Compiling takes the source code from a low-level programming language and converts it into machine code.  
There exist a lot of different approaches for each of those tasks.  
However, Charles Babbage had already written his first program for the Analytical Engine in 1837.  
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.  
It affects the aspects of quality above, including portability, usability and most importantly maintainability.  
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.  
Use of a static code analysis tool can help detect some possible problems.  
 In the 1880s, Herman Hollerith invented the concept of storing data in machine-readable form.  
Scripting and breakpointing is also part of this process.  
Transpiling on the other hand, takes the source-code from a high-level programming language and converts it into bytecode.  
Scripting and breakpointing is also part of this process.  
This is interpreted into machine code.  
 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.  
Later a control panel (plug board) added to his 1906 Type I Tabulator allowed it to be programmed for different jobs, and by the late 1940s, unit record equipment such as the IBM 602 and IBM 604, were programmed by control panels in a similar way, as were the first electronic computers.  
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.