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..  
 High-level languages made the process of developing a program simpler and more understandable, and less bound to the underlying hardware.  
 Code-breaking algorithms have also existed for centuries.  
By the late 1960s, data storage devices and computer terminals became inexpensive enough that programs could be created by typing directly into the computers.  
It is usually easier to code in "high-level" languages than in "low-level" ones.  
  
For example, when a bug in a compiler can make it crash when parsing some large source file, a simplification of the test case that results in only few lines from the original source file can be sufficient to reproduce the same crash.  
 Programmable devices have existed for centuries.  
 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.  
 After the bug is reproduced, the input of the program may need to be simplified to make it easier to debug.  
 The first computer program is generally dated to 1843, when mathematician Ada Lovelace published an algorithm to calculate a sequence of Bernoulli numbers, intended to be carried out by Charles Babbage's Analytical Engine.  
He gave the first description of cryptanalysis by frequency analysis, the earliest code-breaking algorithm.  
 New languages are generally designed around the syntax of a prior language with new functionality added, (for example C++ adds object-orientation to C, and Java adds memory management and bytecode to C++, but as a result, loses efficiency and the ability for low-level manipulation).  
Many applications use a mix of several languages in their construction and use.  
Text editors were also developed that allowed changes and corrections to be made much more easily than with punched cards.