

.Net Framework Architecture

Required to learn

1. *Datatype*
2. *If/else*
3. *For/else*
4. *Operators*
5. *Conversion/Memory allocation*

For: Initialization start from in between for loop. Line termination done at initialization.

While:

1. *Print at runtime (declare in between runtime).*
2. *Work fast with comparison to for loop.*
3. *Initialization after scope.*

Do While: in do while, if condition become fail, then at least one time it checks condition & work(while vs do while).

Line Termination (;): It done in between {} **scope** only not before or after brackets.

Ques. Print 4 time + AND *.

Int a=0,b;

While(a<4)

{ for(a=0;a

class class2

output

Hello (upto 10)

```

9 { int a=0;
   while(a<10)
10 { Console.WriteLine("Hello");
    a++;
11 } }

```

welcome

12 Q Do-while

```

do { c
1 { Console.WriteLine("Hello");
  a++;
2 }
  while(a>10)
3 { }

```

output

Hello (one time)

4 Flow

Ex:-

```

5 int a=0;
  while(a<10)
  {
6   if(a==5)
     { Console.WriteLine("Good");
7   } else if(a==7)
     { Console.WriteLine("Hello India");
       }
     else { Console.WriteLine("Hello");
           }
           a++;
           }
           Console.ReadLine();
           } } }

```

Output

```

Hello
↓
Hello
↓
Good
↓
Hello
↓
Hello India
↓
Hello

```

upto 5
upto 7

29-Jan-2016

Collection: using array indexing in Array.

Array: Collection of similar datatypes.

```
String[] aa=new string[5];
```

↓ ↓
First indexing Indexing number

```
Aa[0]="abcd";
```

```
Aa[1]="xyz";
```

```
Aa[3]="hello";
```

```
Aa[4]="123";
```

Note: 0: Indexing starts from Zero.

1: Count or length starts from one.

Drawback: **i)** Similar datatypes.

ii) Size is fixed; we can't increase the size.

Implicit Conversion : if data match then it may convert into another datatype.

Ex: <input checked="" type="checkbox"/> string a="101";		<input checked="" type="checkbox"/> string a="xyz";
Int b=int.Parse(a);		

Difference between Convert.ToInt32 VS. int.Parse().

Convert.ToInt32	int.Parse().
➡ Is a class type	is a struct type
➡ Memory allocation done at runtime.	Struct is user defined and memory allocation done at compiles time.
➡ Accept Null value.	Never accept null value
➡ It utilize the memory.	It wastages the extra memory space.

Object: object classhold any type of data type values. Objects have permission for execution work on real world entity.

Class: Collection of items, datamembers.

Ex. Object a=100;

Object b="abcd";

Object b=123.45;

01-Feb-2016

DateTime and TimeSpan

DateTime: The DateTime value type represents dates and times with values ranging from 00:00:00 midnight, in the Gregorian calendar.

TimeSpan: A timespan object represents a time interval (duration of time or elapsed time) that is measured as a positive or negative number of days, hours, minutes, seconds, and a fraction of a second. The timespan structure can also be used to represent the time of day. But only if the time is unrelated to a particular date.s1

DateTime.Now.ToString();
↓ ↓ ↓
Structure Property Method

To print Date only

```
Console.WriteLine(DateTime.Now.ToShortDateString());
```

To print Time only

```
Console.WriteLine(DateTime.Now.ToShortTimeString());
```

Customize the datetime format as per Indian time format

```
//Console.WriteLine(DateTime.Now.ToString("dd MMMM yyyy"));
```

Note: Here we can use "/" and "-". i.e. 01/02/2016.

Capital MM ->Numeric month

Capital MMM ->First 3 letters or character of month. i.e. 02-Feb-2016.

Capital MMMM ->Full name of month. i.e. February

To print Hour and Mintues

```
//Console.WriteLine(DateTime.Now.ToString("dd MMMM yyyy hh:mm:ss tt"));
```

Note: We can't edit or adjust the time format. It will be taken as per system.

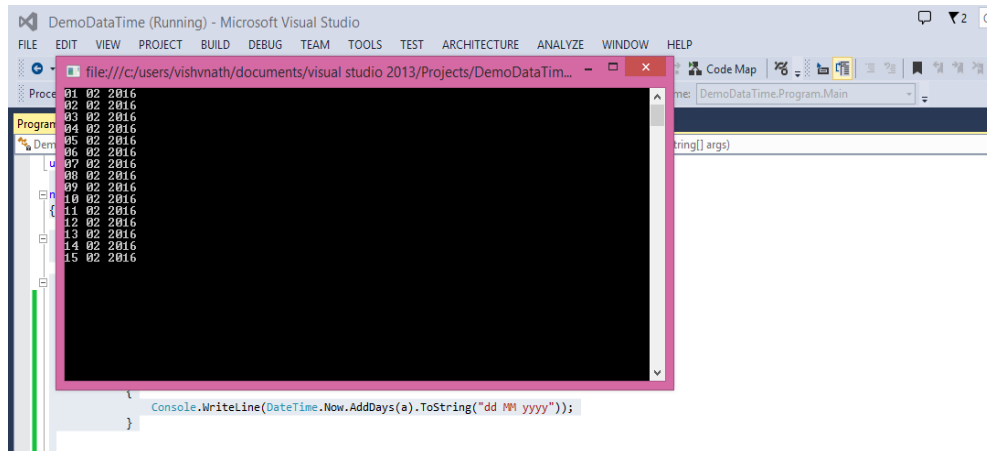
To print day after 15 days

```
for (int a = 0; a < 15; a++)  
{
```

```

    Console.WriteLine(DateTime.Now.AddDays(a).ToString("dd MM yyyy"));
}
Console.ReadLine();

```



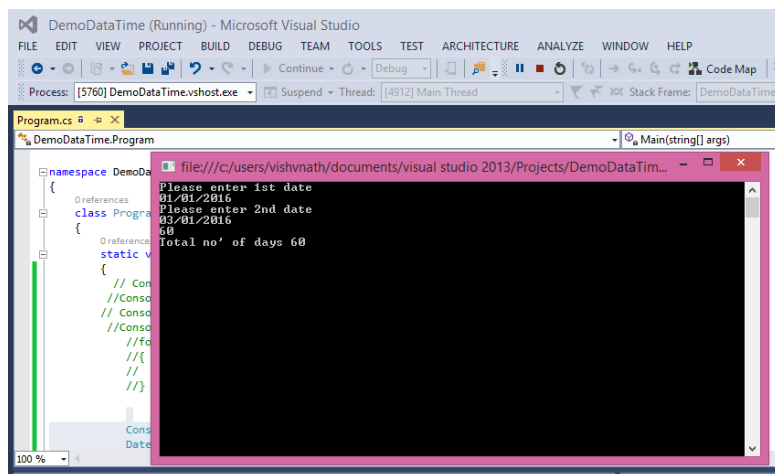
To print with difference b/w two dates

Time span: It is a structure. [days 365.25 (a year)]

```

Console.WriteLine("Please enter 1st date");
DateTime dd = DateTime.Parse(Console.ReadLine());
Console.WriteLine("Please enter 2nd date");
DateTime dd1 = DateTime.Parse(Console.ReadLine());
TimeSpan tt = dd1.Subtract(dd);
Console.WriteLine(tt.Days.ToString());
Console.WriteLine("Total no' of days {0}", tt.Days.ToString());
Console.ReadLine();

```



02-Feb-2016

ID Generation

- ✓ **Requirement**
- ✓ **No duplicates/No data inherit/Repeated data.**

Ticks: Convert date time, second, mile second into numbers.

24588 → 9-10 days.

☉ `DateTime.Now.Ticks.ToString()`
Advantage: No repetition.duplicates. i.e. 22022016
Note: in ticks only last 4 digits will change.

☉ `DateTime.Now.ToString("dd MMMM yyyy hh mm ss");`

i.e. 2202016031515
Note: only last *ss* digits will change.

☉ `Random aa=new Random().Next(9999.99999);`

Note: Here repetition increase (using random)

To stop repetition via random.

☉ `Random aa=new Random().Next(9999.99999)+ DateTime.Now.Ticks.ToString();`

Note: Here it will add or concatenate the ticks number then it gives the addition or unique number but repetition chances will be decreased(99%).

GUID: Global Unique Identifier

A GUID is a 128-bit integer (16-bytes) that you can use across all computers and networks wherever a unique identifier is required.

System.Guid class represents a GUID in .Net framework.

We can generate GUID in sqlserver with the help of NEWIDU function.

System.Runtime.InteropServices **namespace** : must be referred to utilize the GUID attribute.

Also note that GUID only contains alphanumeric characters and none of non-alphanumeric character will be seen in GUID except “-”.


Random : Pseudo-random numbers are chosen with equal probability from a finite set of numbers. The chosen numbers are not completely random because a mathematical algorithm is used to select them, but are sufficiently random number generator algorithm.

Random class has three public methods-Next, NextBytes & NextDouble.

Random is available when we inclusive the System namespace.

In this program, we use the random class and its Net Method, with two arguments.

Next(): is the inclusive minimum number allowed by the random number generator.

```
 Guid.NewGuid().ToString();
```

Date time, hour, second, mili second, with the collection of character & string. It generate character, special chararter(i.e. -,/,etc.).

Ex. 123AX-9XYZ-XYZAB

[Assembly info .CS file has—> GUID ID]

Controls: are classes likebutton etc. in V. Studio **[drag drop feature is object]**.

Objects: holds the string type data

[Text type data hold string]

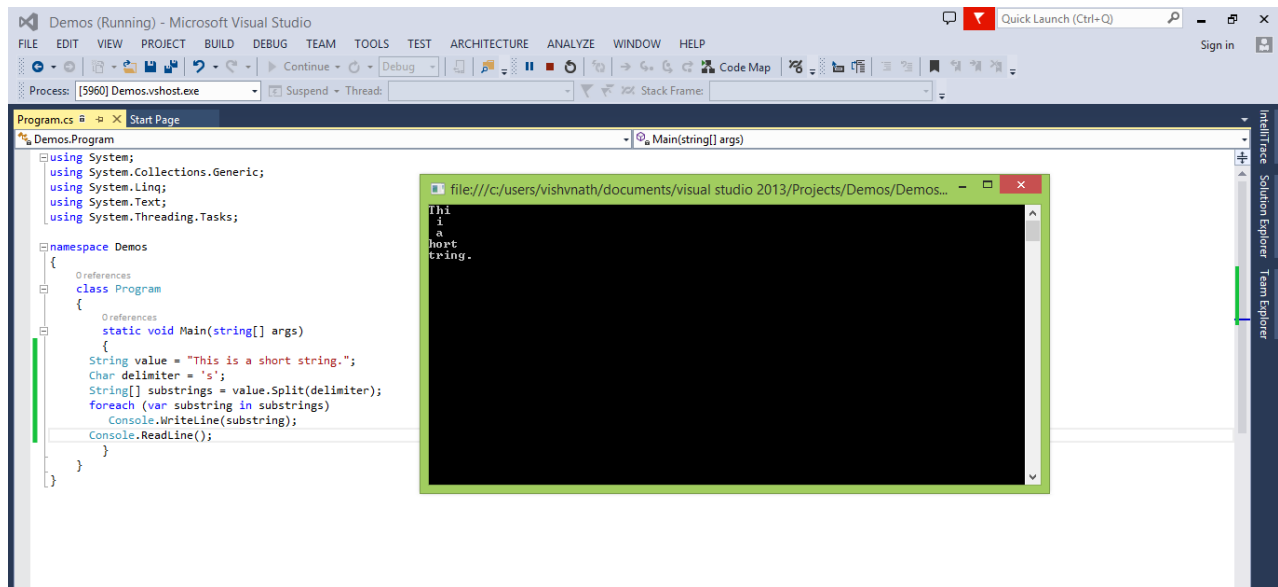
03-Feb-2016

Split Method: Common character divides the string is called split method.

In other word: Splits a string into substrings that are based on the characters in an array.
It will take 02 columns:- Data id and set.

Namespace: [System](#)

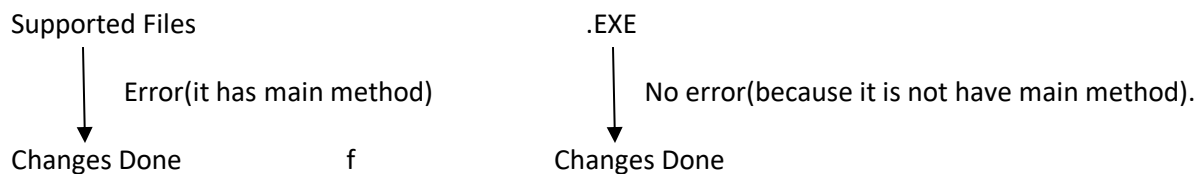
Assembly: mscorlib (in mscorlib.dll)



Seed Method: It is a pre-build property; generate system details.

EXE file: Created if it has main method (it take/use support files).

Dll → don't have exe (It only have all supported files).



Enumeration

We can change into numbers; it is user-defined; allocates the memory during compile time.

In other words: **Enumerations:** Enum or Enumeration, a distinct type that consists of a set of named constants called enumerator.

An enum can also be nested within a class or struct.

By default, the first enumerator has the value 0, and the value of each successive enumerator is increased by 1. Ex. Sat is 0, sun is 1, Mon is 2 and so forth.

Strack : Last in first out like values are stored on one another like a stack.

04-Feb-2016

Points to remember

- ✓ If we create library or services or web application then class will be public.
- ✓ Window & Console application(created) then access modifier will be internal.
- ✓ .Net don't have pointer → this keyword can be used on behalf of pointer.
- ✓ Wide type value.

```
Public void add(int a, int b)
{
    Public void add(int a, int b, ref int c)
    { ----- }
```

Ques : What id ref and out?

ref: will be used to remove the value; value will be initiated to overwrite.

The **ref** keyword causes an argument to be passed by reference, not by value. An argument that is passed to a **ref** parameter must be initialized before it is passed. This differs from **out** parameters, whose arguments do not have to be explicitly initialized before they are passed.

```
Ex: Public void add(int a, int b, ref int c)
    { ----- }
```

out : Value will never be initialized.

Members of a class can't have signatures that differ only by **ref** and **out**. A compiler error occurs if the only difference between two members of a type is that one of them has a **ref** parameter and the other has an **out** parameter.

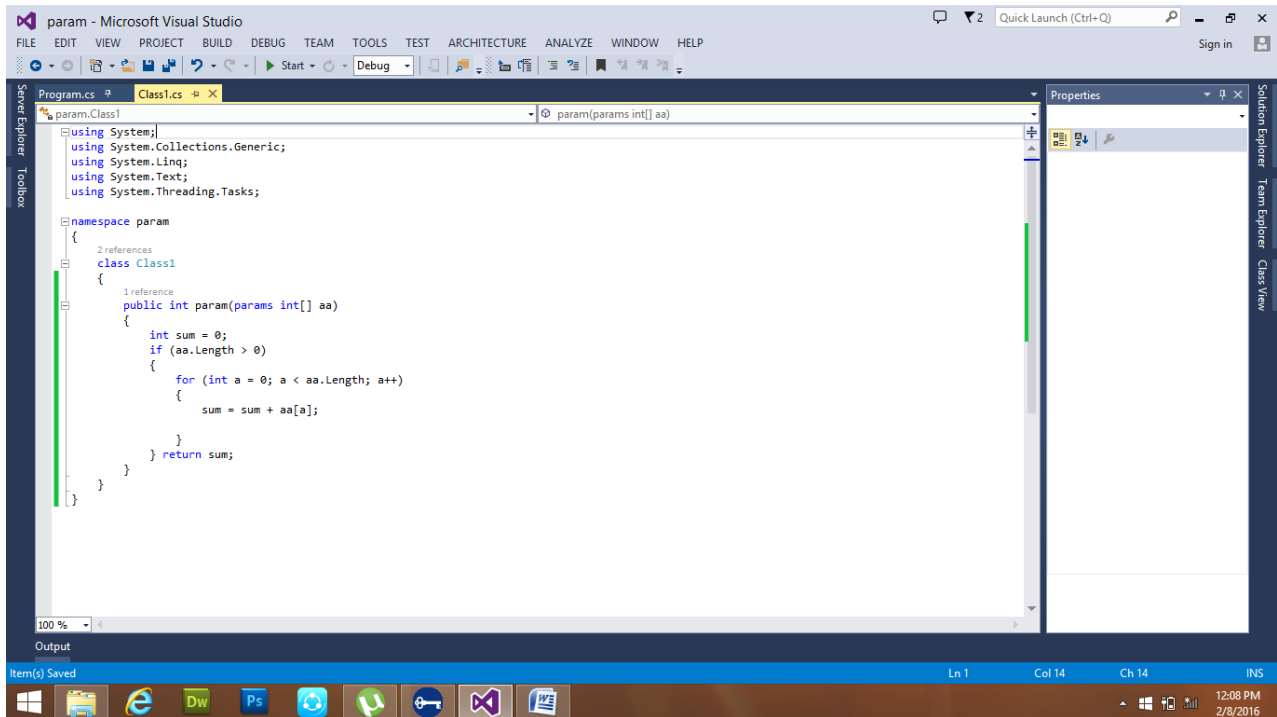
You can't use the **ref** and **out** keywords for the following kinds of methods:

- Async methods, which you define by using the async modifier.
- Iterator methods, which include a yield return or **yield break** statement.

```
Ex : int d=0;           // out
    int c;              // ref
    int d=aa.add(55,04, ref d)
    int c=bb.add(01,02, out c)
```

Using Param

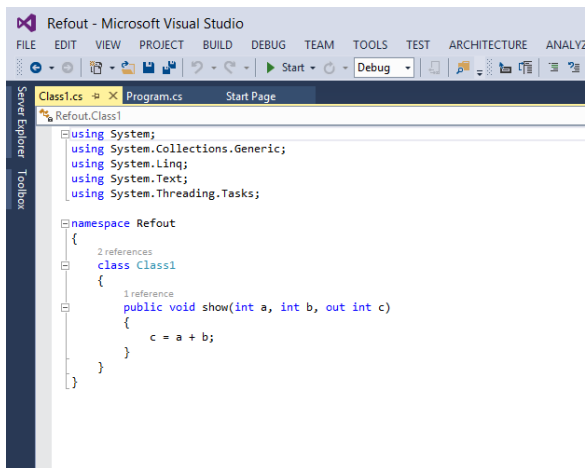
Class file : Class1



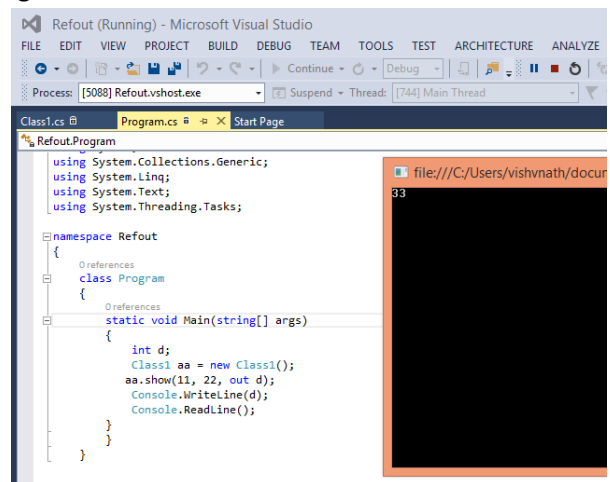
Program: Param

Using Out

Class file : Class1

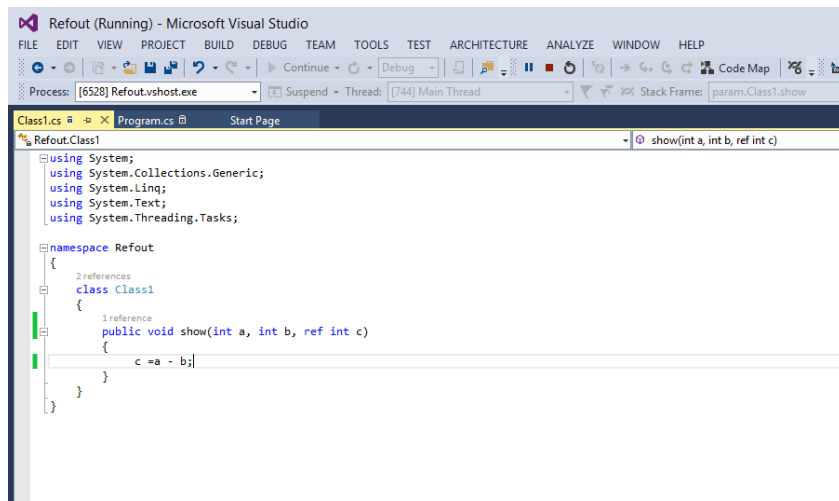


Program: Refout

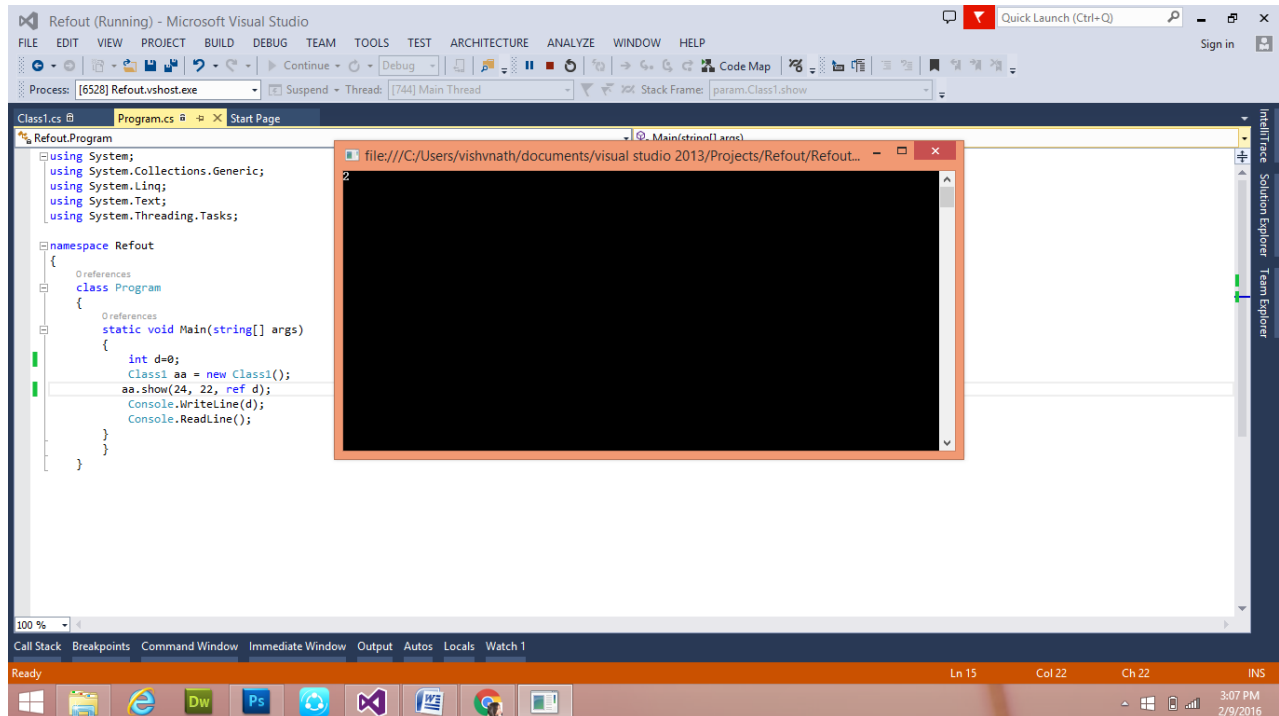


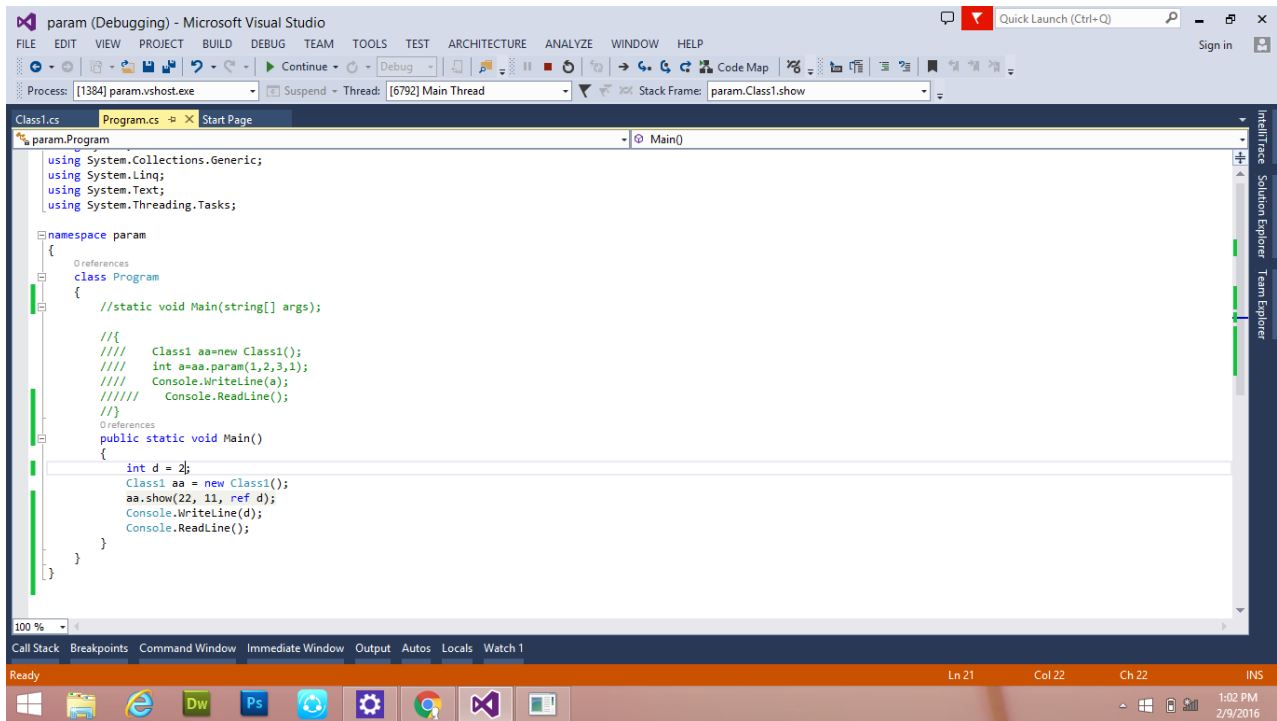
Using Ref

Class file: Class1

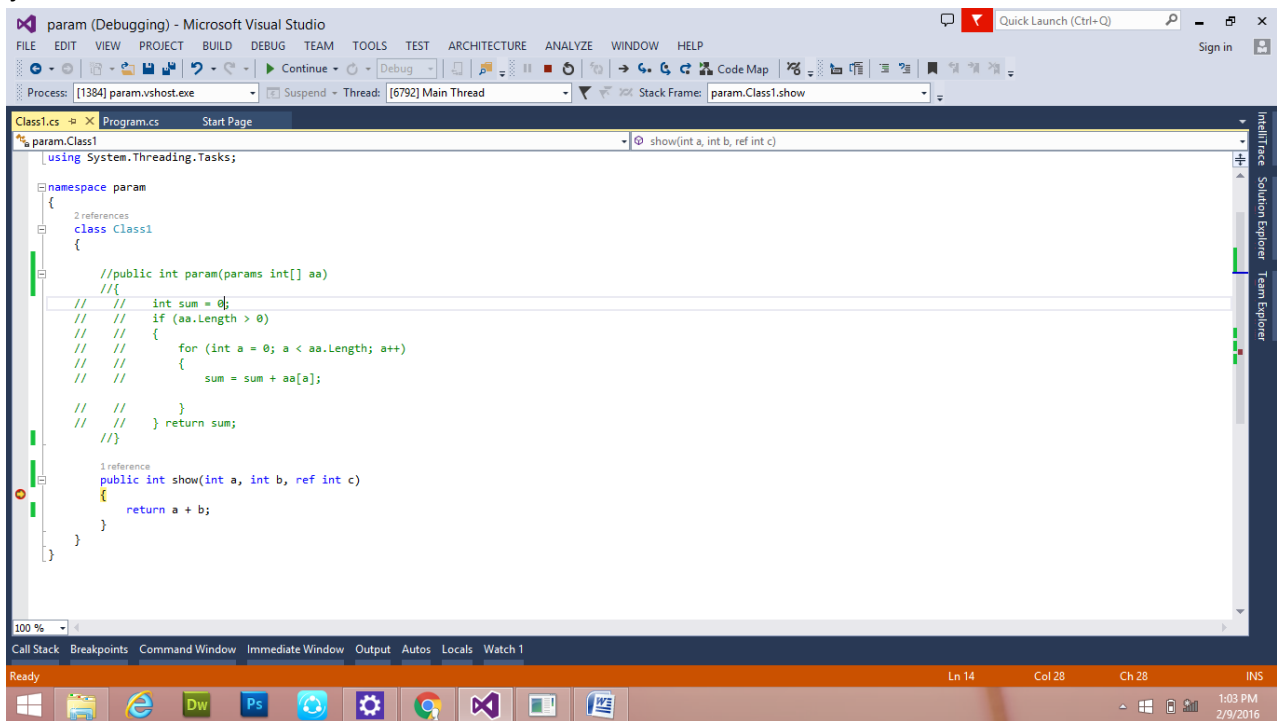


Program : Refout





Jj



Date: **09-Feb-2016**

What is System and Structure?

System: A set of things working together as part of a mechanism or an interconnecting network. In other words: a set of principles or procedures according to which something is done; an organized scheme or method.

Structure: Perform with rules and condition. Create multiple structures & use implicitly to club all methods. In other methods: the arrangement of and relations between the parts or elements of something complex.

OOPs concept consist

▶ Modular Programming

▶ Re-usability

Modular Programming: Structured programming (sometimes known as modular programming) is a subset of procedural programming that enforces a logical structure on the program being written to make it more efficient and easier to understand and modify. Certain languages such as Ada, Pascal, and dBASE are designed with features that encourage or enforce a logical program structure.

Re-usability Programming: Re-use libraries or other methods.

Oops divided into 03 parts

◆ ***Pura OOPs***

◆ ***OOPs***

◆ ***Object based***

Pure OOPs : Who have all instances, create object, multiple inheritance.

OOPs: Working on real world entity.

Object Based : Who doesn't have particular instance & unable to implement. It doesn't follow the concepts of pure & oops.

Note: Reference variable can inherit the value of variable.

Public int a=1

Class1 aa; //aa.a :- /inherit the space & gets permission for execution.

Class ab=new Class1();

Constructor : Special types of method which has same name of the classes.

A value type data member require a memory allocation. Data member require a memory through class-during allocation(non-static variable).

Class Class1

```
{ PublicT int a; //Non-static: it depends on class1 : it tells that which non-static member require memory
```

```
Public static int b; //Static:
```

Note: Non-static memory only has memory allocation(during compile time) it will never done execution. So we need to execute and pass.

```
Private int c; // Here object can't be executable because it will not provide permission for execution.
```

✚ Constructor is used for initialize the memory.

✚ It will never be a return type. Because to return any value we require a datamember. Therefore, we take access modifier with name.

i.e. public int a; OR public Class1();

Here in between, there is no datatype like void etc.

11-Feb-2016

Types of Constructor

• Private • Default • Parameterized • Copy • Static

Private Constructor: is a special instance constructor. It is generally used in classes that contain static members only. If a class has one or more private constructors and no public constructors, other classes (except nested classes) cannot create instances of this class.

The declaration of the empty constructor prevents the automatic generation of a default constructor. Note that if you still be private by default.

Private constructor are used to prevent creating instances of a class when there are no instance fields or methods.

Default: Have no parameters; they follow slightly different rules. Default constructors are one of the special member functions; if no constructor are declared in a class, the compiler default constructors are declared, the compiler does not provide a default constructor.

Parameterized: When an object is declared in a parameterized constructor, the initial values have to be passed as arguments to the constructor function. The normal way of object declaration may not work.

Copy: Defines the actions performed by the compiler when copying class objects. A copy constructor has one form parameter that is the type of the class (the parameter may be a reference to an object). It is used to create a copy of an existing object of the same class.

Static: There can be only one static constructor in the class. The *static constructor should be without parameters or parameter less*. It can only access the static members of the class. There should be no *access modifier in static constructor*.

The call to the static method is made by the CLR and not by the object, so we do not need to have the access modifier to it.

Using Construtor

Default

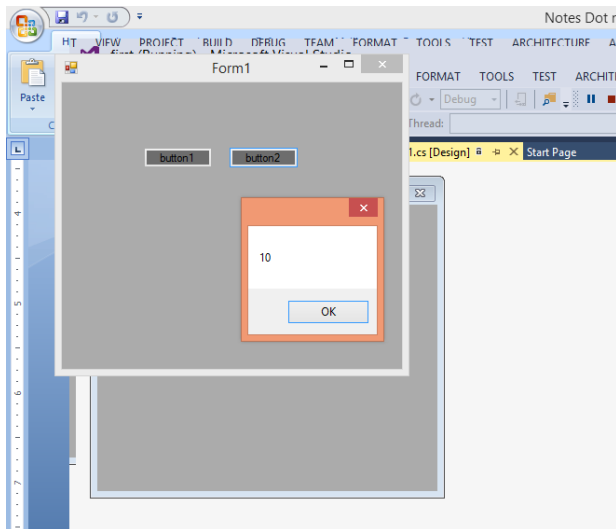
File name: first, Form 1 : Button 2

Namespace (common): `using System.Windows.Forms;`

```
private void button2_Click(object sender, EventArgs e)
{
    Class2 bb = new Class2();
    int b = bb.a;
    MessageBox.Show(b.ToString());
}
```

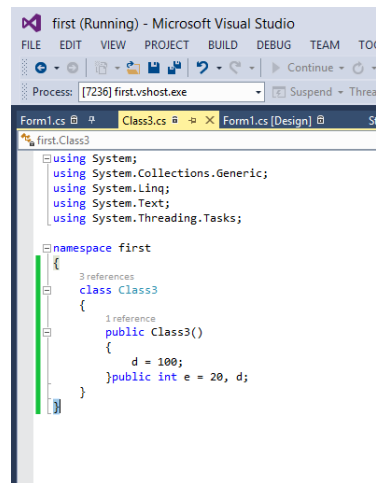
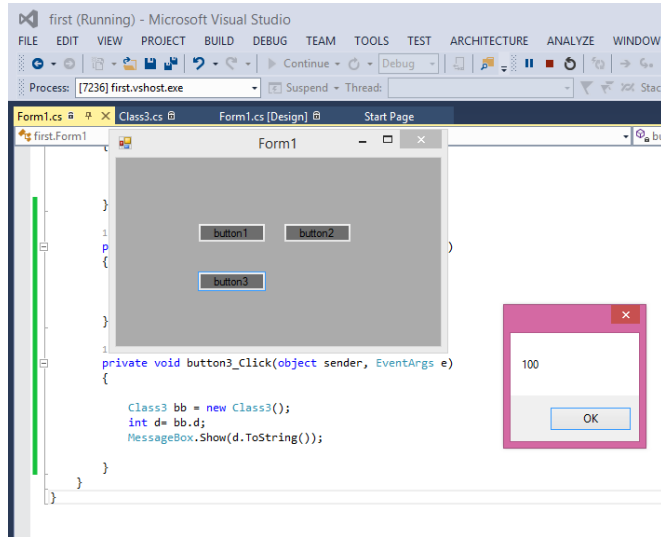
Class name : Class2

```
namespace first
{
    class Class2
    {
        public int a = 10;
    }
}
```



File name: first, Form 1: Button 3

Class: Class3



Parameterized

File name: first, Form 1: Button 3

Class: Class3

16-Feb-16

Abstraction: Hiding initial features.

Library(DLL)

- Here DLL is used for re-usability.
- DLL has no main method. So we can never execute directly.
- DLL not required supported files.

Note: Class library are having same name with namespace.

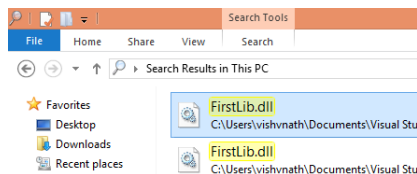
s.ExE → Can never be update (Because it is having supported files).

.DLL → Can be updated.

How to create Class Library

1. New Project → Class Library → OK
2. Create program in Class Library and build application.

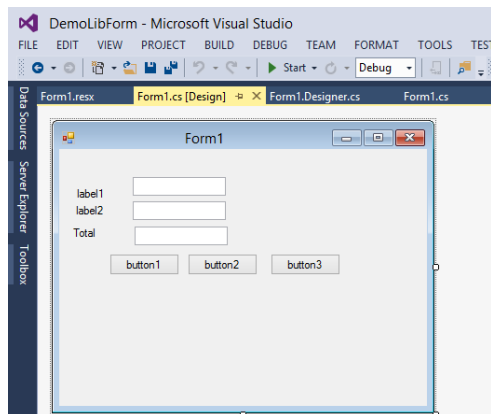
Note: While creating new Library we need to put in our knowledge that we (developer can re-use its code) but other person can't edit so we need to give.



Also class should be public & it is by default **Public**.

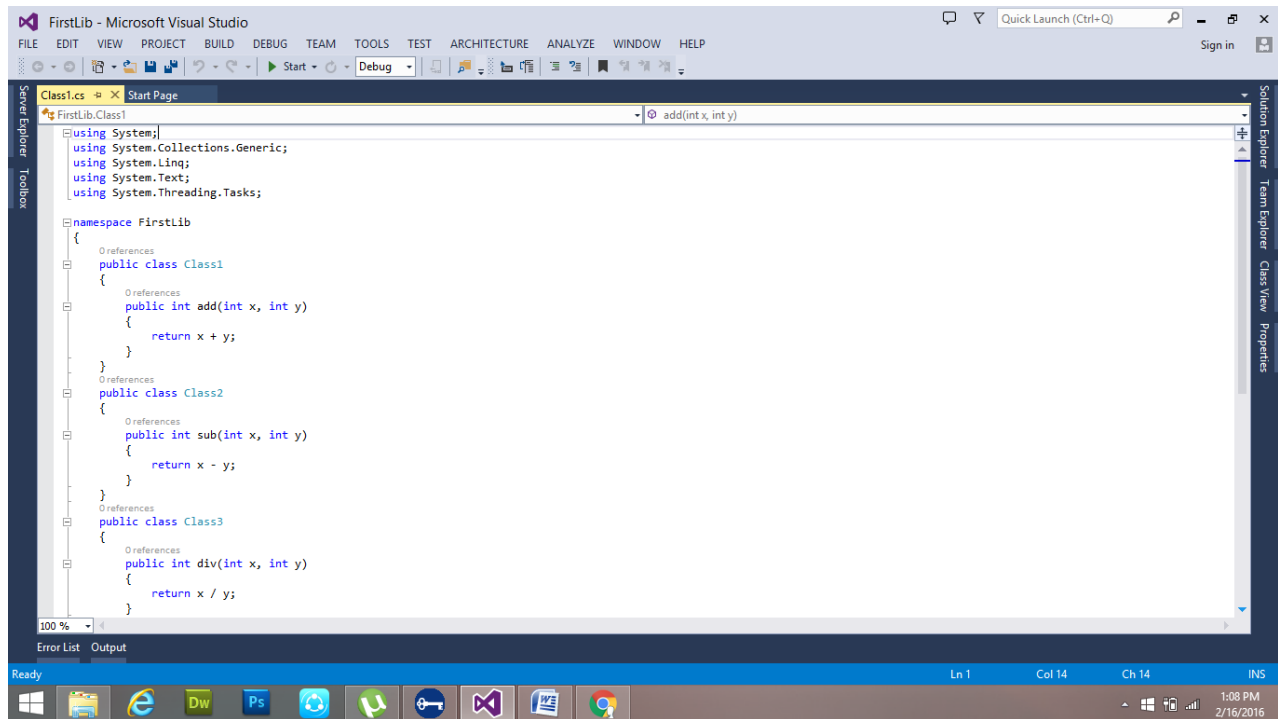
Ques. Create a form with add, s sub & div buttons. Also use DLL in it.

Create Form



Butn1: Add, Butn2:Sub, Butn3: Div

New Project → Class Library → (FirstLib)Name → OK

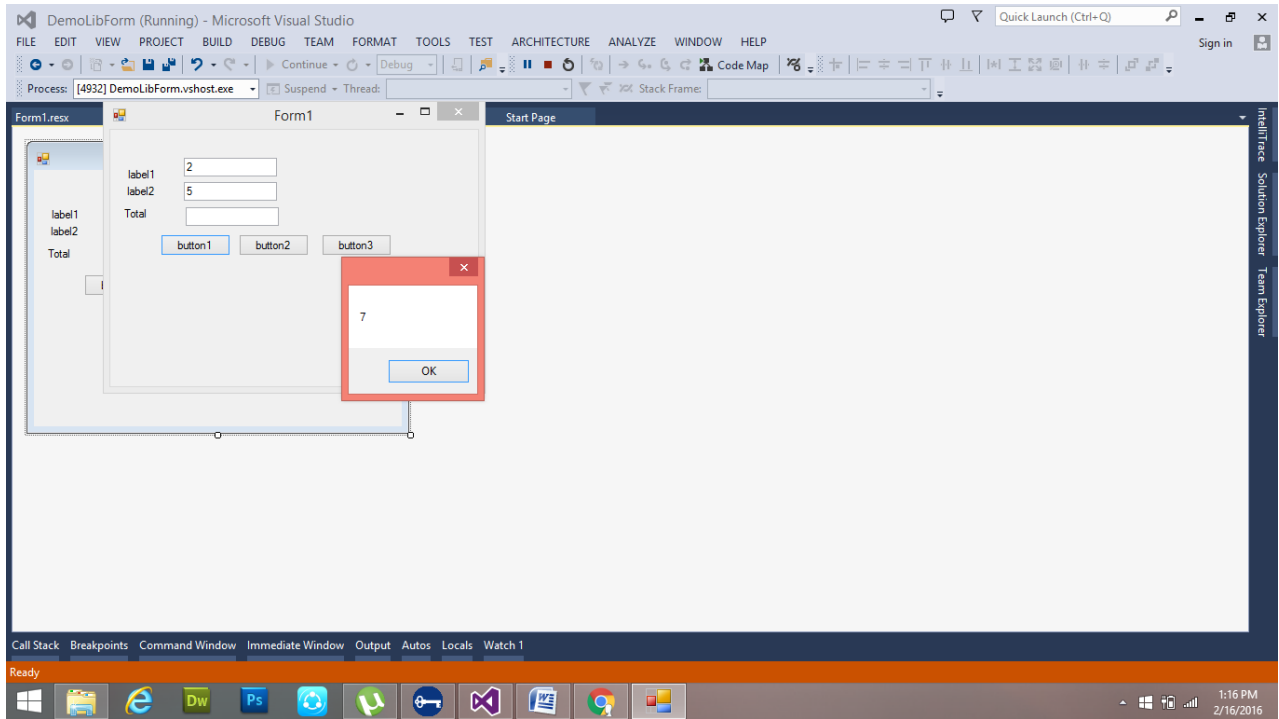
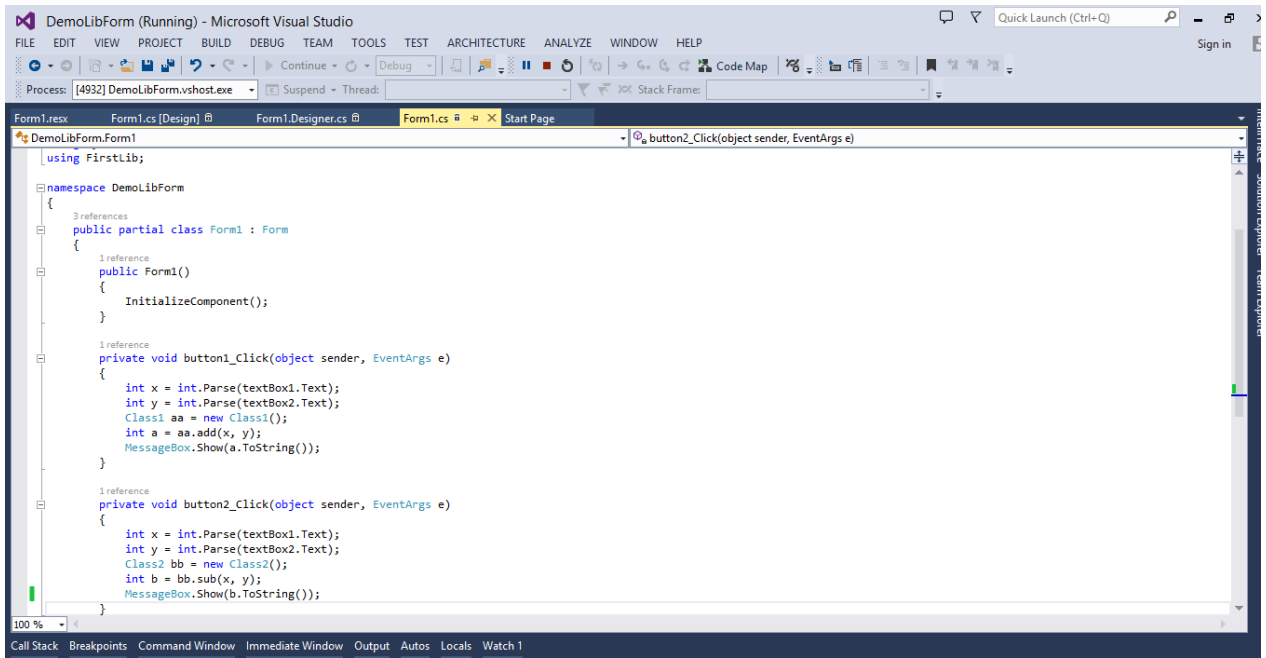


Use DLL

- New Project → Select Window Application Form & name(DemoLibForm) → OK
- Right Click on Solution Explorer → add reference in reference folder.
- Select DLL from the location of DLL then OK

Namespace: `using FirstLib;`

```
private void button1_Click(object sender, EventArgs e)
{
    int x = int.Parse(textBox1.Text);
    int y = int.Parse(textBox2.Text);
    Class1 aa = new Class1();
    int a = aa.add(x, y);
    MessageBox.Show(a.ToString());
}
```



17-FEB-2016

Polymorphism: Polymorphism is an object-oriented programming concept that refers to the ability of a variable, function or object to take on multiple forms. A language that features polymorphism allows developers to program in the general rather than program in the specific.

In other words, Single item work as multiple type.

Types of Polymorphism:



Overloading : Same class, same method name but different parameters.

Overriding: Same method name, same parameter but different classes.

Virtual Keyword: in base class method.

When a derived class inherits from a base class, it gains all the methods, fields, properties and events of the base class. The designer of the derived class can choose whether to

- override virtual members in the base class,
- inherit the closest base class method without overriding it
- define new non-virtual implementation of those members that hide the base class implementations

A derived class can override a base class member only if the base class member is declared as virtual or abstract. The derived member must use the override keyword to explicitly indicate that the method is intended to participate in virtual invocation.

Override Keyword: in drive class method.

Ques. To print simply

Class: Class1

Button : Btn1

```

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

namespace DemoPoly
{
    2 references
    class Class1
    {
        1 reference
        public int add(int a, int b)
        {
            return a+b;
        }
        0 references
        public int add(int a)
        {
            return a;
        }
    }
    0 references
    class Class2
    {
        0 references
        public int add()
        {
            return 10;
        }
    }
}

```

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace DemoPoly
{
    3 references
    public partial class Form1 : Form
    {
        1 reference
        public Form1()
        {
            InitializeComponent();
        }
        1 reference
        private void button1_Click(object sender, EventArgs e)
        {
            Class1 aa = new Class1();
            int a=aa.add(11,22);
            MessageBox.Show(a.ToString());
        }
    }
}

```

Return Single value

```

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

namespace DemoPoly
{
    4 references
    class Class1
    {
        1 reference
        public int add(int a, int b)
        {
            return a+b;
        }
        1 reference
        public int add(int a)
        {
            return a;
        }
    }
    0 references
    class Class2
    {
        0 references
        public int add()
        {
            return 10;
        }
    }
}

```

```

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


namespace DemoPoly
{
    4 references
    class Class1
    {
        1 reference
        public int add(int a, int b)
        {
            return a+b;
        }
        1 reference
        public int add(int a)
        {
            return a;
        }
    }
    0 references
    class Class2
    {
        0 references
        public int add()
        {
            return 10;
        }
    }
}

```

Inheritance

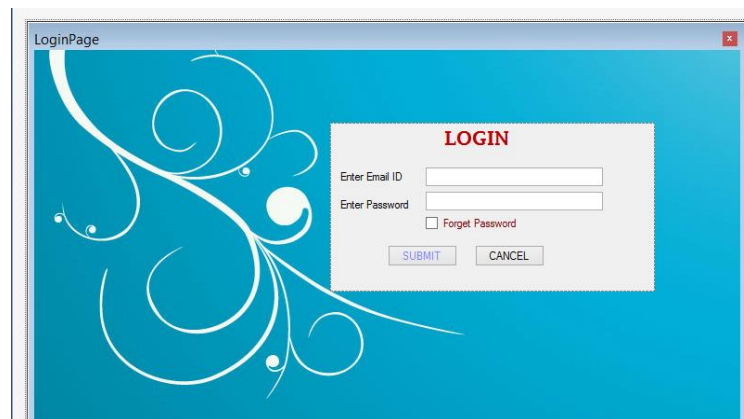
Transaction System

Product Tracker



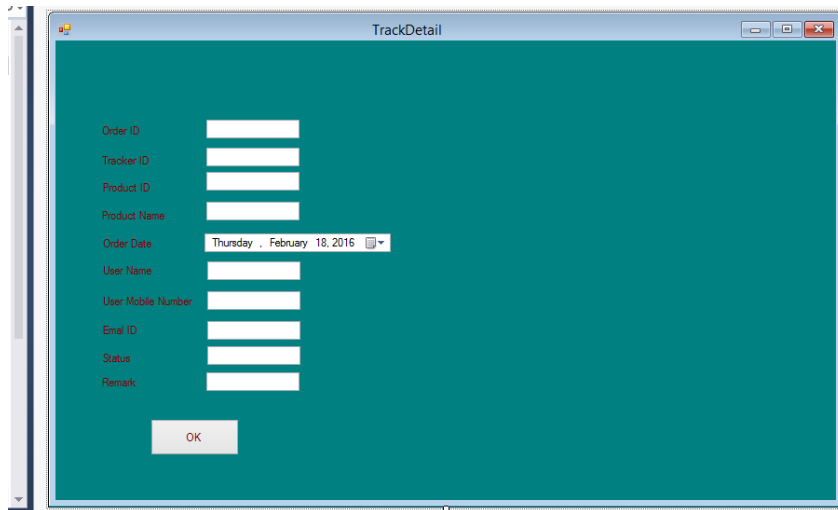
A screenshot of a software window titled "Form1". The window has a blue background with a white decorative swirl on the left. In the center, there is a text label "Track the details of your product" above a white input field. To the right of the input field is a button labeled "SEARCH".

Login Page



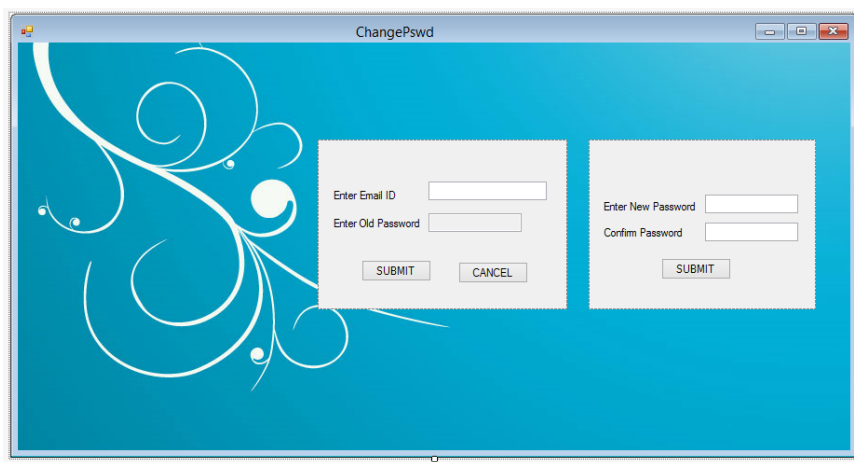
A screenshot of a software window titled "LoginPage". The window has a blue background with a white decorative swirl on the left. In the center, there is a white box with a red title "LOGIN". Inside the box, there are two input fields: "Enter Email ID" and "Enter Password". Below the "Enter Password" field is a checkbox labeled "Forget Password". At the bottom of the box are two buttons: "SUBMIT" and "CANCEL".

Tracker Registration



A screenshot of a web application window titled "TrackDetail". The window has a teal background. On the left side, there is a list of labels for registration fields: Order ID, Tracker ID, Product ID, Product Name, Order Date, User Name, User Mobile Number, Email ID, Status, and Remark. To the right of these labels are corresponding input fields. The "Order Date" field is a date picker showing "Thursday, February 18, 2016". At the bottom left of the form area, there is a button labeled "OK".

Change Password



A screenshot of a web application window titled "ChangePswd". The window has a blue background with a white decorative swirl on the left. The form is divided into two main sections. The left section contains two input fields: "Enter Email ID" and "Enter Old Password", followed by "SUBMIT" and "CANCEL" buttons. The right section contains two input fields: "Enter New Password" and "Confirm Password", followed by a "SUBMIT" button.

Forget Password



A screenshot of a web application window titled "ForgetPswd". The window has a blue background with a white decorative swirl on the left. The form contains a single input field with the placeholder text "Please enter your Email ID" in red. To the right of the input field is a button labeled "SUBMIT".