IPB University
Bogor Indonesia

KOM120C -- PEMROGRAMAN

Object Oriented Programming

Inner Class

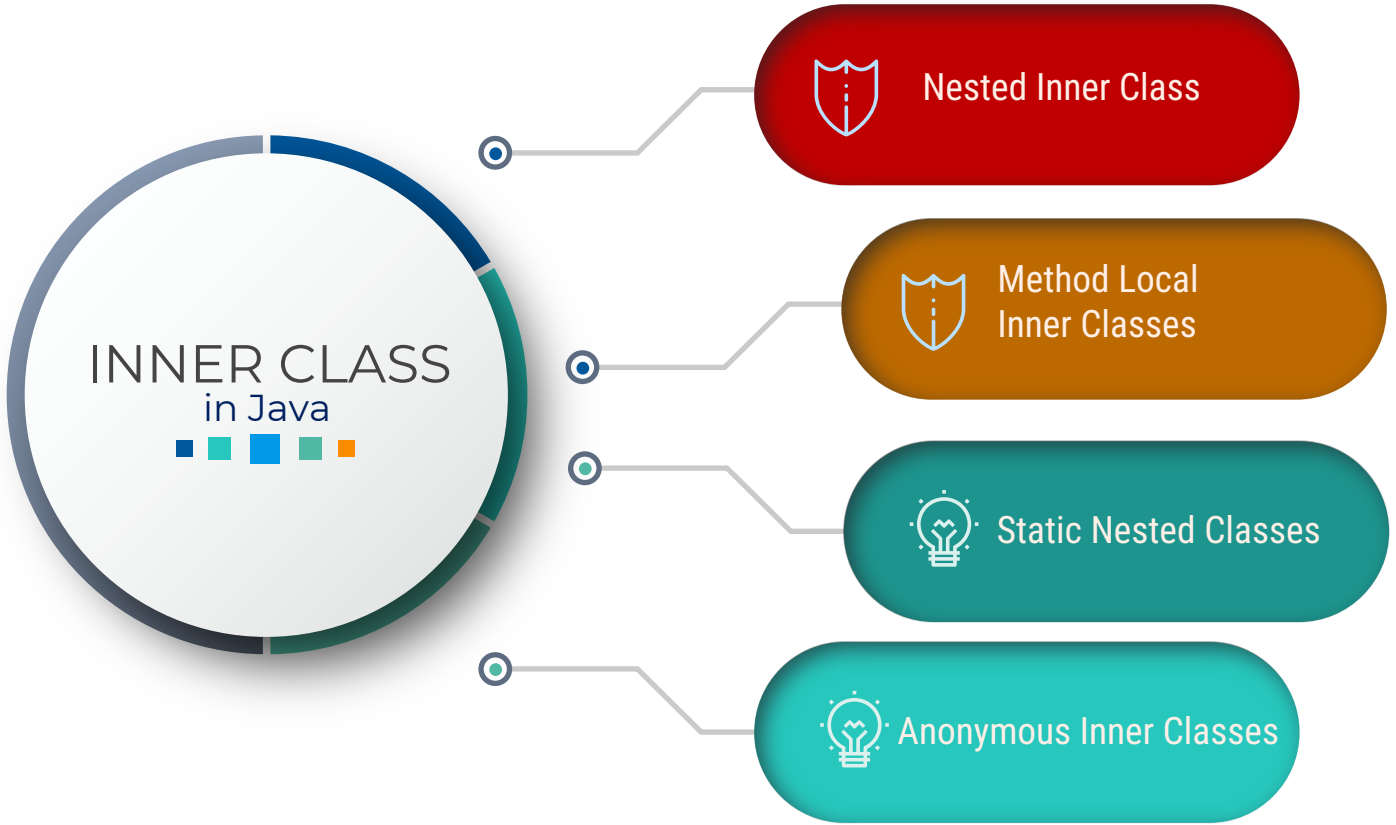
Tim Pengajar Pemrograman IPB University

Inner Class

Nested Class

Class yang dibuat di dalam suatu class atau interface.

Inner class:
- Membuat program menjadi clean and readable
- Private method dapat diakses


INNER CLASS
in Java
Nested Inner Class
Method Local
Inner Classes
Static Nested Classes
Anonymous Inner Classes

[1] Nested Inner Class

Dapat mengakses private dari outer class. Berlaku juga untuk nested inner interface.

```
class Outer {
    class Inner {
        public void show() { System.out.println("Nested class method"); }
    }
}

public class Driver {
    public static void main(String[] args) {
        Outer.Inner obj = new Outer().new Inner();
        obj.show();
    }
}
```

Output:

```
Nested class method
```

[1] Nested Inner Class :: Interface

```
class Outer {
    interface Inner { void show(); }
}

class Testing implements Outer.Inner {
    public void show() { System.out.println("Method of interface"); }
}

public class Main {
    public static void main(String[] args) {
    Outer.Inner obj;
    Testing t = new Testing();
    obj=t;
    obj.show();
    }
}
```

Output:

```
Method of interface
```

[2] Method Local Inner Classes

Inner class yang dideklarasikan di dalam method dari outer class.

```
class Outer {
    void outerMethod() {
        System.out.println("Outer Method");
        class Inner {
            void innerMethod() { System.out.println("Inner Method"); }
        }
        Inner y = new Inner();
        y.innerMethod();
    }
}

public class Main {
    public static void main(String[] args) {
        Outer obj = new Outer();
        obj.outerMethod();
    }
}
```

Output:

```
Outer Method
Inner Method
```

[2] Method Local Inner Classes

Inner class tidak dapat mengakses variabel lokal yang bukan final.

```
class Outer {
   void outerMethod() {
      final int x=98;
      System.out.println("Outer Method");
      class Inner {
         void innerMethod() {System.out.println("x = "+x); }
      }
      Inner y = new Inner();
      y.innerMethod();
   }
}
public class MethodLocalVariableDemo {
   public static void main(String[] args){
      Outer x = new Outer();
      x.outerMethod();
   }
}
```

Output:

```
Outer Method
x = 98
```

[3] Static Nested Classes

Sebenarnya secara teknis bukan inner class. Lebih mirip static member dari outer class.

```
class Outer {
    private static void outerMethod() {
        System.out.println("Outer Method");
    }
    static class Inner {
        public static void display() {
            System.out.println("Inner Class Method");
            outerMethod();
        }
    }
}
public class Main {
    public static void main(String args[]) {
        Outer.Inner.display();
    }
}
```

Output:

```
Inner Class Method
Outer Method
```

[4] Anonymous Inner Classes

Mendeklarasikan inner class tanpa nama.

```
class Demo {
    void show() { System.out.println("class Demo"); }
}

public class Main {
    static Demo d = new Demo() {
        void show() {
            super.show();
            System.out.println("class Main");
        }
    };

    public static void main(String[] args) {
        d.show();
    }
}
```

Output:

```
class Demo
class Main
```

[4] Anonymous Inner Classes :: Interface

Inner class tanpa nama mengimplementasikan interface.

```
interface Hello {
   void show();
}

public class Main {
   static Hello h = new Hello() {
      public void show() { System.out.println("Class Tanpa Nama"); }
   };

   public static void main(String[] args) {
      h.show();
   }
}
```

Output:

```
Class Tanpa Nama
```

Latihan Kelas

Silakan menggunakan konsep-konsep OOP yang pernah dipelajari

Suatu kalkulator akan memiliki nilai *currentValue* yang awalnya bernilai 0. Setelah itu, nilai tersebut akan dimodifikasi dengan operasi aritmatika yang diberikan terus menerus hingga kalkulator dimatikan. Pada soal ini, kalian diminta untuk membuat suatu interface yang mendefinisikan sifat suatu kalkulator yang dapat melakukan beberapa operasi matematika. Detail interface yang perlu kalian buat dapat dilihat pada tabel berikut:

- AritmatikaDasar - berisi fungsi-fungsi:
  - double tambah(double a, double b): menjumlahkan a dan b.
  - double kurang(double a, double b): mengurangi a dengan b.
  - double kali(double a, double, b): mengalikan a dan b;
  - double bagi(double a, double b): membagi a dengan b
- AritmatikaLanjut - berisi fungsi-fungsi:
  - double akarKuadrat(double a): mengembalikan nilai akar kuadrat dari a.
  - double pangkat(double a, double b): mengembalikan nilai a pangkat b
- KalkulatorSaintifik - merupakan kalkulator yang dapat melakukan operasi AritmatikaDasar dan AritmatikaLanjut. Juga memiliki fungsi void clear(): mengembalikan nilai currentValue pada kalkulator menjadi 0.

Latihan Kelas

Silakan menggunakan konsep-konsep OOP yang pernah dipelajari

Format Masukan

Setiap baris masukan merupakan salah satu dari tujuh kemungkinan berikut. Masukan akan diakhiri dengan simbol ~.

- `+ X` : menambah *currentValue* sebanyak X.
- `- X` : mengurangi *currentValue* sebanyak X.
- `* X` : mengalikan *currentValue* dengan X.
- `/ X` : membagi *currentValue* dengan X.
- `^ X` : memangkatkan *currentValue* sebanyak X.
- `C` : mengembalikan nilai *currentValue* menjadi 0.
- `~` : mematikan kalkulator, program berhenti

Format Keluaran

Nilai-nilai *currentValue* setiap kali suatu operasi selesai dilakukan. Cetak dengan dua angka di belakang koma dan akhiri dengan newline.

Contoh Masukan

```
+ 10
- 5
* 2
/ 5
^ 2
# 2
C
+ 5
~
```

Contoh Keluaran

```
10.00
5.00
10.00
2.00
4.00
2.00
0.00
5.00
```