Modul 1 : Class & Object

1.1 Tujuan

Setelah mengikuti praktikum ini mahasiswa diharapkan dapat:

1. Memahami konsep class & object.
2. Dapat mengimplementasikan konsep class & object pada kasus sederhana.

1.2 Dasar Teori

1.2.1 Class & Object

Object adalah gambaran dari entity, baik dunia nyata atau konsep dengan batasan-batasan dan pengertian yang tepat. Objek-objek ini kemudian juga dapat berupa gabungan dari beberapa objek yang lebih kecil. Sebagai contoh, lihatlah sebuah mobil. Mobil adalah sebuah objek dalam kehidupan nyata. Namun mobil sendiri merupakan gabungan beberapa objek yang lebih kecil seperti roda ban, mesin, jok, dan lainnya. Mobil sebagai objek yang merupakan gabungan dari objek yang lebih kecil dibentuk dengan membentuk hubungan antara objek-objek penyusunnya.

Kelas mendeskripsikan suatu objek. Dalam bahasa biologi, dapat dikatakan bahwa kelas adalah species, sedangkan objek merupakan individu. Kelas merupakan sebuah "blueprint" atau cetak biru yang sama untuk objek yang dibentuk dengan nilai yang berbeda-beda.

Kelas memiliki variabel yang disebut sebagai ***attribute*** dan *subroutine (a set of instructions designed to perform a frequently used operation*) yang biasa disebut ***method***. Dalam sudut pandang pemrograman, kelas digunakan untuk menciptakan suatu obyek. Atau dengan kata lain, kelas merupakan pembuat objek. Pada class terdapat suatu *access modifier.* Hal ini berguna untuk menentukan tipe hak akses bagi sebuah attribute dan method:

Tabel 1: Daftar Access Modifier

Modifier	**Class dan Interface**	**Method dan Variabel**
Default (tak ada modifier) **friendly**	Tampak di Paketnya	Diwarisi oleh subclassnya di paket yang sama dengan classnya. Dapat diakses oleh method method di class-class yang sepaket.
public	Tampak di manapun	Diwarisi oleh semua subclassnya. Dapat diakses dimanapun.
protected	Tidak dapat diterapkan	Diwarisi oleh semua subclassnya. Dapat diakses oleh method-method di class- class yang sepaket

private	Tidak dapat diterapkan	Tidak diwarisi oleh subclassnya Tidak dapat diakses oleh class lain

1.2.2 Method

Method dikenal juga sebagai suatu *function* dan *procedure*. Dalam OOP, method digunakan untuk memodularisasi program melalui pemisahan tugas dalam suatu class. Pemanggilan *method* menspesifikasikan nama *method* dan menyediakan informasi (parameter) yang diperlukan untuk melaksanakan tugasnya.

1.2.3 Keyword this

Di dalam Java terdapat suatu besaran referensi khusus yang disebut this, yang digunakan di dalam method yang dirujuk untuk objek yang sedang berlaku. Nilai this merujuk pada objek di mana method yang sedang berjalan dipanggil.

1.2.4 Konstruktor

Constructor atau konstruktor digunakan untuk melakukan inisialisasi variable-variabel instan class serta melakukan persiapan-persiapan yang diperlukan oleh suatu objek untuk dapat beroperasi dengan baik.

Format umum pendeklarasian dan pendefinisian *constructor* adalah :

a. Nama *constructor* sama dengan nama *class*.

b. Sebelum itu dapat diberi *access modifier* untuk mengatur *visibility constructor*.

Dalam suatu Class dapat lebih dari satu *constructor*, masing-masing harus mempunyai parameter yang berbeda sebagai penandanya. Hal seperti ini disebut ***overloading*** terhadap *constructor*.

1.2.5 Enkapsulasi

Enkapsulasi merupakan proses pemaketan objek beserta methodnya untuk menyembunyikan rincian implementasi dari pemakai/objek lainnya. Inti dari enkapsulasi atau pengkapsulan adalah ketidaktahuan apa yang ada dalam suatu objek dan bagaimana pengimplementasiannya. Yang dibutuhkan hanyalah apa kegunaan, bagaimana cara memakainya dan apa yang akan terjadi.

Dengan enkapsulasi, maka programmer tidak dapat mengakses suatu atribut yang dimiliki oleh suatu *class*. Kemampuan ini ditujukan untuk mendapatkan desain suatu *software* yang baik dan untuk keamanan *software* itu sendiri. Segala yang tidak perlu diketahui oleh yang lain, tidak perlu di*publish*.

Salah satu implementasi dari enkapsulasi adalah adanya *setter* dan *getter* untuk suatu atribut dalam suatu kelas. Jika pada suatu kelas terdapat atribut a dan b, maka terdapat method setA-getA dan setB-getB. Bentuk lain dari enkapsulasi adalah memasukkan nilai atribut dengan menggunakan konstruktor.

1.3 Latihan Praktikum

1. Buatlah sebuah program untuk menentukan apakah sebuah bilangan termasuk ke dalam salah satu kriteria berikut:

a. Positif Genap
b. Negatif Genap
c. Positif Ganjil
d. Negatif Ganjil

Gunakan prinsip class & object dengan sebuah konstruktor untuk mengeset bilangan.

2. Buatlah program untuk meng-generate deret dengan ketentuan terdapat suatu:

a. Nilai awal
b. Beda
c. Jumlah kemunculan angka pada deret

Program akan menghitung nilai rataan dari deret tersebut.

Contoh:

Masukkan jumlah kemunculan deret: 4

Deret: 2 5 8 11

Gunakan prinsip class & object dengan sebuah konstruktor untuk mengeset hal yang diketahui untuk menghasilkan deret.

3. Conan merupakan anak SD penggemar novel detektif. Ia ingin membuat sebuah program untuk mendata setiap novel detektif yang ia miliki. Setiap novel memiliki judul, nama pengarang dan tahun terbit. Tapi Conan ingin mengetahui isi dari setiap novel, sehingga ia tahu deskripsi novel tersebut.

Conan juga menginginkan informasi harga beli tercantum di program. Dikarenakan sewaktu-waktu ia ingin menjual kembali novelnya, terdapat mekanisme untuk menghitung harga jual novel. Rumus yang ia gunakan adalah “harga jual = harga beli – 20% * harga beli”.

Conan menginginkan programnya dibuat menggunakan konsep OOP, menggunakan bahasa Java dengan terdapat konsep enkapsulasi di dalamnya.

Bantulah Conan untuk merancang program yang akan ia buat. Tentukanlah atribut dan method terlibat, gambarkan class diagramnya. Buatlah 3 objek berdasarkan class tersebut (data bebas), dan tampilkan informasi dari setiap buku tersebut. Lalu, tampilkan harga total beli buku serta harga total buku jika dijual.