

1)Static class

```
using System;
public static class MyMath
{
    public static float PI=3.14f;
    public static int cube(int n){return n*n*n;}
}
class TestMyMath{
    public static void Main(string []args)
    {
        Console.WriteLine("Value of PI is:"+MyMath.PI);
        Console.WriteLine("Cube of 3 is :" + MyMath.cube(3));
    }
}
```

2)Abstract class

```
using System;
abstract class Animal
{
    public abstract void animalSound();
    public void sleep()
    {
        Console.WriteLine("Zzz..");
    }
}
class pig:Animal{
    public override void animalSound()
    {
        Console.WriteLine("Pig's sound : wee weee...");
    }
}
```

```
class Program
{
    static void Main(string []args)
    {
        pig p = new pig();
        p.animalSound();
        p.sleep();
    }
}
```

3)DLL

```
//class library project
using system;
namespace geeksforgeeks
{
    public class GFG
    {
        public void sayHello()
        {
            Console.WriteLine("Hello From GeeksForGeeks");
        }
    }
}

//console
using geeksforgeeks;
class Test
{
    public static void Main(string[] args)
    {
        Class1 ob = new Class1();
        ob.sayHello(); 
    }
}
```

```
4)IPAddress

//Console

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Net;

namespace ipAddress
{
    internal class program
    {
        static void Main(string[] args)
        {
            string hostName= Dns.GetHostName();
            Console.WriteLine(hostName);
            string myIP = Dns.GetHostByName(hostName).AddressList[0].ToString();
            Console.WriteLine("My IP Address is :" + myIP);
            Console.ReadKey();
        }
    }
}
```

```
5)Polymorphism

using System;
class Animal
{
    public void animalSound()
    {
        Console.WriteLine("The animal makes a sound");
    }
}
```

```
class pig:Animal
{
    public void animalSound()
    {
        Console.WriteLine("Pig's sound : wee weee...");
    }
}

class dog:Animal
{
    public void animalSound(){
        Console.WriteLine("Dog's sound : bow wow...");
    }
}

class Program
{
    static void Main(string [] args){
        Animal a = new Animal();
        Animal p = new pig();
        Animal d = new dog();
        a.animalSound();
        p.animalSound();
        d.animalSound();

    }
}

6)Regex

using System;
using System.Text.RegularExpressions;
class GFG
{
```

```

static void Main(string[] args)
{
    string[] str = { "9405065173", "8956605273", "02378263833" };
    foreach (string s in str)
    {
        Console.WriteLine("{0} {1} a valid mobile number.", s, isValidMobileNumber(s) ? "is" : "is not");
    }
    Console.ReadKey();
}

public static bool isValidMobileNumber(string inputMobileNumber)
{
    string strRegex = @"(^[0-9]{10}$) | 
(^[0-9]{2}\s+[0-9]{2}[0-9]{8}$) |
(^[0-9]{3}-[0-9]{4}$)";

    Regex re = new Regex(strRegex);
    if (re.IsMatch(inputMobileNumber))
        return (true);
    else
        return (false);
}

```

7)String Builder

```

using System;
using System.Text;
class GFG
{
    //Append Method
    public static void Main(string[] args)

```

```
/* {  
    StringBuilder s = new StringBuilder("HELLO",20);  
    s.Append("GFG");  
    s.AppendLine("GEEKS");  
    s.Append("GeeksForGeeks");  
    Console.WriteLine(s);  
}  
 */  
//AppendFormat  
/*{  
    StringBuilder s = new StringBuilder("Your Total amount is ");  
    s.AppendFormat("{0:C}", 50);  
    Console.WriteLine(s);  
}  
 */  
//StringBuilder.Insert  
/* {  
    StringBuilder s = new StringBuilder("HELLO", 20);  
    s.Insert(6, "GEEKS");  
    Console.WriteLine(s);  
}  
 */  
//StringBuilder.Remove  
/*{  
    StringBuilder s = new StringBuilder("GeeksForGeeks",20);  
    s.Remove(5, 3);  
    Console.WriteLine(s);  
}  
 */  
//StringBuilder.Replace  
{  
    StringBuilder s = new StringBuilder("GFG Geeks",20);  
    s.Replace("GFG", "Geeks For");  
    Console.WriteLine(s);  
}  
}
```

8)String Manipulation

```
using System;

namespace ConsoleApp2
{
    internal class Program
    {
        static void Main(string[] args)
        {
            string a = "      C# Program      ";
            string b = "C# Programming";

            Console.WriteLine("length of string a is " + a.Length);
            Console.WriteLine("length of string b is " + b.Length);

            string str = string.Concat(b, a, b, a);

            Console.WriteLine("Concatenated string are : " + str);
            Console.WriteLine();

            Boolean result = a.Equals(b);
            if (result == true)
            {
                Console.WriteLine("result of equal string is true");
            }
            else
            {
                Console.WriteLine("result of equal string is False");
            }
            Console.WriteLine();

            Console.WriteLine("lower case : " + a.ToLower());
            Console.WriteLine("Upper case : " + a.ToUpper());

            Console.WriteLine("is b contains : " + b.Contains("Prog"));

            Console.WriteLine("is b starts with : " + b.StartsWith("C#"));

            Console.WriteLine("is b ends with : " + b.EndsWith("ing"));

            Console.WriteLine("index of : " + a.IndexOf("r"));
            Console.WriteLine("LastIndexOf : " + a.LastIndexOf("r"));

            Console.WriteLine(b.Remove(4));
        }
    }
}
```

```
        Console.WriteLine(b.Replace("r", "z"));
        Console.WriteLine(a.Substring(3, 2));

        Console.WriteLine(a.Trim());
    }
}
```

10) Inheritance

```
using System;

// single inheritance class Animal
{
    public void Eat()
    {
        Console.WriteLine("Animal is eating.");
    }
}

class Dog : Animal
{
    public void Bark()
    {
        Console.WriteLine("Dog is barking.");
    }
}

// multi-level inheritance class Mammal : Animal
{
    public void Run()
    {
        Console.WriteLine("Mammal is running.");
    }
}

class Horse : Mammal
{
    public void Gallop()
    {
        Console.WriteLine("Horse is galloping.");
    }
}
```

```
}

}

// hierarchical inheritance class Bird : Animal

{ public void Fly() { Console.WriteLine("Bird is flying."); }

}

class Eagle : Bird

{

    public void Hunt()

    {

        Console.WriteLine("Eagle is hunting.");

    }

}

class Penguin : Bird

{

    public void Swim()

    {

        Console.WriteLine("Penguin is swimming.");

    }

}

// multiple inheritance interface I1

{ void Method1();

}

interface I2

{

    void Method2();

}

class MyClass : I1, I2

{

    public void Method1()

    {
```

```
Console.WriteLine("Method1 is called.");
}

public void Method2()
{
    Console.WriteLine("Method2 is called.");
}

}

// main program class Program

{
    static void Main(string[] args)
    {
        // single inheritance Dog dog = new Dog();
        dog.Eat(); dog.Bark();

        // multi-level inheritance Horse horse = new Horse();
        horse.Eat(); horse.Run(); horse.Gallop();

        // hierarchical inheritance Eagle eagle = new Eagle();
        Penguin penguin = new Penguin();

        eagle.Fly();
        eagle.Hunt();
        penguin.Fly();
        penguin.Swim();

        // multiple inheritance
        MyClass myClass = new MyClass();
        myClass.Method1();
        myClass.Method2();
        Console.ReadLine();
    }
}
```