Dt : 21/10/2022

ClassFiles:

SubClass1.class

SubClass1$1SubClass2.class

SubClass1$1SubClass22.class

DemoInnerClass3.class(MainClass)

==========

(2)Anonymous InnerClasses:

=>The InnerClasses which are declared without name are known as Anonymous InnerClasses

=>These Anonymous InnerClasses are categorized into two types:

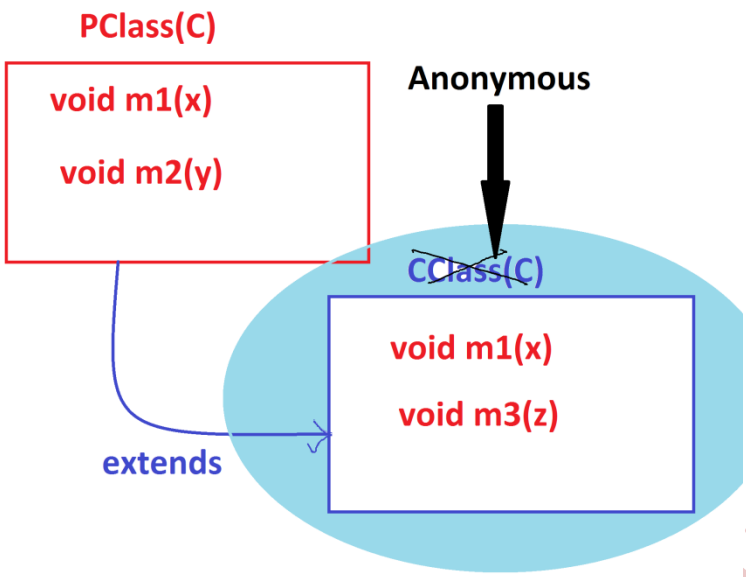
(a)Anonymous InnerClass as Class extention

(b)Anonymous InnerClass as Implementation class

(a)Anonymous InnerClass as Class extention:

=>The process of declaring the CClass without name is known as "Anonymous InnerClass as Class extention".

diagram:



syntax:

class PClass

{

//PClass_body

}

PClass ob = new PClass()

{

//CClass_body

};

Ex-program:

PClass.java

```
package test;
```

```
public class PClass {
    public void m1(int x) {
     System.out.println("====PClass m1(x)====");
     System.out.println("The value x:"+x);
    }
    public void m2(int y) {
     System.out.println("====PClass m2(y)====");
     System.out.println("The value y:"+y);
    }
}
```

DemoAnonymous1.java(MainClass)

```
package maccess;
import test.*;
public class DemoAnonymous1 {
     public static void main(String[] args) {
          //Anonymous InnerClass as Class extention
        PClass ob = new PClass()
        {
          public void m1(int x) //Overriding method
          {
               System.out.println("====CClass m1(x)====");
               System.out.println("The value x:"+x);
            }
            public void m3(int z)//NonOverriding method
            {
               System.out.println("====CClass m3(z)====");
               System.out.println("The value z:"+z);
            }
        };
        ob.m1(11);
        ob.m2(12);
        //ob.m3(13);
     }
}
```

o/p:

====CClass m1(x)====

The value x:11

====PClass m2(y)====

The value y:12

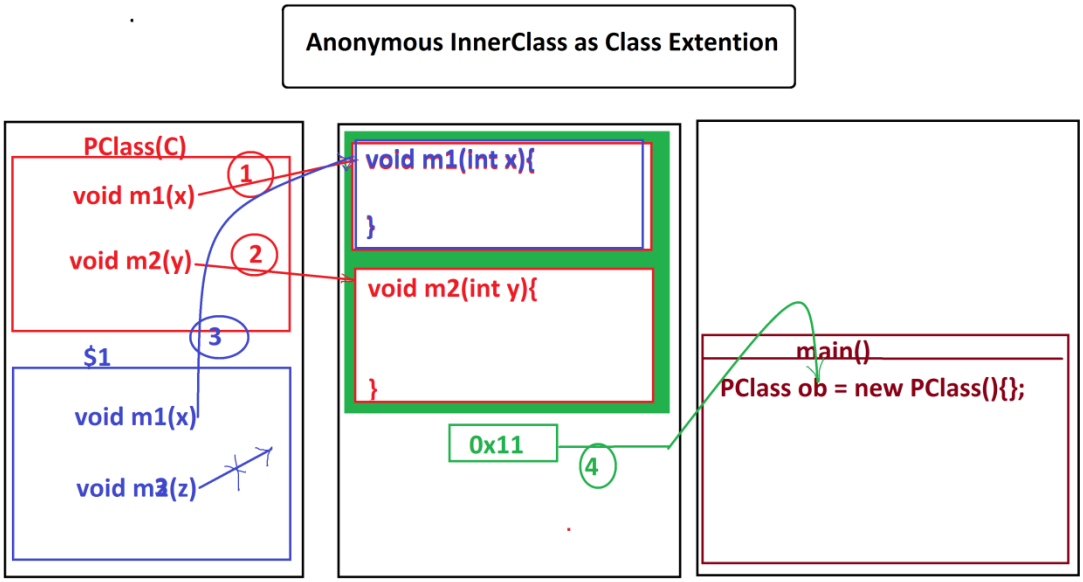
Execution flow of above program:

ClassFiles:

PClass.class

DemoAnonymous1.class(MainClass)

DemoAnonymous1$1.class



Note:

=>Generalization process using Classes model is modified as "Anonymous InnerClass as Class extention"

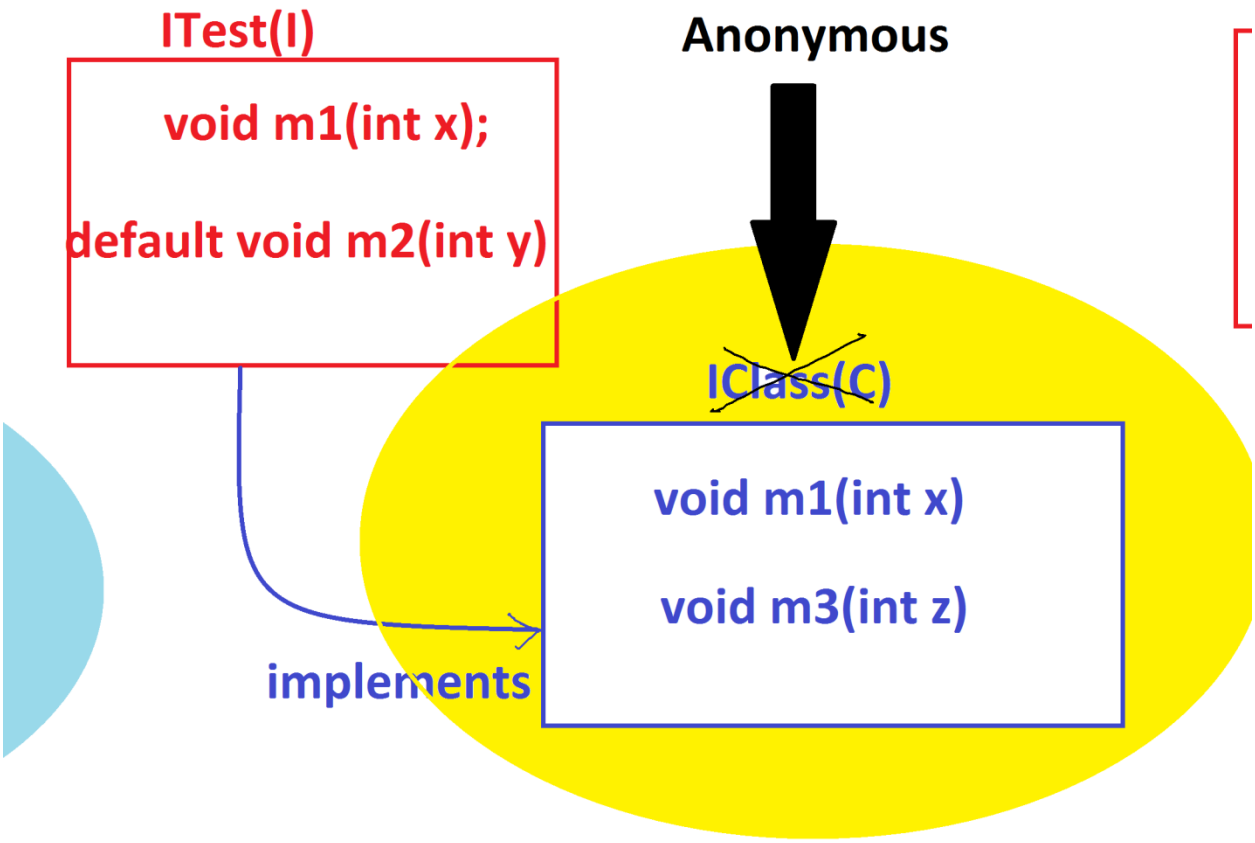
==========

**imp*

(b)Anonymous InnerClass as Implementation class:

=>The process of declaring implementation class without name is known as "Anonymous InnerClass as Implementation Class".

Diagram:



syntax:

```
interface ITest
{
 //Interface_body
}
```

ITest ob = new ITest()

{

//IClass_body

};

Ex:

ITest.java

```
package test;
public interface ITest {
   public abstract void m1(int x);
   public default void m2(int y) {
        System.out.println("====default m2(y)====");
        System.out.println("The value y:"+y);
   }
}
```

DemoAnonymous2.java(MainClass)

```
package maccess;
import test.*;
public class DemoAnonymous2 {
     public static void main(String[] args) {
       ITest ob = new ITest()
       {
         public void m1(int x)//Implemented and Overriding
method
         {
          System.out.println("===method m1(x)===");
          System.out.println("The value x:"+x);
         }
         public void m3(int z)//NonImplemented and NonOverriding
method
         {
          System.out.println("===method m3(z)===");
          System.out.println("The value z:"+z);
         }
     };
       ob.m1(11);
       ob.m2(12);
```

```
        //ob.m3(13);
    }
}
```

o/p:

===method m1(x)===

The value x:11

====default m2(y)====

The value y:12

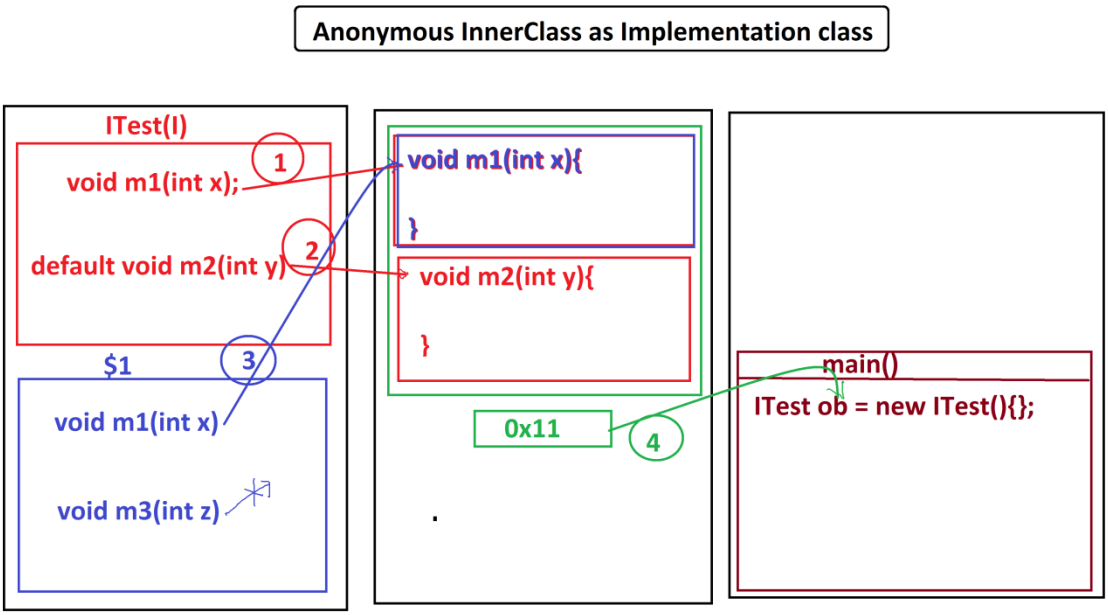
Execution flow of above program:

ClassFiles:

ITest.class

DemoAnonymous2.class(MainClass)

DemoAnonymous2$1.class



Note:

=>Generalization process using Interfaces model is modified as "Anonymous InnerClass as implementation class"

==========

Ex-Program:

Convert IComparable application into 'Anonymous InnerClass as implementation class" model.

IComparable.java

```
package test;
public interface IComparable {
    public abstract int compare(int x,int y);
}
```

DemoAnonymous3.java(MainClass)

```
package maccess;

import test.*;

import java.util.*;

public class DemoAnonymous3 {

        public static void main(String[] args) {

    Scanner s = new Scanner(System.in);

    System.out.println("Enter the value x:");

    int x = s.nextInt();

    System.out.println("Enter the value y:");

    int y = s.nextInt();
```

```
System.out.println("====Choice====");
System.out.println("1.GreaterValue\n2.SmallerValue");
System.out.println("Enter the Choice:");
switch(s.nextInt())
{
case 1:
        //GreaterVlaue class as Anonymous
        IComparable gv = new IComparable()
        {
                public int compare(int x,int y) {
                        if(x>y) return x;
                        else return y;
                }
        };
        int r1 = gv.compare(x, y);
        System.out.println("GreaterValue:"+r1);
        break;
case 2:
        //SmallerValue class as Anonymous
        IComparable sv = new IComparable()
        {
                        public int compare(int x,int y) {
                                if(x<y) return x;
                                else return y;
```

```
                    }
                };
                int r2 = sv.compare(x, y);
                System.out.println("SmallerValue:"+r2);
            break;
        default:
            System.out.println("Invalid Choice...");
        }//end of switch
        s.close();
        }
}
```

ClassFiles:

IComparable.class

DemoAnonymous3.class(MainClass)

DemoAnonymous3$1.class

DemoAnonymous3$2.class

==========

Assignment-1:

Convert IArithmetic application into "Anonymous InnerClass as implementation class" model

Note:

Addition,Subtraction,Multiplication,Division,ModDivision class as Anonymous

Assignment-2:

Convert BankTransaction application into "Anonymous InnerClass as implementation class" model

Note:

WithDraw and Deposit classes as Anonymous

==========