There are many approaches to the Software development process. In the 9th century, the Arab mathematician Al-Kindi described a cryptographic algorithm for deciphering encrypted code, in A Manuscript on Deciphering Cryptographic Messages. In the 9th century, the Arab mathematician Al-Kindi described a cryptographic algorithm for deciphering encrypted code, in A Manuscript on Deciphering Cryptographic Messages. 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. Normally the first step in debugging is to attempt to reproduce the problem. Machine code was the language of early programs, written in the instruction set of the particular machine, often in binary notation. Different programming languages support different styles of programming (called programming paradigms). 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. Some languages are very popular for particular kinds of applications, while some languages are regularly used to write many different kinds of applications. Scripting and breakpointing is also part of this process. There exist a lot of different approaches for each of those tasks. High-level languages made the process of developing a program simpler and more understandable, and less bound to the underlying hardware. Popular modeling techniques include Object-Oriented Analysis and Design (OOAD) and Model-Driven Architecture (MDA). It involves designing and implementing algorithms, step-by-step specifications of procedures, by writing code in one or more programming languages. It involves designing and implementing algorithms, step-by-step specifications of procedures, by writing code in one or more programming languages. However, readability is more than just programming style. There are many approaches to the Software development process. 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. The first step in most formal software development processes is requirements analysis, followed by testing to determine value modeling, implementation, and failure elimination (debugging). The first compiler related tool, the A-0 System, was developed in 1952 by Grace Hopper, who also coined the term 'compiler'. 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. 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. 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. Readability is important because programmers spend the majority of their time reading, trying to understand, reusing and modifying existing source code, rather than writing new source code.