Operators

C# language supports a rich set of built-in operators. An operator is a symbol that tells the Web Server scripts engine to perform certain mathematical or logical manipulations. Operators are used in program to manipulate data and variables. C# operators are binary, unary and ternary operators. A unary operator required one operand, and binary operator required two operands between the one operators. List of main types of C# operators. C# operators can be classified into following operation types.

S.No.	Operator Type	Description
1	Arithmetic	Performs mathematical calculations.
2	Increment/Decrement	One-by one-increment or decrement arithmetic operations.
3	Comparison/Relational	Compares operands and return a Boolean value
4	Logical	Performs Boolean operations on Boolean operands
5	Assignment	Assign values to variables.
6	Conditional	Conditional operator is use to apply if .. else condition in single line form and result produced in Boolean true or false.
7	Bitwise	Bitwise operators perform manipulations of data at **bit level**.
8	Special	Performs various task special things

1. Arithmetic operators: C# supports all the basic arithmetic operators.

Symbol	Description
+	adds two operands
-	subtract second operands from first
*	**multiply two operand**
/	divide numerator by denominator
%	remainder of division

2. Increment/Decrement operators: C# supports all the basic increment/decrement arithmetic operators.

Symbol	Description
++	Increment operator increases integer value by one
--	Decrement operator decreases integer value by one

3. Relation operators: The following table shows all relation operators.

Operator	Description
==	Check if two operand are equal
!=	Check if two operands are not equal.
>	Check if operand on the left is greater than operand on the right
<	Check operand on the left is smaller than right operand
>=	check left operand is greater than or equal to right operand
<=	Check if operand on left is smaller than or equal to right operand

4. Logical operators: C# language supports following 3 logical operators. Suppose a=1 and b=0,

Operator	Description	Example
&&	Logical AND	(a && b) is false
||	Logical OR	(a || b) is true
!	Logical NOT	(!a) is false

5. Assignment Operators: Assignment operators supported by C#.

Operator	Description	Example
=	assigns values from right side operands to left side operand	a=b
+=	adds right operand to the left operand and assign the result to left	a+=b is same as a=a+b
-=	subtracts right operand from the left operand and assign the result to left operand	a-=b is same as a=a-b
*=	multiply left operand with the right operand and assign the result to left operand	a*=b is same as a=a*b
/=	divides left operand with the right operand and assign the result to left operand	a/=b is same as a=a/b
%=	calculate modulus using two operands and assign the result to left operand	a%=b is same as a=a%b

6. Special operator: operator are use in some special cases.

Operator	Description
sizeof	The ***sizeof*** is a unary operator that returns the size of data (constants, variables, array, structure, etc).
[]	Accesses an element of an array
->	Specified the structure to pointer
.	Dot operator is use to structure member variable.
*	This is a pointer operator

7. Bitwise operators: Bitwise operators perform manipulations of data at **bit level**. These operators also perform **shifting of bits** from right to left. Bitwise operators are not applied to **float** or **double,** only apply Integer value.

Operator	Description
&	Bitwise AND
|	Bitwise OR
^	Bitwise exclusive OR(XOR)
<<	left shift
>>	right shift

Truth table for bitwise **& , | and ^**

a	B	a & b	a | b	a ^ b
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

The bitwise shift operators shift the bit value. The left operand specifies the value to be shifted and the right operand specifies the number of positions that the bits in the value are to be shifted. Both operands have the same precedence.

8. **Conditional operators:** It is also known as ternary operator and used to evaluate conditional expression.

Operator	Description
( )? :	Executes one of two expression based on the results of a conditional expression.
(Expre.)?Expe2 : Expre3;	If **Expre1 Condition** is true? value **Expre2**:Otherwise value **Expre3;**

Preceding of operator

Operator precedence determines the order in which operators are evaluated. Operators with higher precedence are evaluated first. Operator precedence refers to the order in which operators execute within an expression.

The precedence of an operator specifies how "tightly" it binds two expressions together. For example, in the expression 1 + 5 * 3, the answer is 16 and not 18 because the multiplication ("*") operator has a higher precedence than the addition ("+") operator. Parentheses may be used to force precedence, if necessary. For instance: (1 + 5) * 3 evaluates to 18.

Use of parentheses, even when not strictly necessary, can often increase readability of the code by making grouping explicit rather than relying on the implicit operator precedence and associativity. The following table lists the C# operators, ordered from highest to lowest precedence.

Operator (symbols)	Type Operator	Associativity
++ --	Increment and decrement	Right to left
!	Logical Not	Right to Left
/ * %	Multiply , division and modulus	Left to right
+ - .	Addition , subtraction and concatenation string	Left to Right
== <= >= > < != <>	Relational or comparisons	None
&& ||	Logical And , Or	Left to Right
= += -= *= /= %=	Assignment operator	Right to left

Expressions Builds

An **expression** in a programming language is a combination of one or more constants, variables, operators, and functions that the programming language interprets and is put together in a single statement. Variable and data become most useful when you use them in an expression. You use operands and operators to create expression in C#.

```
using System;

namespace OperatorsClass1
{
   class Program
   {
      static void Main(string[] args)
      {
         int age,days;
         age = 25;
         days =  (age * 365 )  ;          // this is  Expression  result in variable days @location
        Console.WriteLine (" Age is  "+age +" Years and Days are "+days);

      }
   }
}
```

$$\underbrace{days}_{\text{Variable}}\ \overbrace{=}^{\text{Operator}}\ (\ \underbrace{\$age}_{\text{Variable}}\ \overbrace{*}^{\text{Operator}}\ \underbrace{365}_{\text{constant}}\ )\ .$$

Exercise

Theory Questions

1. What is Operator and describe of relational operators with symbols.
2. What difference between the binary and unary operators?
3. What do you mean by preceding Operators?
4. What do you mean by BODMAS operation?
5. What difference between the (=, ==) operators.

Practical Questions

1. Write a simple program to calculate total numbers of days you lived up to until now

   Hint: Input your age (in years) and then calculate number of days

   number _of _days= (age in years) * 365

2. Write a code to calculate and print percentage of a student where, Total_Marks=400, obtained_Marks=Sum of obtained marks of all 4 subjects Percentage= (Obtained_Marks*100)/Total_Marks. (Hint: input marks of four subjects out of 100).

3. Write a code to input Temperature in Centigrade and convert to Fahrenheit.
4. If A= 34 and B=55 then what will be output of the following Exercise; Also Justify your answer

   a) A= = 34 && B = = 55　　b) A>=30 || B<=50　　c) B= = 55 || A= =35

   d) A != 34　　e) A>=30 && A<35　　f) B>50 || B<56

5. Mention the output in memory of the following code at each step and display final result on the screen.

```
int a,b,c;                                   Output
 a = 33;    b = 55;                 ____________________
a += b;                             ____________________
c =  a;                             ____________________
c -=  b;                            ____________________
c *=  a;    a++;                    ____________________
                                    ____________________

Console.WriteLine( " a= "+a+"  b=  "+ b+" c="+c);
```

6. What is the result after the execution of the following program segment?

```
static void main()
{
    int a =  8,b,c;
    b = ++a  + 5;
   c = b--  + 10;
 Console.WriteLine( " a= "+a+"  b=  "+ b+" c="+c);
}
```

7. Write down the output of following code and also justify your answer
    a) Console.WriteLine (4 + 2 - 12 * 3) ; //what will be output
    b) Console.WriteLine (4 + (2 – 12) * 3) ; //what will be output
    c) Console.WriteLine (((4 + 2) - 12) * 3); //what will be output
8. What value is assigned to ***ReturnValue*** for each of the following expression?
    a) ReturnValue = 2 == 3; b) ReturnValue = "2" + "3";
    c) ReturnValue = 2 > 3; d) ReturnValue = 2 < 3;
    e) ReturnValue = (2 > 3) && (2 < 3) ; f) ReturnValue = (2 > 3) | | (2 < 3)
9. Write down the "size of" each constant/variable also justify your answer
    a) int str=55; b) float chr=5.666; d) double d =54;
    e) char it[70]= "CIT"; f) char it='F'; g) char c[100];
10. Write down the output of following code and also justify your answer

A

```
static void Main(string[] args)
{
    int a=3;
    Console.WriteLine (a);
    Console.WriteLine (a++);
    Console.WriteLine (a);
    Console.WriteLine(++a);
}
```

B

```
static void Main(string[] args)
{
    int a=3;
    Console.WriteLine (a);
    Console.WriteLine (a- -);
    Console.WriteLine (a);
    Console.WriteLine (- -a);
}
```

Objective MCQ's

1. What is the value of the expression 4 * 2 + 3?
    a) 11
    b) -11
    c) 20
    d) 24
2. The logical And (&&) operator return TRUE if __________
    a) The left operand return a value is true
    b) The right operand return a value is true
    c) The left and right operand both return a value true is true
    d) The left and right operand both return a value true is false
3. Which arithmetic operator can be used as both prefix and postfix operators?
    a) ++
    b) –
    c) +
    d) *

4. Which of the following values can be assigned to a Boolean variable?
   a) 0 or 1
   b) true
   c) false
   d) Yes
5. What value is assigned to the ***ReturnValue*** variable in the statement ***ReturnValue*** = 100 != 200 ;
   a) 0
   b) 1
   c) 100
   d) 200
6. Which of the following in an example of initializing a variable.
   a) num= 2;
   b) num >= 2;
   c) num <= 2;
   d) num == 2;
7. A relational operator is used to
   a) Combine the values
   b) Distinguish different types of variables.
   c) Change variables to logical values
   d) Compare the values and result will be Boolean
8. Operators are used to perform some operation on the ________.
   a) Operands
   b) Function
   c) Variables
   d) Constant
9. Conditional Operators has ________.
   a) One operand
   b) Two operands
   c) Three operands
   d) Four operands
10. You use operands and operators to create_______in C#.
    a) variables
    b) functions
    c) Expression
    d) constant