**Practical 3 Qs. WAP to demonstrate boxing and unboxing.** using System;

using System.Collections.Generic; using System.Text;

namespace PracticalFile { class MySample

{

static void Main(string[] args) { Console.WriteLine("\t\tBoxing and Unboxing\n"); int num = 50;

object obj = num; //boxing int i = (int)obj; //Unboxing;

Console.WriteLine("Value of num is : "+num); Console.WriteLine("Value of ob object is : "+obj); Console.WriteLine("Value of i is : "+i); Console.ReadKey();

}

}

}

**Practical 4**

**Qs. WAP to demonstrate to demonstrate “out” and “params” methods parameter modifier followed by pass by reference using “ref” keyword.**

// Program using out keyword using System;

namespace PracticalFile

{

class MySample {

static void add(int x, int y, out int ans) { ans = x + y;

}

static void Main(String[] args)

{

int answer;

add(10, 20, out answer); Console.WriteLine(“10 + 20 = {0}”, answer); Console.ReadLine();

}

}

}

1. // Program using “params” keyword

using System; namespace PracticalFile

{

class MySample {

static double Average(params double[] values) { double sum = 0;

if (values.Length == 0) return sum;

for (int I = 0; I < values.Length; i++) sum += values[i];

return (sum / values.Length);

}

static void Main(String[] args) {

Console.WriteLine(“Avg with 2 arguments : “+ Average(10,20)); Console.WriteLine(“Avg with 4 arguments : “ + Average(10, 20,30,40)); Console.WriteLine(“Avg with 6 arguments : “ + Average(10, 20,30,40,50,60)); Console.ReadLine();

}

}

}

1. // Program using “ref” keyword . using System;

namespace PracticalFile

{

class MySample

{

static void SwapInteger(ref int x, ref int y)

{

int temp = x; x = y;

y = temp;

}

static void Main(String[] args) { int x = 4, y = 10;

Console.WriteLine(“Before swapping x = “+x+” and y = “+y); SwapInteger(ref x, ref y);

Console.WriteLine(“After swapping x = “ + x + “ and y = “ + y); Console.ReadLine();

}

}

}

**Practical- 5**

**Qs. WAP to demonstrate jagged array.**

using System; namespace PracticalFile

{

class MySample

{

static void Main(String[] args)

{

int i, j;

Console.Write("Enter the number of rows : "); int n = Convert.ToInt32(Console.ReadLine()); int col;

int[][] a = new int[n][];

Console.Write("\nEnter the no. of columns of each row : "); for (i = 0; i < n; i++)

{

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

}

Console.Write("\nEnter the elements of the array : "); for (i = 0; i < n; i++)

{

for (j = 0; j < a[i].Length; j++)

{

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

}

}

Console.WriteLine("Printing the array"); for (i = 0; i < n; i++)

{

for (j = 0; j < a[i].Length; j++)

{

Console.Write(a[i][j] + " ");

}

Console.WriteLine();

}

Console.ReadLine();

}

}

}

**Practical- 6**

**Qs. WAP to demonstrate System.String members like Contains(), Insert(), Remove(), Replace() and ToUpper().**

using System; namespace PracticalFile

{

class MySample

{

static void Main(String[] args)

{

String firstname = " Johnny",lastname="Wick"; Console.WriteLine("CompareTo() : "+firstname.CompareTo(lastname)); Console.WriteLine("Contains(\"nn\") : "+firstname.Contains("nn")); Console.WriteLine("EndsWith('y') : "+firstname.EndsWith("y")); Console.WriteLine("Equals() : "+firstname.Equals(lastname)); Console.WriteLine("GetHashCode() : "+firstname.GetHashCode()); Console.WriteLine("GetType() : " + firstname.GetType()); Console.WriteLine("GetTypeCode() : " + firstname.GetTypeCode()); Console.WriteLine("IndexOf('hn') : " + firstname.IndexOf("hn")); Console.WriteLine("ToLower() : " + firstname.ToLower()); Console.WriteLine("ToUpper() : " + firstname.ToUpper()); Console.WriteLine("Insert(0, 'Hello') : " + firstname.Insert(0, "Hello")); Console.WriteLine("LastIndexOf('n') : " + firstname.LastIndexOf("n")); Console.WriteLine("Length : " + firstname.Length); Console.WriteLine("Remove(5) : " + firstname.Remove(5)); Console.WriteLine("Replace('e', 'i') : " + firstname.Replace('e', 'i'));

String[] split = firstname.Split(new char[] { 'o' });

Console.WriteLine("Split[0] : "+split[0]); Console.WriteLine("Split[1] : "+split[1]); Console.WriteLine("StartsWith(' ') : " + firstname.StartsWith(" ")); Console.WriteLine("Substring(2, 5) : " + firstname.Substring(2, 5)); Console.WriteLine("ToCharArray() : " + firstname.ToCharArray()); Console.WriteLine("Trim() : " + firstname.Trim()); Console.ReadLine();

}

}

}

**Practical- 7**

**Qs. WAP to demonstrate encapsulation concept.**

using System; namespace PracticalFile

{

class Employee

{

private string empName;

public string GetName()

{

return empName;

}

public void setName(string name)

{

if (name.Length > 15) Console.WriteLine("Error, Longer than 15");

else

empName = name;

}

class MySample

{

static void Main(String[] args)

{

Employee empobj = new Employee(); Console.WriteLine("\t\tEncapsulation"); empobj.setName("John Wick");

Console.WriteLine("Employee Name : {0}", empobj.GetName()); Console.ReadKey();

}

}

}

}

**Practical- 8 Qs. WAP to demonstrate custom namespace.**

using System;

namespace first\_namespace

{

class namespace\_c1

{

public void func() {

Console.WriteLine("Inside first\_namespace");

}

}

}

namespace second\_space

{

class namespace\_c1

{

public void func() {

Console.WriteLine("Inside second\_namespace");

}

}

}

namespace PracticalFile

{

class Employee

{

private string empName;

public string GetName() { return empName;

}

public void setName(string name)

{

if (name.Length > 15)

Console.WriteLine("Error, Longer than 15");

else

}

empName = name;

class MySample

{

static void Main(String[] args) {

first\_namespace.namespace\_c1 fc = new first\_namespace.namespace\_c1(); second\_space.namespace\_c1 sc = new second\_space.namespace\_c1(); fc.func();

sc.func(); Console.ReadKey();

}

}

}

}

# Practical- 9 Qs. WAP to demonstrate custom namespace.

using System; namespace PracticalFile

{

class Property

{

private int n; public int Anumber

{

get

{

return n;

}

set

{

n = value;

}

}

}

class MySample {

static void Main(String[] args)

{

Property obj = new Property(); obj.Anumber = 100;

int m = obj.Anumber; Console.WriteLine("\t\tProperty Demo"); Console.WriteLine("The value is : {0} ", m); Console.ReadKey();

}

}

}

**Practical- 10 Qs. WAP to demonstrate Multiple Inheritance.**

using System; namespace PracticalFile

{

interface Addition

{

int sum();

}

interface Multiply

{

int mul();

}

class Multiple : Addition, Multiply

{

int a, b;

public void cal(int x, int y)

{

a = x; b = y;

}

public int sum()

{

return (a + b);

}

public int mul()

{

return a \* b;

}

class MySample

{

static void Main(String[] args) { Multiple c = new Multiple(); c.cal(50, 10);

Addition a = (Addition)c; Multiply m = (Multiply)c;

Console.WriteLine("\t\tMutiple Inheritance"); Console.WriteLine("Sum = " + a.sum()); Console.WriteLine("Multi = " + m.mul()); Console.ReadKey();

}

}

}

}

**Practical- 11**

**Qs. WAP to demonstrate overriding using new, virtual and override keyword.**

using System; namespace PracticalFile

{

class Shibi {

public void Hand() { Console.WriteLine("It is hand");

}

public virtual void eye() { Console.WriteLine("It is an eye");

}

}

class Pinki : Shibi

{

public void Hand()

{

Console.WriteLine("Pinki's hand");

}

public override void eye()

{

Console.WriteLine("Pinki's eye");

}

}

class MySample {

static void Main(String[] args) { Shibi s = new Pinki();

Pinki p = new Pinki(); s.Hand();

p.Hand();

s.eye();

p.eye(); Console.ReadKey();

}

}

}

**Practical- 12 Qs. WAP to demonstrate single cast delegate.**

using System;

delegate int NumberChanger(int n); namespace PracticalFile

{

class MySample

{

static int num = 10;

public static int AddNum(int p)

{

num += p; return num;

}

public static int MulNum(int q)

{

num \*= q; return num;

}

public static int GetNum()

{

return num;

}

public static void Main(String[] args) {

NumberChanger nc1 = new NumberChanger(AddNum); NumberChanger nc2 = new NumberChanger(MulNum); nc1(125);

Console.WriteLine("Value of Num : {0}", GetNum()); nc2(5);

Console.WriteLine("Value of Num : {0}", GetNum()); Console.ReadKey();

}

}

}

**Practical- 13 Qs. WAP to demonstrate multicast delegate.**

using System;

delegate int NumberChanger(int n); namespace PracticalFile

{

class MySample

{

static int num = 10;

public static int AddNum(int p)

{

num += p; return num;

}

public static int MultNum(int q)

{

num \*= q; return num;

}

public static int getNum()

{

return num;

}

public static void Main(String[] args)

{

NumberChanger nc;

NumberChanger nc1 = new NumberChanger(AddNum); NumberChanger nc2 = new NumberChanger(MultNum); nc = nc1;

nc += nc2;

nc(5); //calling multicast Console.WriteLine("Value of Num: {0}", getNum()); Console.ReadKey();

Console.ReadKey();

}

}

}

# Practical- 14 Qs. WAP to demonstrate exception handling.

using System; namespace PracticalFile

{

class MySample

{

public static void Main(String[] args)

{

int a = 10, b = 5, c = 5, x, y; try

{

x = a / (b - c);

}

catch (Exception e)

{

Console.WriteLine("Division by Zero"); Console.WriteLine(e.Message);

}

finally

{

Console.WriteLine("Finally block will run irrespective of exception occured or not");

}

y = a / (b + c); Console.WriteLine("Y = " + y); Console.ReadKey();

}

}

}

**Practical- 15 Qs. WAP to demonstrate indexer.**

using System; namespace PracticalFile

{

class List

{

string[] array = new string[3]; public string this[int idx]

{

get

{

if (idx < 0 || idx >= array.Length)

{

return null;

}

else

{

return (array[idx]);

}

}

set

{

array[idx] = value;

}

}

}

class MySample

{

static int i;

public static void Main(String[] args)

{

List l = new List(); l[0] = "123";

l[1] = "abc";

l[2] = "xyz123";

Console.WriteLine("List members using indexer : "); for (i = 0; i < 3; i++)

Console.WriteLine(l[i]); Console.ReadKey();

}

}

}