failure elimination (debugging). Integrated development environments (IDEs) aim to integrate all such help. 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. However, readability is more than just programming style. Use of a static code analysis tool can help detect some possible problems. Popular modeling techniques include Object-Oriented Analysis and Design (OOAD) and Model-Driven Architecture (MDA). New languages are generally designed around the syntax of a prior language with new functionality added, (for example C++ adds object-orientation to C, and Java adds memory management and bytecode to C++, but as a result, loses efficiency and the ability for low-level manipulation). There are many approaches to the Software development process.