Computer programming or coding is the composition of sequences of instructions, called programs, that computers can follow to perform tasks..  
Their jobs usually involve:  
 Although programming has been presented in the media as a somewhat mathematical subject, some research shows that good programmers have strong skills in natural human languages, and that learning to code is similar to learning a foreign language.  
 Debugging is often done with IDEs. Standalone debuggers like GDB are also used, and these often provide less of a visual environment, usually using a command line.  
Trial-and-error/divide-and-conquer is needed: the programmer will try to remove some parts of the original test case and check if the problem still exists.  
In 1801, the Jacquard loom could produce entirely different weaves by changing the "program" – a series of pasteboard cards with holes punched in them.  
Integrated development environments (IDEs) aim to integrate all such help.  
Some languages are more prone to some kinds of faults because their specification does not require compilers to perform as much checking as other languages.  
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.  
 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.  
 Implementation techniques include imperative languages (object-oriented or procedural), functional languages, and logic languages.  
 Different programming languages support different styles of programming (called programming paradigms).  
As early as the 9th century, a programmable music sequencer was invented by the Persian Banu Musa brothers, who described an automated mechanical flute player in the Book of Ingenious Devices.  
 In the 1880s, Herman Hollerith invented the concept of storing data in machine-readable form.  
 These compiled languages allow the programmer to write programs in terms that are syntactically richer, and more capable of abstracting the code, making it easy to target varying machine instruction sets via compilation declarations and heuristics.  
A study found that a few simple readability transformations made code shorter and drastically reduced the time to understand it.