* The control-flow statements of a language specify the order in which computations are performed.
* Braces { and } are used to group declarations and statements together into a compound statement, or block so that they are syntactically equivalent to a single statement.
* **If-Else**
  + The if-else statement is used to express decisions
  + if (expression)

statement 1

Else

statement 2

* **Else-if**
  + if (expression)

statement

else if (expression)

statement

else if (expression)

statement

else if (expression)

statement

else

statement

* **Switch**
  + The switch statement is a multi-way decision that tests whether an expression matches one of a number of constant integer values, and branches accordingly.
  + switch (expression) {

case const-expr: statements

case const-expr: statements

default: statements

* **Loops** - While, For, do-while
  + while (expression)

statement

* + for (initialization; condition; increment/decrement)

Statement

* + do

statement

while (expression)

* **Break** 
  + The break statement provides an early exit from for, while, and do, just as from switch.
  + A break causes the innermost enclosing loop or switch to be exited immediately
* **Continue** 
  + When a continue statement is encountered inside a loop, control jumps to the beginning of the loop for the next iteration, skipping the execution of statements inside the body of loop for the current iteration.
  + The continue statement applies only to loops, not to switch.
* **Goto and labels**
  + C provides the infinitely-abusable goto statement and labels to branch to.
  + The commonplace where this is used to break out two or more number of loops at a time
  + A label has the same form as a variable name and is followed by a colon. It can be attached to any statement in the same function as the goto.
* **Functions**
  + Functions break large computing tasks into smaller ones and enable people to build on what others have done instead of starting over from scratch
  + Functions can have different types of return types
  + Variable
    - External variables - defined outside a function
    - Internal variables - defined inside a function
* **Scope Rules**
  + The functions and external variables that make up a C program need not all be compiled at the same time; the source text of the program may be kept in several files, and previously compiled routines may be loaded from libraries
* **Register variables**
  + The idea is that the register variable is to be placed in machine registers, which may result in smaller and faster programs
  + Eg: register int X;

register char c

* **C Preprocessors**
  + C provides certain language facilities by means of a preprocessor, which is conceptually a separate first step in the compilation.
  + The two most frequently used features are #include, to include the contents of a file during compilation, and #define, to replace a token by an arbitrary sequence of characters.