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..  
In the 9th century, the Arab mathematician Al-Kindi described a cryptographic algorithm for deciphering encrypted code, in A Manuscript on Deciphering Cryptographic Messages.  
Expert programmers are familiar with a variety of well-established algorithms and their respective complexities and use this knowledge to choose algorithms that are best suited to the circumstances.  
However, Charles Babbage had already written his first program for the Analytical Engine in 1837.  
 Programs were mostly entered using punched cards or paper tape.  
 High-level languages made the process of developing a program simpler and more understandable, and less bound to the underlying hardware.  
 Machine code was the language of early programs, written in the instruction set of the particular machine, often in binary notation.  
Also, specific user environment and usage history can make it difficult to reproduce the problem.  
There exist a lot of different approaches for each of those tasks.  
Unreadable code often leads to bugs, inefficiencies, and duplicated 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.  
However, with the concept of the stored-program computer introduced in 1949, both programs and data were stored and manipulated in the same way in computer memory.  
 Different programming languages support different styles of programming (called programming paradigms).  
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.