Florence Regis Lake
105223006
Praktikum PBO

Pendahuluan Modul 5

1. Cakupan variabel (Variable Scope):
   a. Jelaskan perbedaan antara variabel lokal, instance, kelas (static), dan parameter dalam Java!
      → **Variabel lokal** adalah variabel yang hanya bisa bekerja dalam suatu blok tertentu, seperti variabel yang dideklarasikan dalam sebuah method atau constructor.

      Variabel instance adalah variabel yang dibuat pada sebuah class, tetapi berada diluar dari method dan terkait dengan object tertentu.

      Variabel kelas (static) adalah variabel yang dideklarasikan menggunakan sinntaks static di dalam sebuah class ttapi berada di luar method.

      Variabel Parameter merupakan variabel yang dideklarasikan pada sebuah method atau constructor, lebih tepatnya pada tanda kurung, dan variabel jenis ini digunakan untuk menerima nilai saat method atau constructor tersebut dipanggil.
   b. Mengapa variabel static memiliki nilai yang sama untuk semua objek dari kelas yang sama?
      → Karena, variabel static merupakan variabel pada sebuah class, dan bukan variabel dari object individu dan hanya ada satu salinan variabel ststic pada memori ststis sehingga semua object berbagi nilai yang sama dan nilainya tetap konsister untuk seluruh object tersebut.
2. Java packages:
   a. Apa tujuan utama penggunaan package dalam Java?
      → Package digunakan untuk mengorganisir kode, mencegah konflik nama class, mengontrol aksesibilitas, mendukung penggunaan ulang kode, dan meningkatkan modularitas dalam proyek Java.
   b. Jelaskan perbedaan antara built-in package dan user-defined package dalam Java!
      → Built-in package adalah package yang sudah disediakan oleh Java Development Kit (JDK) dan dapat langsung digunakan dalam program. Sedangkan, user-defined

package adalah package yang dibuat sendiri oleh programmer untuk mengorganisir kode sesuai dengan kebutuhan proyek.

3. Access Modifiers:
   a. Sebutkan dan jelaskan empat jenis access modifier dalam Java!
      → **Public**, merupakan access modifier yang memungkinkan pengaksesan dimana saja, baik dari class maupun package lain.

      Protected, memungkinkan akses dari class dalam package yang sama dan juga dari subclass yang berada di package berbeda.

      Default, modifier ini tidak perlu ditulis secara langsung, dan hanya memungkinkan akses dari class dalam package yang sama.

      Private, membatasi akses hanya dalam class yang sama, sehingga class lain, bahkan yang berada dalam package yang sama, tidak dapat mengaksesnya.
   b. Mengapa access modifier private sering digunakan dalam konsep enkapsulasi?
      → Enkapsulasi adalah prinsip OOP yang bertujuan untuk melindungi data dengan membatasi akses langsung dari luar class. Oleh karena itu, access modifier private sering digunakan dalam konsep enkapsulasi karena memungkinkan kontrol penuh terhadap data dalam suatu class, seperti dapat meningkatkan keamanan data dengan metode getter dan setter.
4. Enkapsulasi:
   a. Apa yang dimaksud dengan enkapsulasi dalam pemrograman berorientasi objek?
      → Enkapsulasi adalah prinsip OOP yang bertujuan untuk melindungi data dengan membatasi akses langsung dari luar class. Dengan enkapsulasi, data dalam suatu objek hanya dapat diakses dan dimodifikasi melalui metode tertentu, sehingga meningkatkan keamanan dan keteraturan kode.
   b. Mengapa penggunaan getter dan setter penting dalam implementasi enkapsulasi?
      → Penggunaan getter dan setter sangat penting dalam implementasi enkapsulasi karena memungkinkan kontrol akses yang lebih aman, memastikan integritas data, serta membuat kode lebih fleksibel dan mudah dikelola.
5. Array of object:
   a. Apa perbedaan antara array biasa dan array of object dalam Java?

→ Array biasa dan array of object memiliki perbedaan utama dalam cara penyimpanannya dan elemen yang dikandungnya, array biasa digunakan untuk menyimpan nilai dengan tipe data primitif seperti int, double, atau char. Sedangkan array of object digunakan untuk menyimpan sekumpulan objek dari suatu class.

b. Sebutkan dan jelaskan dua cara untuk menginisialisasi array of object dalam Java!

→ Menggunakan constructor untuk membuat array terlebih dahulu, kemudian mengisi objek satu per satu, yang lebih fleksibel untuk perubahan atau pengisian data bertahap. Contohnya:

```
Mahasiswa[] dataMhs = new Mahasiswa[2];
dataMhs[0] = new Mahasiswa("Sofia Elfrida", 107424002);
dataMhs[1] = new Mahasiswa("Euphoric Loss", 105223006);
```

Menggunakan array literal untuk langsung mengisi objek dalam satu pernyataan, yang lebih praktis dan ringkas jika jumlah elemen sudah tetap. Contohnya:

```
Mahasiswa[] dataMhs = {
        new Mahasiswa("Ecaa"),
        new Mahasiswa("Euph"),
        new Mahasiswa("Florence")
};
```