Following a consistent programming style often helps readability. Code-breaking algorithms have also existed for centuries. 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. However, readability is more than just programming style. Programmers typically use high-level programming languages that are more easily intelligible to humans than machine code, which is directly executed by the central processing unit. 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. Implementation techniques include imperative languages (object-oriented or procedural), functional languages, and logic languages. In 1801, the Jacquard loom could produce entirely different weaves by changing the "program" – a series of pasteboard cards with holes punched in them. By the late 1960s, data storage devices and computer terminals became inexpensive enough that programs could be created by typing directly into the computers. Programmers typically use high-level programming languages that are more easily intelligible to humans than machine code, which is directly executed by the central processing unit. Implementation techniques include imperative languages (object-oriented or procedural), functional languages, and logic languages. 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. 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. Normally the first step in debugging is to attempt to reproduce the problem. 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. There are many approaches to the Software development process. 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. He gave the first description of cryptanalysis by frequency analysis, the earliest code-breaking algorithm. The first compiler related tool, the A-0 System, was developed in 1952 by Grace Hopper, who also coined the term 'compiler'. High-level languages made the process of developing a program simpler and more understandable, and less bound to the underlying hardware. There are many approaches to the Software development process. 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. Debugging is a very important task in the software development process since having defects in a program can have significant consequences for its users.