They are the building blocks for all software, from the simplest applications to the most sophisticated ones.  
Transpiling on the other hand, takes the source-code from a high-level programming language and converts it into bytecode.  
It is usually easier to code in "high-level" languages than in "low-level" ones.  
Ideally, the programming language best suited for the task at hand will be selected.  
Programming involves tasks such as analysis, generating algorithms, profiling algorithms' accuracy and resource consumption, and the implementation of algorithms (usually in a particular programming language, commonly referred to as coding).  
The purpose of programming is to find a sequence of instructions that will automate the performance of a task (which can be as complex as an operating system) on a computer, often for solving a given problem.  
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.  
Techniques like Code refactoring can enhance readability.  
The source code of a program is written in one or more languages that are intelligible to programmers, rather than machine code, which is directly executed by the central processing unit.  
Techniques like Code refactoring can enhance readability.  
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.  
In 1801, the Jacquard loom could produce entirely different weaves by changing the "program" – a series of pasteboard cards with holes punched in them.  
Use of a static code analysis tool can help detect some possible problems.  
Compilers harnessed the power of computers to make programming easier by allowing programmers to specify calculations by entering a formula using infix notation.  
This can be a non-trivial task, for example as with parallel processes or some unusual software bugs.