Code-breaking algorithms have also existed for centuries..  
A study found that a few simple readability transformations made code shorter and drastically reduced the time to understand it.  
 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.  
Ideally, the programming language best suited for the task at hand will be selected.  
 Whatever the approach to development may be, the final program must satisfy some fundamental properties.  
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.  
 Implementation techniques include imperative languages (object-oriented or procedural), functional languages, and logic languages.  
They are the building blocks for all software, from the simplest applications to the most sophisticated ones.  
However, because an assembly language is little more than a different notation for a machine language, two machines with different instruction sets also have different assembly languages.  
 High-level languages made the process of developing a program simpler and more understandable, and less bound to the underlying hardware.  
Integrated development environments (IDEs) aim to integrate all such help.  
 The first step in most formal software development processes is requirements analysis, followed by testing to determine value modeling, implementation, and failure elimination (debugging).  
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.  
  
The first compiler related tool, the A-0 System, was developed in 1952 by Grace Hopper, who also coined the term 'compiler'.