PEMROGRAMAN BERORIENTASI OBJEK

Pertemuan 4
Implementasi Access Modifier

Konsep PBO

- Pada PBO membuat semua komponen program seolah-olah adalah semua objek.
- Sebuah objek selalu memiliki identitas (*attribute*) dan juga perilaku (*behavior*) atau kemampuan untuk melakukan tugas tertentu.
- Konsep PBO pada python memiliki tujuan untuk menciptakan potongan-potongan kode yang bersifat *reusable* dan tidak *redundan*.
- Terdapat 3 konsep utama PBO pada python, yaitu:
  - Encapsulation
  - Inheritance
  - Polymorphism

Encapsulation

- Konsep enkapsulasi di dalam PBO mengijinkan suatu class untuk mengatur visibilitas dari atribut-atributnya.
- Manfaat enkapsulasi:
  a) Menyembunyikan kompleksitas internal dari sebuah class/object, sehingga yang terlihat dari luar hanya attibutes/methods yang relevan
  b) Menjaga perubahan langsung pada attribute yang dimiliki oleh object, ini dapat berguna saat nilai baru sebuah attribute perlu divalidasi terlebih dahulu
  c) Menjaga agar method dengan hak akses public pada class tidak perlu berubah saat method internal class bersangkutan berubah, sehingga kode program di tempat lain yang menggunakan class tidak perlu berubah.

Jenis Hak Akses

Pada dasarnya terdapat 3 jenis hak akses:

1. *Public*, dapat diakses oleh class lain
2. *Private*, tidak dapat diakses oleh class lain, bahkan oleh turunan dari class tersebut
3. *Protected*, tidak dapat diakses di class lain, tetapi dapat diakses oleh turunan dari class tersebut.

Penulisan Hak Akses

Berikut ini penulisan hak akses dalam python:

1. **namavariabel** #pendeklarasian variabel public
2. **__namavariabel** #pendeklarasian variabel private
3. **_namavariabel** #pendeklarasian variabel protected

Contoh Hak Akses

Contoh program dengan hak akses public:

```
1  class Segitiga:
2      def __init__(self, alas, tinggi):
3          self.alas = alas
4          self.tinggi = tinggi
5          self.luas = 0.5 * alas * tinggi
```

```
1  segitiga_besar = Segitiga(100,80)
2
3  #akses variabel public: alas, tinggi, dan luas dari luar kelas Segitiga
4  print("alas: ",segitiga_besar.alas)
5  print("tinggi: ",segitiga_besar.tinggi)
6  print("luas: ",segitiga_besar.luas)
```

Output:

```
alas:  100
tinggi:  80
luas:  4000.0
```

Contoh Hak Akses

Contoh program dengan hak akses protected:

```
1  class Utama:
2      def __init__(self):
3          self._a = 2
4
5  class Turunan(Utama):
6      def __init__(self):
7          #Memanggil konstruktor pada kelas Utama
8          Utama.__init__(self)
9          print("Memanggil variabel protected pada class Utama: ",self._a)
10
11         # Modify the protected variable:
12         self._a = 3
13         print("Memanggil variabel protected yang dimodifikasi diluar class: ",self._a)
14
15
16 objek1 = Turunan()
17 objek2 = Utama()
18
19 #Memanggil variabel protected
20 print("Mengakses variabel protected dari objek1: ", objek1._a)
21 print("Mengakses variabel protected dari objek2: ", objek2._a)
```

Output:

```
Memanggil variabel protected pada class Utama:  2
Memanggil variabel protected yang dimodifikasi diluar class:  3
Mengakses variabel protected dari objek1:  3
Mengakses variabel protected dari objek2:  2
```

Contoh Hak Akses

Contoh program dengan hak akses private:

```
1  class Hitung:
2      def __init__(self):
3          self. __angkaRahasia=0
4
5      def tampilkanHitung(self):
6          self. __angkaRahasia += 1
7          print (self. __angkaRahasia)
8
9  hitungan=Hitung()
10 hitungan.tampilkanHitung()
```

```
1
```

```
1  hitungan.tampilkanHitung()
```

```
2
```

```
1  hitungan._Hitung__angkaRahasia
```

```
2
```

```
1  #namaobjek._namakelasnamaatribut
```

Terima Kasih