The first compiler related tool, the A-0 System, was developed in 1952 by Grace Hopper, who also coined the term 'compiler'..  
 Code-breaking algorithms have also existed for centuries.  
This can be a non-trivial task, for example as with parallel processes or some unusual software bugs.  
A study found that a few simple readability transformations made code shorter and drastically reduced the time to understand it.  
Provided the functions in a library follow the appropriate run-time conventions (e.g., method of passing arguments), then these functions may be written in any other language.  
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.  
 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.  
Many applications use a mix of several languages in their construction and use.  
Proficient programming usually requires expertise in several different subjects, including knowledge of the application domain, details of programming languages and generic code libraries, specialized algorithms, and formal logic.  
 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 text editors such as Emacs allow GDB to be invoked through them, to provide a visual environment.  
Methods of measuring programming language popularity include: counting the number of job advertisements that mention the language, the number of books sold and courses teaching the language (this overestimates the importance of newer languages), and estimates of the number of existing lines of code written in the language (this underestimates the number of users of business languages such as COBOL).  
Use of a static code analysis tool can help detect some possible problems.  
It is usually easier to code in "high-level" languages than in "low-level" ones.  
Scripting and breakpointing is also part of this process.