This is interpreted into machine code.  
Ideally, the programming language best suited for the task at hand will be selected.  
Compiling takes the source code from a low-level programming language and converts it into machine code.  
Use of a static code analysis tool can help detect some possible problems.  
However, while these might be considered part of the programming process, often the term software development is more likely used for this larger overall process – whereas the terms programming, implementation, and coding tend to be focused on the actual writing of code.  
To produce machine code, the source code must either be compiled or transpiled.  
It is usually easier to code in "high-level" languages than in "low-level" ones.  
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.  
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.  
He gave the first description of cryptanalysis by frequency analysis, the earliest code-breaking algorithm.  
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.  
 High-level languages made the process of developing a program simpler and more understandable, and less bound to the underlying hardware.  
Scripting and breakpointing is also part of this process.  
Many programmers use forms of Agile software development where the various stages of formal software development are more integrated together into short cycles that take a few weeks rather than years.  
One approach popular for requirements analysis is Use Case analysis.