**Data Type**

**Data Type**: data type describes that *what type of data* a *variable can store*.

Variables in C# are categories into 3 types.

1. Value type
2. Reference type
3. Pointer type

|  |  |  |
| --- | --- | --- |
| Value Type | Reference Type | Pointer Type |
| In value type a variable can be assigned a value directly | Reference types do not contain the actual data stored in a variable, but they contain a reference to the variables | Pointer type variables store the memory address of another type |
| Ex: int, char, float, double, string etc | Ex: build in reference types are   1. Object 2. Dynamic 3. String | Ex: type\* identifier  char\* cptr  int\* pointer |

**C# Build in Data types**

|  |  |  |
| --- | --- | --- |
| **Type** | **Represents** | **Default** |
| Value type |  |  |
| * bool | Boolean value ( True or False) | false |
| * char | 16-bit Unicode character | ‘\0’ |
| * byte | 8 bit unsigned integer (0 to 255) | 0 |
| * sbyte | 8-bit signed integer type (-128 to 127) | 0 |
| * short | 16 bit signed integer (-128 to 127) | 0 |
| * ushort | 16 bit unsigned integer(0 to 65,535) | 0 |
| * int | 32-bit signed integer type | 0 |
| * uint | 32-bit unsigned integer type | 0 |
| * long | 64-bit signed integer type |  |
| * ulong | 64-bit unsigned integer type |  |
| * float | 32-bit single-precision floating point type | 0.0F |
| * double | 64-bit double-precision floating point type | 0.0D |
| * decimal | 128-bit precise decimal values with 28-29 significant digits | 0.0M |

|  |  |  |
| --- | --- | --- |
| Reference Type | Represents | Syntax |
| * string | A string of Unicode characters. It allows to assign any string to a variable | string a =”abcd” |
| * object | An object type can be assigned a values of any other value type, reference type, predefined or user defined type | **Boxing** -> converting value type to object type.  **Un-Boxing** -> converting object type to value type |
| * dynamic | Can store any type of value in dynamic data type variable. Type checking of this variable is done at run time whereas for object type checking is done at compile time | Ex: dynamic d = 30; |

**Properties, Variables and methods**

**Variables**: variables are just the name of the field which is used to store a data type value.

**Properties**: properties represents the data of a class. Properties are the accessors that reads, writes or manipulate the values of a class.

**Methods**: methods defines the operations that are performed by a class.

**Arithimatic operations**

Addition, subtraction, division, multiplication, modulus