BAB 3
CLASS MEMBER : VARIABEL DAN METHOD

Dalam pemrograman berorientasi object, setiap object memiliki propertis dan *behaviour* atau perilaku yang menunjukkan karakter dari object tersebut. Selain itu, setiap object juga mampu berkomunikasi satu dengan yang lainnya melalui sebuah method dengan cara mengirimkan pesan tertentu. setiap pesan yang dikirim oleh suatu object akan diterima dan dikenali oleh object lain jika pesan tersebut memiliki tipe data yang dikenali oleh object lain.

3.1. VARIABEL

Variabel atau disebut juga field merupakan member dari class dimana setiap variabel harus memiliki tipe data tertentu. Ketika kita membuat suatu variabel maka akan ada memori komputer yang disediakan untuk menyimpan variabel tersebut. Tipe data di dalam pemrograman java bisa berupa tipe data primitif dan tipe data reference atau tipe data object. Tipe data primitif artinya tipe data ini sudah didefinisikan oleh bahasa pemrograman dengan kata kunci (*keyword*) khusus. Sedangkan tipe data reference merupakan tipe data yang merujuk pada suatu class object. Tipe data primitif diperlihatkan pada Tabel 3.1 Penulisan tipe data pada variabel harus sesuai dengan syntax pemrograman java, seperti contoh berikut ini.

```
tipe data namaVariabel = value;
tipe data namaVariabel;
namaVariabel = value;
```

Tabel 3.1. Tipe data primitive

Data Type	Default Value	Default Size
int	0	4 byte
float	0.0f	5 byte
long	0L	8 byte
double	0.0d	8 byte
boolean	false	1 bit

char	‘u0000’	2 byte
byte	0	1 byte
short	0	2 byte

Contoh :

```
int a, b, c, d = 10;
byte e = 22;
double pi = 3.14159;
char f = 'f';
```

Ada beberapa jenis variabel yang biasa digunakan dalam pemrograman berorintasi object yaitu *local variabel*, *instance variabel* dan *class/static variabel*. Ketiga jenis variabel tersebut memiliki karakteristik yang berbeda-beda.

3.1.1. LOCAL VARIABLE

Local variable / Variabel lokal merupakan variabel yang dideklarasikan di dalam sebuah method, konstruktor ataupun dalam suatu blok program. Variabel jenis ini akan diciptakan ketika method dijalankan dan akan dihapus dari memori komputer ketika method tersebut dihentikan. Sehingga lokal variabel hanya bisa digunakan oleh method yang mendeklarasikannya dan blok atau fungsi lain tidak bisa menggunakan variabel lokal dari method lain. Hal yang perlu diperhatikan adalah lokal variabel tidak memiliki nilai default sehingga setiap variabel harus di deklarasikan nilai awalnya.

Program 1. adalah contoh penerapan variabel lokal terdapat pada sebuah program. Amati hasilnya.

```
1  public class LocalVariable{
2       public void hitungUsia() {
3            int usia = 0;    // local variable
4            int tahunSekarang = 2019;
5            int tahunLahir = 1993;
6
7            usia = tahunSekarang - tahunLahir;
8
9            System.out.println("Usia saya : " + usia);
10      }
}
```

Program 1. Penerapan variabel lokal

LATIHAN 1

1. Buatlah method baru untuk menghitung berat badan dengan nama `void beratBadan()`, di dalam method tersebut buatlah variabel lokal `beratLahir`. Isikan nilai awal pada berat lahir kemudian hitunglah berat badan dengan rumus **beratBadan = beratLahir + (~~umur~~usia/2)**
2. Bisakah nilai dari variabel ~~umur~~usia dipanggil dari method void beratBadan()? berilah alasannya!

3.1.2. INSTANCE VARIABLE

Instance variabel atau bisa kita sebut dengan variabel global merupakan variabel milik dari suatu class dan berada diluar blok atau method tertentu. Variabel jenis ini bisa diakses dari semua method yang menjadi member dari class, sehingga semua method bisa memanipulasi nilai dari variabel global tersebut. Berbeda dengan variabel lokal, variabel global bisa memiliki akses modifier. Program 2. adalah contoh penerapan variabel global.

```
1    public class InstanceVariable{
2       int nilai; // instance variable
3
4       void firstMethod() {
5            // detail
6       }
7
8       void secondMethod() {
9            // detail
10      }
11   }
```

Program 2. Penerapan variabel global

LATIHAN 2

1. Modifikasi class `LocalVariable` pada Program 1., dengan menambahkan satu variabel global untuk menampung nilai dari ~~umur~~usia.
2. Gunakan nilai ~~umur~~usia untuk menyelesaikan permasalahan pada method menghitung berat badan yang ada di Latihan 1.

3. Analisa hasilnya dan bandingkan dengan hasil percobaan sebelumnya!

3.1.3. STATIC VARIABLE

Variabel statis merupakan variabel yang disimpan di dalam memori statis dan hanya memiliki satu nilai di mana nilai tersebut akan dipakai oleh semua object yang dibuat tanpa ada batasan jumlah object. Variabel statis biasanya digunakan untuk membuat nilai konstanta. Variabel statis juga bisa diakses oleh semua method yang ada pada suatu class dengan cara memanggil nama dari class dan variabelnya seperti `ClassName.VariableName`. Tulis dan jalankan kode Program 3 di bawah ini:

```
1   public class StaticVariable{
2      public static char akreditasi;
3      public static final String jurusan = "Informatika";
4
5      void firstMethod() {
6        System.out.println(jurusan);
7      }
8
9      void secondMathod() {
10       System.out.println("Akreditasi : " + akreditasi );
11     }
12  }
```

Program 3. Penerapan variabel statis

Variabel statis tidak bisa dijadikan sebagai variabel lokal. Sehingga untuk membuatnya, static modifier harus dituliskan pada variable global. Selain itu, untuk mengakses variabel statis kita tidak perlu membuat object dari class yang memiliki variabel statis. Perhatikan Program 4. dibawah ini.

```
1    public class StaticVariableAccess{
2
2       public static void main(String[] args) {
3
4             StaticVariable.akreditasi = 'B';
5             StaticVariable sv = new StaticVariable();
6             sv.firstMethod();
7
8             sv.secondMathod();
9
9       }
10
10  }
```

Program 4. Contoh pengaksesan variabel statis

Program 4. pada baris ke 3 adalah cara mengakses variabel akreditasi dari class `StaticVariable` tanpa perlu membuat object dari class tersebut.

3.2. METHOD

Method merupakan blok kode program yang menggambarkan perilaku atau perintah suatu program. Method memiliki tiga kategori, yang pertama adalah method dengan tipe data tertentu yang memiliki nilai `return` dan yang kedua adalah method dengan kata kunci `void`. Ketiga, method juga bisa memiliki *parameter*. Parameter di dalam method sifatnya tidak wajib dan itu sangat tergantung pada kegunaan dari method tersebut. Akan tetapi setiap parameter di dalam method harus memiliki tipe data tertentu Secara umum bentuk sebuah method adalah seperti blok berikut.

```
TipeMethod NamaMathod (parameter) {
   /* code */
}
```

3.2.1. METHOD NON-VOID

Method jenis ini adalah method yang harus memiliki type data pada body method. Dengan demikian maka Method harus memiliki nilai `return` dimana nilai `return` akan digunakan oleh method lain. Nilai `return` yang dihasilkan oleh method ini memiliki tipe data yang sama dengan tipe data yang dituliskan sebelum nama method. Contoh :

```
public String nama() {
   return nama;
}

public double nilaiMax() {
   return 100;
}
```

Dari contoh di atas terlihat bahwa nilai `return` yang dihasilkan oleh method nama akan bertipe data `String` dan method nilai akan menghasilkan nilai `return` dengan tipe data `double`**.** Tulis ulang kode Program 5. di bawah ini!

```
1    public class NonVoidMethod{
2       public String getNama() {
3             return nama;
4       }
5
6       public String getNIM() {
7             return nim;
8       }
9    }
```

Program 5. Metode non-void

Cobalah untuk memodifikasi kode Program 5, dengan menambahkan dua variabel `nama` dan `nim`, kemudian buatlah fungsi `main`() yang bisa menampilkan nama dan nim Saudara. Tunjukkan hasilnya kepada Asisten Praktikum / Dosen Pengampu.

3.2.2. PARAMETER METHOD

Parameter pada method merupakan sebuah variabel lokal yang hanya bisa diakses oleh method tersebut. Sebuah method boleh memiliki parameter ataupun tidak bahkan bisa memiliki lebih dari satu parameter, tergantung dari kegunaan method yang kita buat. Karena parameter merupakan variabel lokal maka setiap parameter yang dituliskan dalam method harus memiliki tipe data tertentu. Program 6 merupakan contoh parameter pada method.

```
1    public class MethodParameter {
2       String nama;
3       public String setNama (String nama) {
4          return this.nama = nama;
5       }
6    }
```

Program 6. Parameter pada method

Berdasarkan Program 6 di atas, terlihat bahwa baris ke-2 merupakan sebuah variabel global yang bertipe data `String`, kemudian pada baris ke-3 sampai 4

merupakan sebuah blok method. Di dalam method tersebut terdapat parameter yang bertipe `String`. Ketika kita akan menggunakan atau mengirimkan nilai dari suatu parameter ke parameter lain, maka parameter tersebut harus memiliki tipe data yang sama. Baris ke-4 pada kode di atas akan mengembalikan nilai dari `beriNama` dan kemudian akan disimpan ke dalam variabel `nama`. Tambahkan kode berikut ini kedalam class `MethodParameter` untuk melihat proses kerja dari method `setNama()`.

```
1    public static void main (String[] args) {
2        MethodParameter mp = new MethodParameter();
3        mp.setNama("Luffy");
4        System.out.println(mp.nama);
5    }
```

Program 7. Method `main()` pada class `MethodParameter`

LATIHAN 3

Lengkapilah kode Program 8 di bawah ini dengan menambahkan method yang memiliki parameter kemudian tampilkan hasilnya! Buatlah minimal 5 Object pegawai dengan nama, nim dan gaji yang berbeda-beda!

```
1    public class Pegawai{
2        String nama;
3        int nip;
4        double gaji;
5    }
```

Program 8. Latihan membuat parameter pada method

3.2.3. METHOD VOID

Method void merupakan method yang tidak menghasilkan nilai kembalian atau return value. Method dengan tipe void juga bisa memiliki parameter ataupun tidak, tergantung dari tujuan penggunaannya. Tuliskan kode di bawah ini dan amati perbedaannya dengan method non-void.

```
1  public class VoidMethod{
2     int hour, minute, second;
3
4     public  void  duration(int  hour,int  minute,  int
5                            second) {
6        this.hour = hour;
7        this.minute = minute;
8        this.second = second;
9     }
10
11    public void info() {
12       System.out.println("Total Waktu \n" +
13             hour + " jam " + minute +
14             " menit " + second + " detik");
15    }
16
17    public static void main (String[] args) {
18       VoidMethod vm = new VoidMethod();
19       vm.duration(1, 30, 15);
20       vm.info();
21    }
22 }
```

Program 9. Method void

PEKERJAAN RUMAH

```
1  public class Nilai{
2     int nilaiUTS;
3     int nilaiUAS;
4     int nilaiTugas;
5  }
```

Program 10. Tugas implementasi method

1. Lengkapilah kode pada Program 10 di atas dengan menambahkan method void dan method return, yang mengembalikan nilai dari setiap parameter method void.
2. Ubahlah tipe data dari `int` ke `double` dan tambahkan satu variabel `double nilaiTotal`, kemudian hitung `nilaiTotal` dengan rumus berikut :

   nilaiTotal = (nilaiUTS + nilaiUAS + nilaiTugas)/3