**A**

**LAB REPORT**

**ON**

**Dotnet Technology**

**By**

**Prajwal Dahal**



**Submitted to:**

**Saurab Adhikari**

**Lecturer**

**Kantipur College of Management and Information Technology**

In partial fulfillment of the requirements for the Course

Dot net Technology

Mid Baneshwor, Kathmandu

November 2022

TABLE OF CONTENTS

[1 Write a program to print factorial of a number. 1](#_Toc121864252)

[1.1 Source Code 1](#_Toc121864253)

[1.2 Output Window 1](#_Toc121864254)

[2 Write a program to create an array by taking input from user and displaying odd and even number from that array. 2](#_Toc121864255)

[2.1 Source Code 2](#_Toc121864256)

[2.2 Output Window 3](#_Toc121864257)

[3 Write a program to show constructor and destructor in C#. 4](#_Toc121864258)

[3.1 Source Code 4](#_Toc121864259)

[3.2 Output Window 5](#_Toc121864260)

[4 Write a program to demonstrate data encapsulation in c#. 6](#_Toc121864261)

[4.1 Source Code 6](#_Toc121864262)

[4.2 Output Window 7](#_Toc121864263)

[5 Write a program to demonstrate inheritance in C#. 8](#_Toc121864264)

[5.1 Source Code 8](#_Toc121864265)

[5.2 Output Windows 8](#_Toc121864266)

[6 Write a program to return multiple value from method using OUT keyword. 9](#_Toc121864267)

[6.1 Source Code 9](#_Toc121864268)

[6.2 Output Window 9](#_Toc121864269)

[7 Write a program to swap two variable using method ref keyword. 10](#_Toc121864270)

[7.1 Source Code 10](#_Toc121864271)

[7.2 Output Window 10](#_Toc121864272)

[8 Write a program to take variable number of parameter in method and calculate its sum. 11](#_Toc121864273)

[8.1 Source Code 11](#_Toc121864274)

[8.2 Output Window 11](#_Toc121864275)

[9 Write a Program to demonstrate method overloading in C#. 12](#_Toc121864276)

[9.1 Source Code 12](#_Toc121864277)

[9.2 Output Window 12](#_Toc121864278)

[10 Write a program to demonstrate operator overloading in C#. 13](#_Toc121864279)

[10.1 Source Code 13](#_Toc121864280)

[10.2 Output Window 14](#_Toc121864281)

[11 Write a Program to demonstrate use of override, virtual and abstract keyword. 15](#_Toc121864282)

[11.1 Source Code 15](#_Toc121864283)

[11.2 Output Window 16](#_Toc121864284)

[12 Write a program to demonstrate interface in c#. 17](#_Toc121864285)

[12.1 Source Code 17](#_Toc121864286)

[12.2 Output Window 17](#_Toc121864287)

[13 Write a program to demonstrate enumeration in C#. 18](#_Toc121864288)

[13.1 Source Code 18](#_Toc121864289)

[13.2 Output Window 18](#_Toc121864290)

[14 Write a Program to show difference between struct and class. 19](#_Toc121864291)

[14.1 Source Code 19](#_Toc121864292)

[14.2 Output Window 21](#_Toc121864293)

[15 Write a Program to demonstrate use of generics. 22](#_Toc121864294)

[15.1 Source Code 22](#_Toc121864295)

[15.2 Output Windows 22](#_Toc121864296)

[16 Write a Program to demonstrate dynamic binding. 23](#_Toc121864297)

[16.1 Source Code 23](#_Toc121864298)

[16.2 Output Window 23](#_Toc121864299)

[17 Write a program to return total length of two string using Lambda Expression 24](#_Toc121864300)

[17.1 Source code 24](#_Toc121864301)

[17.2 Output Window 24](#_Toc121864302)

[18 Write a Program to create an array to store number and display number greater than 3 in ascending order using LINQ. 25](#_Toc121864303)

[18.1 Source Code 25](#_Toc121864304)

[18.2 Output Window 25](#_Toc121864305)

[19 Write a Program to declare delegate and show multicast delegates. 26](#_Toc121864306)

[19.1 Source Code 26](#_Toc121864307)

[19.2 Output Windows 27](#_Toc121864308)

[20 Write a Program to show database connection. 28](#_Toc121864309)

[20.1 Source Code 28](#_Toc121864310)

[20.2 Output Window 29](#_Toc121864311)

[21 Write a program to insert a record on the table and retrieve it. 30](#_Toc121864312)

[21.1 Source Code 30](#_Toc121864313)

[21.2 Output Window 31](#_Toc121864314)

# Write a program to print factorial of a number.

## Source Code

using System;

class Factorial{

public static int CalcFactorial(int n){

if(n==0 || n==1)

return 1;

else

return n\*CalcFactorial(n-1);

}

public static void Main(){

Console.Write("enter a number: ");

int a=Convert.ToInt32(Console.ReadLine());

Console.WriteLine("the factorial of {0} is {1}",a,CalcFactorial(a));

}

}

## Output Window

# Write a program to create an array by taking input from user and displaying odd and even number from that array.

## Source Code

using System;

class OddEven{

public static void Main(){

int[]a=new int[5];

int counte=0;

int counto=0;

for(int i=0;i<5;i++){

Console.Write("enter a number: ");

a[i]=Convert.ToInt32(Console.ReadLine());

}

int[]odd=new int[5];

int[]even=new int[5];

for(int i=0;i<5;i++){

if(a[i]%2==0){

even[counte]=a[i];

counte++;

}

else{

odd[counto]=a[i];

counto++;

}

}

Console.WriteLine("even numbers are: ");

for(int i=0;i<counte;i++){

Console.Write(" {0}",even[i]);

}

Console.WriteLine();

Console.WriteLine("odd numbers are: ");

for(int i=0;i<counto;i++){

Console.Write(" {0}",odd[i]);

}

}

}

## Output Window

# Write a program to show constructor and destructor in C#.

## Source Code

using System;

class Member

{

public Member()

{

Console.WriteLine("Default Constructor was called.");

}

public Member(string name)

{

Console.WriteLine("Parameterized Constructor was called.");

}

~Member()

{

Console.WriteLine("Destructor was called.");

}

public static void Main(){

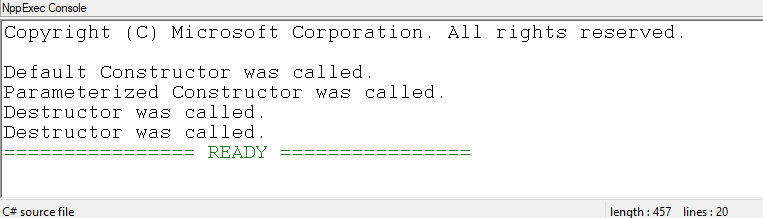
Member m = new Member();

Member m1 = new Member("Hello");

}

}

## Output Window



# Write a program to demonstrate data encapsulation in c#.

## Source Code

using System;

class Sum{

private int a;

private int b;

public int A{

set{

A=value;

}

get{

return A;

}

}

public int B{

get;

set;

}

public void Add(){

Console.WriteLine("sum is {0}",a+b);

}

}

class Demo{

public static void Main(){

Sum s = new Sum();

s.a=12;

s.b=13;

s.Add();

}

}

## Output Window

# Write a program to demonstrate inheritance in C#.

## Source Code

using System;

class X{

public X(){

Console.WriteLine("hello from class X");

}

public void Sum(int a,int b){

Console.WriteLine(a+b);

}

}

class W:X{

public W(){

Console.WriteLine("hello from class W");

}

public static void Main(){

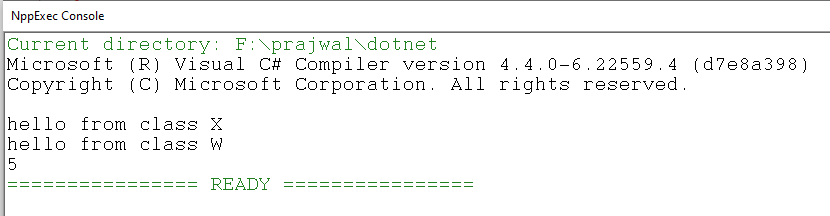
W obj1=new W();

obj1.Sum(2,3);

}

}

## Output Windows



# Write a program to return multiple value from method using OUT keyword.

## Source Code

using System;

class ABC{

public static void ChangeValue(out int sum, out int product){

int x=2;

int y=3;

sum=0;

product=0;

sum=x+y;

product=x\*y;

}

public static void Main(){

int sum,product;

ABC.ChangeValue(out sum,out product);

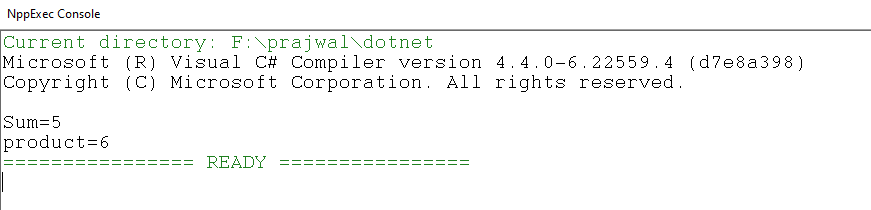
Console.WriteLine("Sum={0}",sum);

Console.WriteLine("product={0}",product);

}

}

## Output Window



# Write a program to swap two variable using method ref keyword.

## Source Code

using System;

class ABC{

public static void swap(ref int x,ref int y){

int temp;

temp=x;

x=y;

y=temp;

}

public static void Main(){

int x=2,y=3;

Console.WriteLine("before Swapping:\nx={0} y={1}",x,y);

ABC.swap(ref x,ref y);

Console.WriteLine("After Swapping:\nx={0} y={1}",x,y);

}

}

## Output Window

# Write a program to take variable number of parameter in method and calculate its sum.

## Source Code

using System;

public class Demo{

public static void Sum(params int[] num){

int sum =0;

for(int i=0;i<num.Length;i++){

sum+=num[i];

}

Console.WriteLine(sum);

}

public static void Main(){

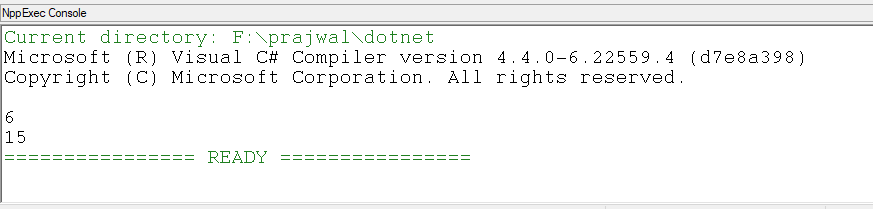
Demo.Sum(1,2,3);

Demo.Sum(1,2,3,4,5);

}

}

## Output Window



# Write a Program to demonstrate method overloading in C#.

## Source Code

using System;

class Sum{

public void add1(int a,int b){

Console.WriteLine(a+b);

}

public void add1(int a,int b,int c){

Console.WriteLine(a+b+c);

}

public static void Main(){

Sum s = new Sum();

s.add1(1,2);

s.add1(1,2,3);

}

}

## Output Window

# Write a program to demonstrate operator overloading in C#.

## Source Code

using System;

class Car{

public int speed;

public Car( bool x){

if(x){

Console.Write("enter speed of car:");

speed=Convert.ToInt32(Console.ReadLine());

}

}

public static Car operator+ (Car c, Car c2){

Car c3 = new Car(false);

c3.speed=c.speed+c2.speed;

return c3;

}

public static void Main(){

Car c = new Car(true);

Car c2 = new Car(true);

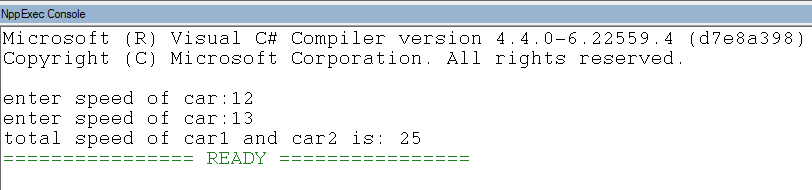
Car c3=c+c2;

Console.WriteLine("total speed of car1 and car2 is: "+c3.speed);

}

}

## Output Window



# Write a Program to demonstrate use of override, virtual and abstract keyword.

## Source Code

using System;

abstract class Sum{

public abstract void add1(int a,int b);

public virtual void mul(int a,int b){

Console.WriteLine(a+b);

}

}

class A:Sum{

public override void add1(int a,int b)

{

Console.WriteLine(a+b);

}

public override void mul(int a,int b)

{

Console.WriteLine((a/(float)b).ToString("0.00"));

}

public static void Main()

{

Sum s = new A();

s.add1(1,2);

s.mul(10,3);

}

}

## Output Window

# Write a program to demonstrate interface in c#.

## Source Code

using System;

public interface Shape{

void Area();

}

class Circle:Shape{

public void Area(){

int radius=8;

Console.WriteLine(22/7\*radius\*radius);

}

public static void Main(){

Circle c = new Circle();

c.Area();

}

}

## Output Window

# Write a program to demonstrate enumeration in C#.

## Source Code

using System;

class EnumDemo{

enum level{

expert,

hard,

medium,

easy

}

public static void Main(){

level l = level.expert;

Console.WriteLine(l);

level z=(level)3;

Console.WriteLine(z);

int y=(int)level.hard;

Console.WriteLine(y);

}

}

## Output Window

# Write a Program to show difference between struct and class.

## Source Code

using System;

struct coordinate{

public int x;

public int y;

public coordinate(int x,int y){

this.x=x;

this.y=y;

}

public void Show(){

Console.WriteLine("x={0} y={1}",x,y);

}

}

class A{

public int x;

public int y;

public A(int x,int y){

this.x=x;

this.y=y;

}

public void Show(){

Console.WriteLine("x={0} y={1}",x,y);

}

}

class Demo{

public static void Main(){

coordinate c = new coordinate(10,12);

Console.WriteLine("when struct object is created");

c.Show();

ModifyStructVal(c);

Console.WriteLine("After ModifyStructVal method is called");

c.Show();

Console.WriteLine(" ");

A obj = new A(14,15);

Console.WriteLine("when class object is created");

obj.Show();

ModifyClassVal(obj);

Console.WriteLine("After ModifyClassVal method is called");

obj.Show();

}

public static void ModifyStructVal(coordinate c){

c.x+=100;

c.y+=100;

Console.WriteLine("inside ModifyStructVal x={0} y={1}",c.x,c.y);

}

public static void ModifyClassVal(A a){

a.x+=100;

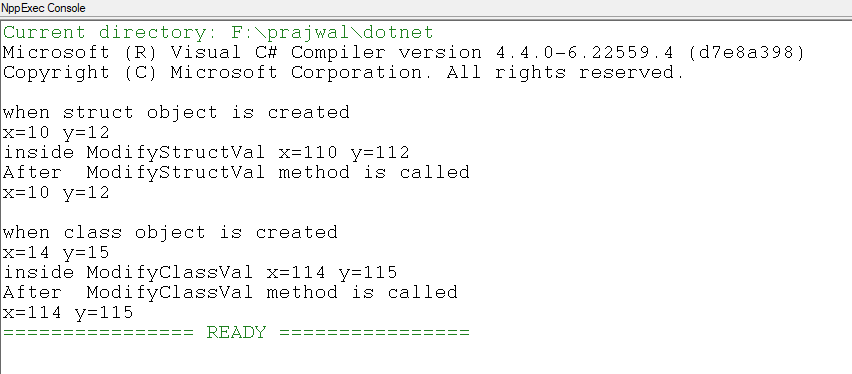
a.y+=100;

Console.WriteLine("inside ModifyClassVal x={0} y={1}",a.x,a.y);

}

}

## Output Window



# Write a Program to demonstrate use of generics.

## Source Code

using System;

public class Stack<T>{

int index=0;

T []data= new T[5];

public void push(T val){

data[index]=val;

index++;

}

public T pop(){

return data[--index];

}

}

class Demo{

public static void Main(){

Stack<int> s= new Stack<int>();

s.push(10);

s.push(20);

Console.WriteLine(s.pop());

}

}

## Output Windows

# Write a Program to demonstrate dynamic binding.

## Source Code

using System;

class DynamicBinding{

public static int Sum(int a,int b){

return a+b;

}

public static double Sum(double a,double b){

return a+b;

}

public static void Main(){

dynamic a=Sum(3,4);

dynamic b =Sum(3.9,4.9);

Console.WriteLine(a.GetType());

Console.WriteLine(a);

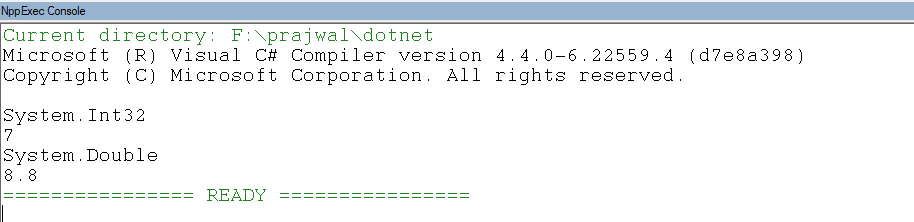
Console.WriteLine(b.GetType());

Console.WriteLine(b);

}

}

## Output Window



# Write a program to return total length of two string using Lambda Expression

## Source code

using System;

using static System.Console;

class LambdaExoressionDemo{

public static void Main(){

Func<string,string,int> TotalLength=(s1,s2)=>s1.Length+s2.Length;

WriteLine(TotalLength("hello","world"));

}

}

## Output Window

# Write a Program to create an array to store number and display number greater than 3 in ascending order using LINQ.

## Source Code

using System.Linq;

using System;

class LinqTest{

public static void Main(){

int []a={45,2,1,87,34};

var ResultQS=(from num in a

where num > 3

orderby num ascending

select num );

foreach(int n in ResultQS){

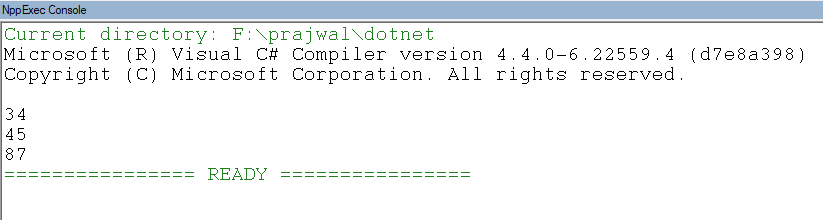
Console.WriteLine("{0}",n);

}

}

}

## Output Window



# Write a Program to declare delegate and show multicast delegates.

## Source Code

using System;

public delegate void MyDelegate(String msg);

public class ClassA{

public static void MethodA(String message){

Console.WriteLine("MethodA called with msg:"+message);

}

}

public class ClassB{

public static void MethodB(String message){

Console.WriteLine("MethodB called with msg: "+message);

}

}

class test{

public static void Main(){

MyDelegate del1 = new MyDelegate(ClassA.MethodA);

MyDelegate del2 = new MyDelegate(ClassB.MethodB);

MyDelegate del = del1+del2;

del("hello");

del=del2-del1;

del("hello");

}

}

## Output Windows

# Write a Program to show database connection.

## Source Code

using System;

using System.Data;

using System.Data.SqlClient;

class program

{

public static void Main()

{

SqlConnectionStringBuilder sb = new SqlConnectionStringBuilder();

sb.DataSource = @"(localdb)\MSSQLLocalDB";

sb.InitialCatalog="dotnet";

sb.IntegratedSecurity = true;

using (SqlConnection con = new SqlConnection(sb.ConnectionString))

{

con.Open();

string createDB = @"create table customer(id int primary key,name varchar(20))";

SqlCommand cmd = new SqlCommand(createDB, con);

try

{

cmd.ExecuteNonQuery();

Console.WriteLine("table created");

}

catch (Exception e)

{

Console.WriteLine(e);

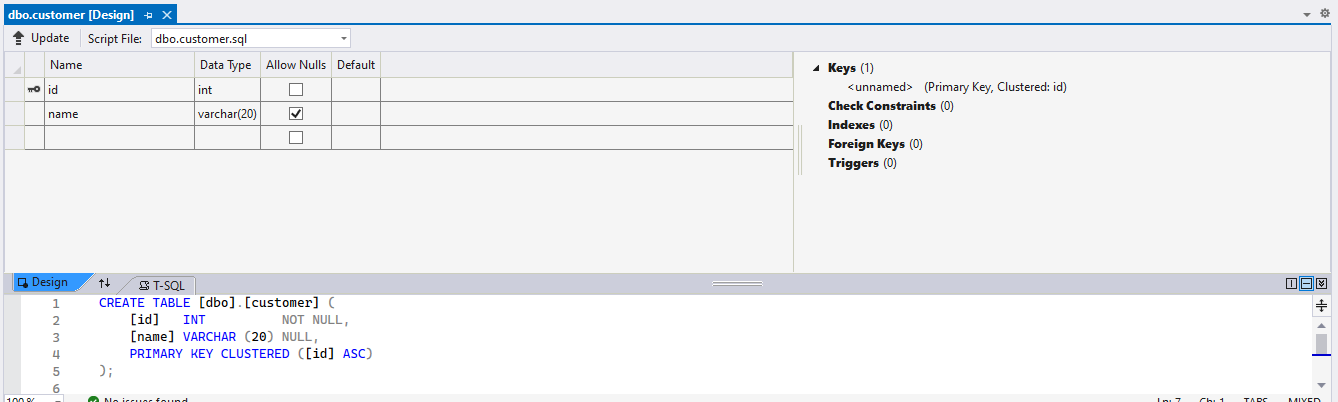
}

}

}

}

## Output Window



# Write a program to insert a record on the table and retrieve it.

## Source Code

using System;

using System.Data;

using System.Data.SqlClient;

class program

{

public static void Main()

{

SqlDataReader reader;

SqlConnectionStringBuilder sb = new SqlConnectionStringBuilder();

sb.DataSource = @"(localdb)\MSSQLLocalDB"; sb.InitialCatalog="dotnet";

sb.IntegratedSecurity = true;

using (SqlConnection con = new SqlConnection(sb.ConnectionString))

{

con.Open();

string insertDB = @"insert into customer values(3,'roshan kumar sunwar'),(4,'padam raj joshi')";

SqlCommand cmd = new SqlCommand(insertDB, con);

string retrieveDB=@"select \* from customer";

SqlCommand cmd1 = new SqlCommand(retrieveDB, con);

try

{

cmd.ExecuteNonQuery();

Console.WriteLine("record inserted");

Console.WriteLine("records: ");

reader=cmd1.ExecuteReader();

while(reader.Read())

{

int x=Convert.ToInt32(reader["id"]);

string name=reader["name"].ToString();

Console.Write("{0} {1}\n",x,name);

}

}

catch (Exception e)

{

Console.WriteLine(e);

}

}

}

}

## Output Window