Data Types

int:

Integers are your primary data-type for number storage.

On the Arduino Uno an int stores a 16-bit (2-byte) value. This yields a range of -32,768 to 32,767 (minimum value of -2^15 and a maximum value of (2^15) - 1).

unsigned int:

Unsigned ints (unsigned integers) are the same as ints in that they store a 2 byte value.

Instead of storing negative numbers however they only store positive values, yielding a useful range of 0 to 65,535 ((2^16) - 1).

long:

Long variables are extended size variables for number storage, and store 32 bits (4 bytes), from -2,147,483,648 to 2,147,483,647.

unsigned long:

Unsigned long variables are extended size variables for number storage, and store 32 bits (4 bytes). Unlike standard longs unsigned longs won't store negative numbers, making their range from 0 to 4,294,967,295 (2^32 - 1).

float:

Datatype for floating-point numbers, a number that has a decimal point.

Floating-point numbers can be as large as 3.4028235E+38 and as low as -3.4028235E+38. They are stored as 32 bits (4 bytes) of information.

The float data type has only 6-7 decimal digits of precision.

double:

Double is same as float, just it has more precision.

Unlike other platforms, where you can get more precision by using a double (e.g. up to 15 digits), on the Arduino, double is the same size as float.

char:

A data type used to store a character value. Character literals are written in single quotes, like this: 'A' (for multiple characters - strings - use double quotes: "ABC").

The size of the char datatype is at least 8 bits.

byte:

A byte stores an 8-bit unsigned number, from 0 to 255.

string:

Text strings can be represented by making a string out of an array of type char and null-terminate it.

All of the following are valid declarations for strings:

```
char Str1[15];

char Str2[8] = {'a', 'r', 'd', 'u', 'i', 'n', 'o'};

char Str3[8] = {'a', 'r', 'd', 'u', 'i', 'n', 'o', '\0'};

char Str4[] = "arduino";

char Str5[8] = "arduino";

char Str6[15] = "arduino";
```

Strings are terminated with a null character (ASCII code 0). This allows functions to tell where the end of a string is.

const:

The const keyword stands for constant. It is a variable qualifier that modifies the behavior of the variable, making a variable "read-only". This means that the variable can be used just as any other variable of its type, but its value cannot be changed.

static:

The static keyword is used to create variables that are visible to only one function. However unlike local variables that get created and destroyed every time a function is called, static variables persist beyond the function call, preserving their data between function calls.

Variables declared as static will only be created and initialized the first time a function is called.

volatile:

volatile is a keyword known as a variable qualifier, it is usually used before the datatype of a variable, to modify the way in which the compiler and subsequent program treat the variable.

Specifically, it directs the compiler to load the variable from RAM and not from a storage register, which is a temporary memory location where program variables are stored and manipulated.

A variable should be declared volatile whenever its value can be changed by something beyond the control of the code.

Overflow/Underflow: When signed variables are made to exceed their maximum or minimum capacity they overflow. The result of an overflow is unpredictable so this should be avoided.

For Reference: https://www.arduino.cc/reference/en/ (Variable Section)