It involves designing and implementing algorithms, step-by-step specifications of procedures, by writing code in one or more programming languages..  
In the 9th century, the Arab mathematician Al-Kindi described a cryptographic algorithm for deciphering encrypted code, in A Manuscript on Deciphering Cryptographic Messages.  
There exist a lot of different approaches for each of those tasks.  
 Following a consistent programming style often helps readability.  
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.  
This can be a non-trivial task, for example as with parallel processes or some unusual software bugs.  
Integrated development environments (IDEs) aim to integrate all such help.  
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.  
 Implementation techniques include imperative languages (object-oriented or procedural), functional languages, and logic languages.  
Scripting and breakpointing is also part of this process.  
 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.  
They are the building blocks for all software, from the simplest applications to the most sophisticated ones.  
In 1801, the Jacquard loom could produce entirely different weaves by changing the "program" – a series of pasteboard cards with holes punched in them.  
For example, COBOL is still strong in corporate data centers often on large mainframe computers, Fortran in engineering applications, scripting languages in Web development, and C in embedded software.