**Data Types & Operators**

**Data Types**

Data Type is a set of allowable values. It defines type of data and size of memory required. C#.net has following of categories of data types.

**Value Types**

Value Type store its contents in memory allocated in stack

**Reference Types**

Reference Type stores the address of memory where data stored. Reference Type store the data in heap memory.

**Pointer** **Types** ( it is unsafe and not advised to use)

**Value Types**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SNO** | **Data Type** | **Size** | **Range** | **Default Value** | **Framework Type** |
| 1 | Int | 4 bytes | -2,147,483,648 to 2,147,483,647 | 0 | System.Int32 |
| 2 | Uint | 4 bytes |  | 0 | System.UInt32 |
| 3 | short | 2 bytes | -32,768 to 32,767 | 0 | System.Int16 |
| 4 | ushort | 2 bytes | 0 to 65,535 | 0 | System.UInt16 |
| 5 | long | 8bytes | –9,223,372,036,854,775,808 to 9,223,372,036,854,775,807 | 0 | System.Int64 |
| 6 | ulong | 8bytes | 0 to 18,446,744,073,709,551,615 | 0 | System.UInt64 |
| 7 | byte | 1 byte | 0 to 255 | 0 | System.Byte |
| 8 | sbyte | 1 byte | -128 to 127 |  | System.SByte |
| 9 | float | 4 bytes |  | 0.0F | System.Single |
| 10 | double | 8 bytes |  | 0.0 | System.Double |
| 11 | decimal | 16 Bytes |  | 0 | System.Decimal |
| 12 | bool | 1 Byte |  | False |  |
| 13 | Struct |  |  |  |  |
| 14 | char | 2 Bytes |  | ‘\0’ | System.Char |
| 15 | enum |  |  |  |  |

**Reference Types**

1. class
2. interface
3. delegate
4. dynamic
5. string
6. object

**Pointer Types**

Pointer Type stores the address of another variable. Pointer Types can be used in unsafe context.

Ex;

int\* p1,p2;

int a=23;

unsafe

{

p1=&a;

}

Any of the following types may be a pointer type:

* sbyte, byte, short, ushort, int, uint, long, ulong, char, float, double, decimal, or bool
* Any **enum** type.
* Any pointer type.
* Any user-defined struct type that contains fields of unmanaged types only.

To make run the unsafe code in c#.net, turn on ‘allow unsafe’ check box in visual studio 2012 under build properties (Project->Properties->Build).

**Variable Declaration**

Syntax: <datatype> <variablename>;

int a =2;

int b=3,c=4;

float f=2.3F; ( float f=2.3 will throw an error. In c# all decimal points will be treated as double. Hence float value has to be suffixed with F or f)

double b=2.45;

decimal c=2.6M;(decimal values should be suffixed with M or m)

string name=”sateesh”;

**Operators**

|  |  |
| --- | --- |
| Arithmetic Operators | +, -, \*, / |
| Assignment Operator | = |
| Arithmetic Assignment Operators | +=, -=, \*=, /= |
| Relational Operators | >,<,>=,<=, !=, == |
| Logical Operators | && (and), || (OR), ! (NOT) |
| Ternary Operator | ?: |
| Increment Operator | ++ |
| Decrement Operator | -- |

**Examples**:-

class Program

{

static void Main(string[] args)

{

int a = 3;

int b = a ^ 2;

int c = a + b;

string s1 = "Jeorge";

string s2 = "Bush";

string s3 = s1 + " " + s2;

Console.Write("Values of variable: a={0},b={1},c={2},s1={3},s2={4},s3={5}",a,b,c,s1,s2,s3);

a+=2; // this is equivalent to a=a+b;

}

}

Output : Values of variable: a=34,b=102,c=56,s1=Jeorge,s2=Bush,s3=Jeorge Bush

Note: + operator on strings concatenates the strings and on numeric data will add. + operator cannot be applied ob byte arguments.

**Examples on Ternary Operator**

int a=23;b=35

int big=a>b?a:b;

int a=34, b=102, c=56;

int big=a>b && a>c?a:(b>c)?b:c;

**Examples on Increment and Decrement Operator**

int a=105;

int b=a++; (post increment)

value of b is 105 and value a is 106;

int a=105;

int b=++a; (pre increment)

value of b is 106 and value a is 106;