**Tutorial - 4:**

1. **The employee list for a company contains employee code, name, designation and basic pay. The employee is given a house rent allowance (HRA) of 10% of the basic pay and dearness allowance (DA) of 45% of the basic pay. The total pay of the employee is calculated as Basic Pay + HRA + DA. Write a class to define the details of the employee. Write a constructor to assign the required initial values. Add a method to calculate HRA, DA and total pay and print them. Write another class with main method. Create objects for three different employees and calculate HRA, DA and total pay.**

**Ans.**

//tt4-1

using System;

class program

{

    static int computeSalary(int basic, char grade)

    {

        double allowance;

        double hra, da, pf;

        hra = 0.2 \* basic;

        da = 0.5 \* basic;

        pf = 0.11 \* basic;

        if (grade == 'A')

        {

            allowance = 1700.0;

        }

        else if (grade == 'B')

        {

            allowance = 1500.0;

        }

        else

        {

            allowance = 1300.0;

        }

        double gross;

        Console.WriteLine("HRA : {0}", hra);

        Console.WriteLine("DA : {0}", da);

        gross = Math.Round(basic + hra + da + allowance - pf);

        return (int)gross;

    }

    public static void Main(String[] args)

    {

        int basic = 10000;

        char grade = 'A';

        Console.WriteLine("\nEmployee 1 : ");

        Console.WriteLine("Total Pay : {0}", computeSalary(basic, grade));

        basic = 2000;

        grade = 'B';

        Console.WriteLine("\nEmployee 2 : ");

        Console.WriteLine("Total Pay : {0}", computeSalary(basic, grade));

        basic = 6000;

        grade = 'A';

        Console.WriteLine("\nEmployee 3 : ");

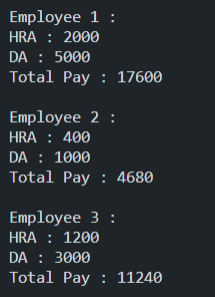
        Console.WriteLine("Total Pay : {0}", computeSalary(basic, grade));

        Console.ReadLine();

    }

}

**Output:**

****

1. **From the following code and given output, complete missing statements and find out error code and correct it.**

//Base class or Parent class.

class Shape

{

    public double Width;

    public double Height;

    public void ShowDim()

    {

        Console.WriteLine("Width and height are " +  Width + " and " + Height);

    }

}

// Triangle is derived from Shape.

//Drived class or Child class.

class Triangle : Shape

{

    public string Style; // style of triangle

    // Return area of triangle.

 //………………………………Missing statement-1……………………………….//

    {

        return Width \* Height / 2;

    }

    // Display a triangle's style.

    public void ShowStyle()

    {

//………………………………Missing statement-2……………………………….//

    }

}

//Driver class which runs the program.

class Driver

{

    static void Main()

    {

        Triangle t1 new Triangle();

        Triangle t2 new Triangle();

        t1.Width =4.0;

        t1.Height =4.0;

        t1.Style ="isosceles";

        t2.Width =8.0;

        t2.Height =12.0;

        t2.Style ="right";

        Console.WriteLine("Info for t1: ");

        t1.ShowStyle();

        t1.ShowDim();

        Console.WriteLine("Area is " + t1.Area());

        Console.WriteLine();

        Console.WriteLine("Info for t2: ");

        t2.ShowStyle();

        t2.ShowDim();

        Console.WriteLine("Area is " + t2.Area());

    }

}

Output  
Info for t1:  
Triangle is isosceles  
Width and height are 4 and 4  
Area is 8

Info for t2:  
Triangle is right  
Width and height are 8 and 12  
Area is 48

**Ans.**

//tt4-2

using System;

class Shape

{

    public double Width, Height;

    public void ShowDim()

    {

        Console.WriteLine("Width and height are " + Width + " and " + Height);

    }

}

class Triangle : Shape

{

    public string? Style;

    public double Area()

    {

        return Width \* Height / 2;

    }

    public void ShowStyle()

    {

        Console.WriteLine(Style);

    }

}

class Driver

{

    public static void Main(string[] args)

    {

        Triangle t1 = new Triangle();

        Triangle t2 = new Triangle();

        t1.Width = 4.0;

        t1.Height = 4.0;

        t1.Style = "isosceles";

        t2.Width = 8.0;

        t2.Height = 12.0;

        t2.Style = "right";

        Console.WriteLine("Info for t1 : ");

        t1.ShowStyle();

        t1.ShowDim();

        Console.WriteLine("Area is " + t1.Area());

        Console.WriteLine();

        Console.WriteLine("Info for t2 : ");

        t2.ShowStyle();

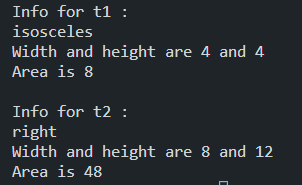
        t2.ShowDim();

        Console.WriteLine("Area is " + t2.Area());

    }

}

**Output:**



1. **Draw a real picture for single level inheritance. Perform following tasks.**

Task 1: Create a class  
Task 2: Add few data members as private, protected and public  
Task 3: Add few methods as public to work on defined data members  
Task 4: Create another applicable class which inherits members from above class  
Task 5: Add few data members as private, protected and public into second class  
Task 6: Add few methods as public to work on defined data members into second class  
Task 7: Create a Demo class with main method.  
Task 8: Create at least two objects of a second class defined in Task 4 into main method and call all methods using that object.  
Task 9: Write comment for each important portion of code like data members’ declaration, methods, some important logic etc.  
Task 10: Summarize above solution in your own few words to visualize the solution to the end user.

**Ans.**

//tt4-3

class A1

{

    public Double width, height;

    public void get\_data(double width, double height)

    {

        this.width = width;

        this.height = height;

    }

}

class B1 : A1

{

    public void area()

    {

        Console.WriteLine("Area of Rectangle is : {0}", (width \* height));

    }

}

class program

{

    public static void Main(string[] args)

    {

        B1 obj1 = new B1();

        Console.Write("\nEnter width : ");

        obj1.width = Convert.ToDouble(Console.ReadLine());

        Console.Write("Enter height : ");

        obj1.height = Convert.ToDouble(Console.ReadLine());

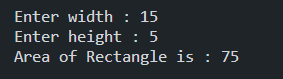
        obj1.area();

        Console.ReadLine();

    }

}

**Output:**



1. **From the following code and given output complete missing statements and find out error code and correct it.**

using System;

namespace StaticVarApplication

{

    class StaticVar

    {

       public static int num;

        public counting()

        {

            num++;

        }

//………………………………Missing statement……………………………….//

        {

            return num;

        }

    }

    class StaticTester

    {

        static void Main(string[] args)

        {

            StaticVar s = new StaticVar();

            s.count();

            s.count();

            s.count();

            Console.WriteLine("Variable num: {0}", StaticVar.getNum());

            Console.ReadKey();

        }

    }

}

Output:

Variable num: 3

**Ans.**

//tt4-4

using System;

namespace StaticVarApplication

{

    class StaticVar

    {

        public static int num;

        public void counting()

        {

            num++;

        }

        public int getNum()

        {

            return num;

        }

    }

    class StaticTester

    {

        public static void Main(string[] args)

        {

            StaticVar s = new StaticVar();

            s.counting();

            s.counting();

            s.counting();

            Console.WriteLine("\nVariable num : {0}", s.getNum());

            Console.ReadLine();

        }

    }

}

**Output:**

****

1. **Find out error code and correct it. Print appropriate output as desired.**

using System;

public class A // This is the base class.

{

    public A(int value)

    {

     // Executes some code in the constructor.

     Console.WriteLine("Base constructor A()");

    }

}

public class B : A // This class derives from the previous class.

{

    public B(int value) : base(value)

    {

     // The base constructor is called first.

     // ... Then this code is executed.

     Console.WriteLine("Derived constructor B()");

    }

}

class Program

{

    static void Main()

    {

     // Create a new instance of class A, which is the base class.

     // ... Then create an instance of B, which executes the base constructor.

     A a = A(0);

     B b = B(1);

    }

}

**Ans.**

//tt4-5

using System;

class A

{

    public A(int value)

    {

        Console.WriteLine("Base constructor A()");

    }

}

class B : A

{

    public B(int value) : base(value)

    {

        Console.WriteLine("Derived constructor B()");

    }

}

class Program

{

    public static void Main()

    {

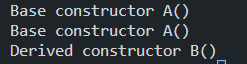
        A a = new A(0);

        B b = new B(1);

    }

}

**Output:**

****

1. **Find out error code and correct it. Print appropriate output as desired.**

using System;

abstract class Test

{

    int a;

    abstract void A();

}

class Example1 : Test

{

    public override void A()

    {

     Console.WriteLine("Example1.A");

     base.a++;

    }

}

class Example2 : Test

{

    public override void A()

    {

     Console.WriteLine("Example2.A");

     base.a--;

    }

}

class Program

{

    static void Main()

    {

     // Reference Example1 through Test type.

     Test test1 = new Example1();

     test1.A();

     // Reference Example2 through Test type.

     Test test2 = new Example2();

     test2.A();

    }

}

**Ans.**

//tt4-6

using System;

abstract class Test

{

    public int a;

    public abstract void A();

}

class Example1 : Test

{

    public override void A()

    {

        Console.WriteLine("Example1.A");

        base.a++;

    }

}

class Example2 : Test

{

    public override void A()

    {

        Console.WriteLine("Example2.A");

        base.a--;

    }

}

class Program

{

    static void Main()

    {

        Test test1 = new Example1();

        test1.A();

        Test test2 = new Example2();

        test2.A();

    }

}

**Output:**

****

1. **Refer given output and find out error code and correct it.**

sealed class A

{

        public int x;

        public int y;

}

class B : A

{

        public int z;

}

class SealedTest2

{

        static void Main()

        {

            A sc = new A();

            sc.x = 110;

            sc.y = 150;

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

        }

}

Output

x = 110, y = 150

**Ans.**

//tt4-7

sealed class A

{

    public int x, y;

}

class program

{

    public static void Main()

    {

        A sc = new A();

        sc.x = 110;

        sc.y = 150;

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

        Console.ReadLine();

    }

}

**Output:**

****

1. **Find out error code and correct it. Print appropriate output as desired.**

 class X

    {

        protected virtual void F() { Console.WriteLine("X.F"); }

        protected virtual void F2() { Console.WriteLine("X.F2"); }

    }

    class Y : X

    {

        sealed protected override void F() { Console.WriteLine("Y.F"); }

        protected override void F2() { Console.WriteLine("Y.F2"); }

    }

    class Z : Y

    {

        // Overriding F

        protected override void F() { Console.WriteLine("Z.F"); }

        // Overriding F2

        protected override void F2() { Console.WriteLine("Z.F2"); }

    }

   class SealedMethodTest

   {

        static void Main()

        {

            X Obj1 = new X();

            Obj1.F();

            Obj2.F2();

            Y Obj2 = new Y();

            Obj2.F();

            Obj2.F2();

            Z Obj3 = new Z();

            Obj3.F();

            Obj3.F2();

         }

   }

**Ans.**

//tt4-8

using System;

class X

{

    public virtual void F() { Console.WriteLine("X.F"); }

    public virtual void F2() { Console.WriteLine("X.F2"); }

}

class Y : X

{

    sealed public override void F() { Console.WriteLine("Y.F"); }

    public override void F2() { Console.WriteLine("Y.F2"); }

}

class Z : Y

{

    //public override void F() { Console.WriteLine("Z.F"); }

    public override void F2() { Console.WriteLine("Z.F2"); }

}

class program

{

    public static void Main()

    {

        X Obj1 = new X();

        Obj1.F();

        Obj1.F2();

        Y Obj2 = new Y();

        Obj2.F();

        Obj2.F2();

        Z Obj3 = new Z();

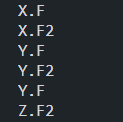
        Obj3.F();

        Obj3.F2();

    }

}

**Output:**

****

1. **This program will throw an exception. Add try, catch and finally blocks to handle this exception.**

using System;  
class MyClient  
{  
      public static void Main()  
      {  
            int x = 0;  
            int div = 100/x;  
            Console.WriteLine(div);  
       }  
}

**Ans.**

//tt4-9

using System;

class program

{

    public static void Main()

    {

        int x = 0;

        int div = 0;

        try

        {

            div = 100 / x;

            Console.WriteLine("This line will be not executed.");

        }

        catch (DivideByZeroException)

        {

            Console.WriteLine("Exception occured.");

        }

        Console.WriteLine($"Result is {div}");

    }

}

**Output:**

****

1. **Arrange the code to get desirable output**

class MyException : Exception  
{

}  
class MyClient  
{  
    public static void Main()  
    {  
         try  
         {  
                 throw new MyException("my exception generated.");  
         }  
         catch(Exception e)  
         {  
                  
         }  
         Console.WriteLine("LAST STATEMENT");  
    }

                using System;

                public MyException(string str)  
    {  
              Console.WriteLine("User defined exception");  
              Console.WriteLine("Exception caught here: " + e.ToString());

                 }  
  
}

           Output:

           Exception caught here: my exception generated.

           LAST STATEMENT

**Ans.**

//tt4-10

using System;

class MyException : Exception

{

    public MyException(string str)

    {

        Console.WriteLine("User defined exception");

    }

}

class MyClient

{

    public static void Main()

    {

        try

        {

            throw new MyException("AB");

        }

        catch (Exception e)

        {

            Console.WriteLine("Exception caught here " + e.ToString());

        }

        Console.WriteLine("LAST STATEMENT");

    }

}

**Output:**

****

****