

# **LAPORAN PRAKTIKUM**

## **PEMROGRAMAN BERORIENTASI OBJEK**

### **PERTEMUAN KE-13**



**Disusun Oleh :**

**NAMA : Raden Isnawan Argi Aryasatya**

**NIM : 195410257**

**JURUSAN : Teknik Informatika**

**JENJANG : S1**

**KELAS : TI-5**

**Laboratorium Terpadu**  
**Sekolah Tinggi Manajemen Informatika Komputer**  
**AKAKOM**  
**YOGYAKARTA**

**2021**

## PERTEMUAN KE-13 (INNER CLASS)

### TUJUAN

Dapat mendefinisikan inner class, nested class, non-static inner class, local class dan Anonymous class, serta dapat mengimplementasi konsep inner class pada aplikasi

### DASAR TEORI

Java mengizinkan kita mendefinisikan suatu kelas di dalam kelas lain, hal ini disebut sebagai nested class. Disebut nested karena class bersifat tersarang terhadap kelas – kelas utamanya, seperti halnya blok penyeleksian (if, for) yang tersarang pada blok penyeleksian lainnya atau method yang tersarang pada method lainnya. nested class dibagi menjadi dua kategori, yaitu static dan non static nested class.

---

### PRAKTIK 1: Tulislah program file Luar.java berikut

```
Luar.java x
public class Luar{
    private int angkaLuar;
    public Luar(int angkaLuar){

        this.angkaLuar=angkaLuar;
    }
    int getAngkaLuar(){
        return angkaLuar;
    }
    //inner class
    class Dalam{
        private int angkaDalam;
        public Dalam(){
            angkaDalam=9;
        }
        public void cetakDalam(){
            System.out.println("Ini angka luar:"+angkaLuar);
            System.out.println("Ini angka dalam:"+angkaDalam);
        }
    } //batas inner class

    public void cetakLuar(){
        Dalam dl=new Dalam();
        dl.cetakDalam();
    }

    public static void main(String args[]){
        Luar lu=new Luar(5);
        lu.cetakLuar();
    }
}
```

output:

```
C:\Windows\system32\cmd.exe
Ini angka luar:5
Ini angka dalam:9
Press any key to continue . . .
```

penjelasan:

Inner Class adalah class didalam outer class. Class itu sendiri merupakan sebuah kerangka model/blueprint yang digunakan sebagai tempat menaruh atribut seperti variable, method, konstruktor, dll. Dalam pemrograman java, kita dapat menambahkan sebuah class (kelas dalam) didalam class (kelas luar) atau bisa disebut juga Nested Class. Inner class mempunyai hak akses

pada atribut atau method yang berada pada outer class dalam seluruh kondisi modifier termasuk modifier private. Sebaliknya class luar tidak memiliki hak akses pada inner class. Pada dasarnya Inner Class atau Nested Class digunakan untuk mengelompokkan class pada satu tempat serta Menciptakan code yang readable dan maintainable.

Untuk pembuatan program di atas, kita menerapkan konsep yang sudah saya jelaskan tersebut dengan membuat class yang akan menjadi outer class yaitu public class Luar. Di dalamnya ada atribut bermodifier private bernama angkaLuar. Private artinya method dan variabel hanya dapat diakses oleh class yang sama. Dalam kasus program ini, inner class yang akan kita buat nanti bisa mengakses atribut angkaLuar karena dianggap sebagai satu kesatuan.

Selanjutnya kita buat konstruktor yang di dalamnya kita mengakses atribut angkaLuar dengan keyword this. this adalah keyword pada bahasa pemrograman java yang berfungsi untuk mereferensikan atau mengacu ke atribut yang bersangkutan. Kemudian kita membuat method getter bernama getAngkaLuar() ber-modifier integer yang akan menampung nilai balik dari data atribut angkaLuar.

Lalu kita lanjutkan dengan membuat inner class bernama class Dalam. Di dalamnya kita deklarasikan atribut angkaDalam bermodifier private. Untuk atribut private milik inner class tidak bisa diakses oleh outer class. Lalu kita buat konstruktor yang di dalamnya kita memberi value kepada atribut angkaDalam. Value atau nilai yang diberikan adalah angka 9. Berikutnya kita buat method untuk mencetak kalimat pada output sekaligus mencetak angka yang nanti akan dimasukkan. Berikut yang akan kita cetak dengan method cetakDalam() :

```
System.out.println("Ini angka luar:"+angkaLuar);  
System.out.println("Ini angka dalam:"+angkaDalam);
```

sekarang kita kembali lagi di outer class. Kita langsung melakukan instansiasi (pembuatan objek) untuk membuat objek bernama dl yang akan menampilkan hasil method cetakDalam() yaitu dengan menuliskan:

```
public void cetakLuar(){  
    Dalam dl=new Dalam();  
    dl.cetakDalam();
```

terakhir kita membuat main method. Eksekusi kita awali dengan menuliskan **public static void main(String args[]){**. Di dalamnya kita instansiasi dengan membuat objek bernama lu yang kita beri value 5 lalu kita cetak dengan method cetakLuar().

## PRAKTIK 2: Tulislah program file RoundShape.java berikut

```
RoundShape.java x Luar.java  
public abstract class RoundShape  
{  
    protected class Center {  
        int x,y;  
    }  
    protected Center C = new Center();  
    protected float radiusOfCircle;  
  
    public RoundShape(int xCenter, int yCenter, float radius)  
    {  
        C.x=xCenter;  
        C.y=yCenter;  
        radiusOfCircle = radius;  
    }  
    abstract public float area();  
}
```

penjelasan:

pada praktik 2 ini, kita membuat outer class yang merupakan abstract class bernama RoundShape. Abstract class adalah kelas yang mengandung satu method abstrak atau lebih, kelas abstrak tersebut digunakan hanya untuk membuat sebuah method yang tanpa ada implementasinya secara langsung. Kemudian langsung saja kita buat inner classnya bernama Center. Inner class center ber-modifier protected yang bisa diakses melalui inheritance. Di dalam inner class tersebut kita deklarasikan dua variabel bernama x dan y yang bertipe data integer. Lalu kita lakukan instansiasi dengan membuat objek bernama C dari class Center. Kita lakukan itu dengan menuliskan **protected Center C = new Center();** lalu dilanjutkan dengan mendeklarasikan atribut protected bernama radiusOfCircle.

Kemudian kita buat konstruktor untuk mengakses atribut x dan y untuk kemudian diinisialisasi dengan objek C dan sekaligus diberi value. Konstruktor kita buat dengan menuliskan **public RoundShape(int xCenter, int yCenter, float radius).** Lalu baru di dalamnya kita beri value pada setiap atribut dari class RoundShape maupun atribut dari class Center yaitu dengan menuliskan:

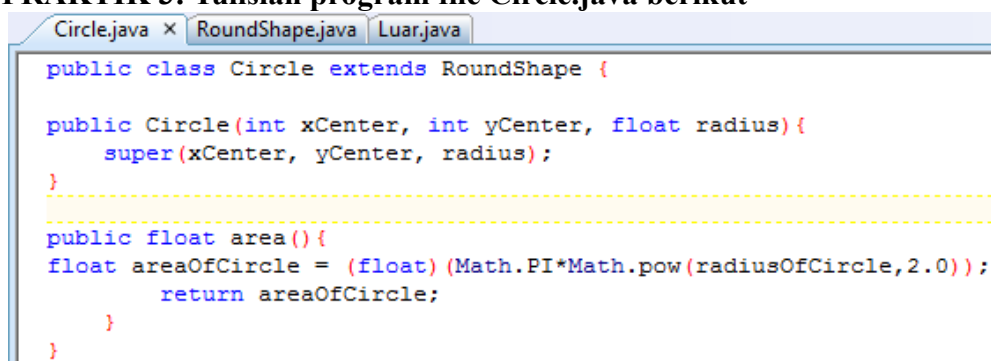
**C.x=xCenter;**

**C.y=yCenter;**

**radiusOfCircle = radius;**

terakhir kita buat method abstrak (method yang tanpa ada implementasinya secara langsung) dengan menuliskan **abstract public float area();**

### PRAKTIK 3: Tulislah program file Circle.java berikut



```
Circle.java x RoundShape.java Luar.java
public class Circle extends RoundShape {

    public Circle(int xCenter, int yCenter, float radius){
        super(xCenter, yCenter, radius);
    }

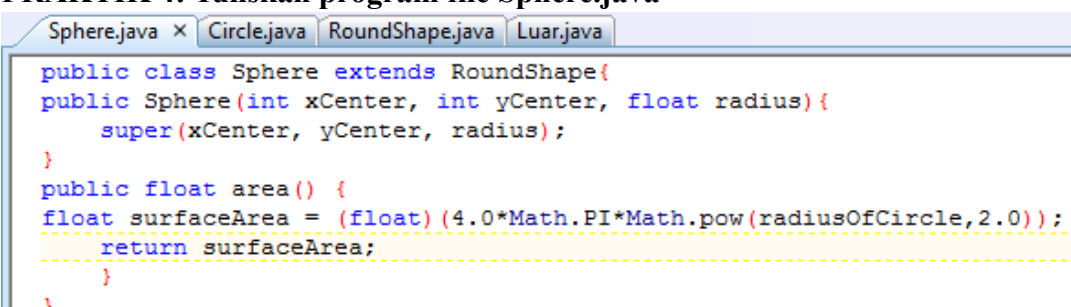
    public float area(){
        float areaOfCircle = (float) (Math.PI*Math.pow(radiusOfCircle,2.0));
        return areaOfCircle;
    }

}
```

penjelasan:

program Circle.java merupakan program subclass yang menerima pewarisan dari superclass RoundShape. Di dalamnya kita buat konstruktor untuk mengakses setiap variabel di superclass RoundShape yaitu dengan menuliskan **public Circle(int xCenter, int yCenter, float radius).** Di dalamnya kita akses dengan keyword super karena kita mengakses atribut dari parent class. Kita lakukan dengan menuliskan **super(xCenter, yCenter, radius);**. Selanjutnya kita buat method bertipe data float bernama area(). Di dalamnya merupakan perhitungan atau kalkulasi untuk luas lingkaran.

### PRAKTIK 4: Tuliskan program file Sphere.java



```
Sphere.java x Circle.java RoundShape.java Luar.java
public class Sphere extends RoundShape{
    public Sphere(int xCenter, int yCenter, float radius){
        super(xCenter, yCenter, radius);
    }

    public float area() {
        float surfaceArea = (float) (4.0*Math.PI*Math.pow(radiusOfCircle,2.0));
        return surfaceArea;
    }

}
```

Di dalam program Sphere.java ini kita buat konstruktor untuk mengakses setiap variabel di superclass RoundShape yaitu dengan menuliskan **public Circle(int xCenter, int yCenter, float radius)**. Di dalamnya kita akses dengan keyword super karena kita mengakses atribut dari parent class. Kita lakukan dengan menuliskan **super(xCenter, yCenter, radius);**. kita buat method bertipe data float bernama area() yang dalamnya ada kalkulasi untuk menghitung luas permukaan bola.

### PRAKTIK 5: Tuliskan TestRoundShape.java

```
TestRoundShape.java x Sphere.java Circle.java RoundShape.java Luar.java
import java.text.DecimalFormat;

public class TestRoundShape{
    public static void main(String args[]){

        DecimalFormat digit = new DecimalFormat ("0.##");
        Circle lingk1=new Circle(20,5,10.0f);
        System.out.println(digit.format(lingk1.area()));
    }
}
```

output:

```
C:\Windows\system32\cmd.exe
314,16
Press any key to continue . . .
```

penjelasan:

program di atas merupakan program yang memuat main method untuk pemrosesan pembuatan objek dan pendeklarasian output. Lalu kita deklarasikan **import java.text.DecimalFormat** supaya kita dapat mengontrol display dari leading dan trailing zeros, prefixes dan suffixes, grouping(ribuan) dan decimal. Dengan DecimalFormat inilah kita akan mendapatkan fleksibilitas dalam memformat angka atau nomer walaupun ini akan membuat kode program kita menjadi lumayan kompleks. Untuk mengawali eksekusi, kita tulis **public static void main(String args[])**. Lalu langsung saja kita buat objek digit dari class DecimalFormat dengan value ("0.##"). kemudian kita buat objek lingk1 dari class Circle untuk memberi value ke method public float area milik class Circle. Hal itu kita lakukan dengan menuliskan **Circle lingk1=new Circle(20,5,10.0f);**. Terakhir, kita tampilkan output dengan format decimal yaitu dengan menuliskan: **System.out.println(digit.format(lingk1.area()));**

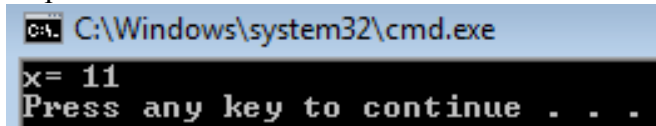
### PRAKTIK 6: membuat Anonymous Class

```
TestAnonymous.java x TestRoundShape.java RoundShape.java Sphere.java
class Awal{
    int x=8;

    void methodAwal(){
        System.out.println("Nilai x= "+x);
    }
}

public class TestAnonymous{
    public static void main(String args[]){
        Awal noName=new Awal(){ //instant anonymous class
            void methodAwal(){
                x+=3;
                System.out.println("x= "+x);
            }
        };
        noName.methodAwal();
    }
}
```

output:



```
C:\Windows\system32\cmd.exe
x= 11
Press any key to continue . . .
```

penjelasan:

Anonymous Class adalah salah satu konsep dari class atau interface pada java yang digunakan untuk mendeklarasikan dan menginisialisasi objek dari class lain dalam waktu bersamaan. Anonymous Class hampir sama dengan Inner Class, hanya saja tidak memiliki nama dan hanya memiliki body. Dengan menggunakan Class Anonymous, kita tidak perlu menggunakan kata kunci implements untuk mengimplementasikan interface pada kelas yang diwariskannya, kita hanya perlu membuat Instance dari class tersebut pada class yang ingin diwariskan.

Untuk proses pembuatan programnya, kita buat class bernama Awal yang memiliki atribut x=8 dan method bernama void methodAwal() yang mencetak kalimat "Nilai x= "+x. Kemudian langsung kita buat main method bernama TestAnonymous, yang di dalamnya kita buat objek bernama NoName yang merupakan proses pembuatan objek anonymous class. Lalu kita deklarasikan methodAwal() untuk menambahkan 3 ke atribut class Awal yaitu 8 sehingga diperoleh hasil 11 untuk objek noName. Kemudian kita print hasil dengan System.out.println("x= "+x); lalu dengan noName.methodAwal();.

---

## KESIMPULAN

pada laporan 13 ini, kita belajar banyak tentang inner class. Setelah mengerjakan 6 praktik di atas, bisa disimpulkan bahwa Nested class merupakan member dari kelas Outer(kelas terluar). Non static nested class (inner class) dapat mengakses semua member dari kelas Outer bahkan yang mempunyai hak akses private. Static nested class tidak mempunyai akses kepada member dari kelas Outer. Sebagai member dari Outer class, sebuah nested class dapat dideklarasikan dengan hak akses private, public, protected, sedangkan Outer class hanya bisa dideklarasikan dengan hak akses public.

Terakhir kita mempraktekkan membuat program Anonymous Class. Anonymous inner class adalah sebuah inner class yang dideklarasikan tanpa memberikan nama kelas. Anonymous class akan membuat kode program menjadi lebih ringkas karena dengan Anonymous class kita dapat mendeklarasikan dan meng-instant suatu kelas sekaligus dalam satu langkah.

**Terima Kasih**