For this purpose, algorithms are classified into orders using so-called Big O notation, which expresses resource use, such as execution time or memory consumption, in terms of the size of an input. Code-breaking algorithms have also existed for centuries. Some languages are very popular for particular kinds of applications, while some languages are regularly used to write many different kinds of applications. Implementation techniques include imperative languages (object-oriented or procedural), functional languages, and logic 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. Machine code was the language of early programs, written in the instruction set of the particular machine, often in binary notation. The first compiler related tool, the A-0 System, was developed in 1952 by Grace Hopper, who also coined the term 'compiler'. Some text editors such as Emacs allow GDB to be invoked through them, to provide a visual environment. Machine code was the language of early programs, written in the instruction set of the particular machine, often in binary notation. High-level languages made the process of developing a program simpler and more understandable, and less bound to the underlying hardware. 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. 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. The first compiler related tool, the A-0 System, was developed in 1952 by Grace Hopper, who also coined the term 'compiler'. Computer programming or coding is the composition of sequences of instructions, called programs, that computers can follow to perform tasks. Ideally, the programming language best suited for the task at hand will be selected. 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. Unreadable code often leads to bugs, inefficiencies, and duplicated code. Unreadable code often leads to bugs, inefficiencies, and duplicated code. Following a consistent programming style often helps readability. 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. Ideally, the programming language best suited for the task at hand will be selected. He gave the first description of cryptanalysis by frequency analysis, the earliest code-breaking algorithm. Normally the first step in debugging is to attempt to reproduce the problem. 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. 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.