Unreadable code often leads to bugs, inefficiencies, and duplicated code..  
This can be a non-trivial task, for example as with parallel processes or some unusual software bugs.  
 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.  
 High-level languages made the process of developing a program simpler and more understandable, and less bound to the underlying hardware.  
Many programmers use forms of Agile software development where the various stages of formal software development are more integrated together into short cycles that take a few weeks rather than years.  
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.  
Sometimes software development is known as software engineering, especially when it employs formal methods or follows an engineering design process.  
  
The first compiler related tool, the A-0 System, was developed in 1952 by Grace Hopper, who also coined the term 'compiler'.  
Techniques like Code refactoring can enhance readability.  
 Different programming languages support different styles of programming (called programming paradigms).  
When debugging the problem in a GUI, the programmer can try to skip some user interaction from the original problem description and check if remaining actions are sufficient for bugs to appear.  
FORTRAN, the first widely used high-level language to have a functional implementation, came out in 1957, and many other languages were soon developed—in particular, COBOL aimed at commercial data processing, and Lisp for computer research.  
 In the 1880s, Herman Hollerith invented the concept of storing data in machine-readable form.  
 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.  
 Debugging is a very important task in the software development process since having defects in a program can have significant consequences for its users.