

# Bab 6

---

## Array

---

### POKOK BAHASAN

- Deklarasi array
- Membuat array
- Mengakses array
- Mendeklarasikan dan membuat array
- Inisialisasi array
- Array multi dimensi
  - Deklarasi array multi dimensi
  - Membuat array multi dimensi
- Mengetahui total elemen array
- Merubah total elemen array
- Mengkopi elemen array
- Referensi array

### TUJUAN BELAJAR

Dengan praktikum ini mahasiswa diharapkan dapat:

- Membuat dan menggunakan array
- Berinteraksi dengan array multi dimensi
- Mengkopi elemen array
- Memahami konsep referensi array

## **Dasar Teori**

- Array adalah suatu kumpulan data pada suatu variabel.
- Cara mendeklarasikan suatu array adalah sebagai berikut:

```
tipe_array nama_array[];  
tipe_array[] nama_array;
```

Contoh : int nilai[];  
char[] huruf;

- Agar kita dapat memesan tempat di memori untuk menampung elemen-elemen array, kita perlu membuat array. Adapun caranya adalah dengan memakai *new* karena di dalam Java suatu array adalah dianggap suatu obyek. Format penulisannya adalah sebagai berikut:

```
nama_array = new tipe_array[total_elelen_array];
```

Contoh : int nilai[];  
nilai = new int[5];

- Untuk dapat mengakses elemen array dapat dilakukan dengan menyebutkan elemen ke berapa dari array yang akan diakses, seperti berikut ini:

```
nama_array[elelen_array]
```

- Kita juga dapat melakukan deklarasi dan pembuatan array hanya pada satu baris *statement*. Adapun format penulisannya adalah sebagai berikut:

```
tipe_array nama_array[] = new tipe_array[total_elelen_array];
```

Contoh : int nilai[] = new int[5];

- Inisialisasi array dapat dilakukan dengan format penulisan sebagai berikut:

```
tipe_array nama_array[] = {nilai_indeks_0, nilai_indeks_1, ..., nilai_indeks_n};
```

Contoh : int nilai[] = {70, 65, 85};

- Kita dapat membuat array multi dimensi dengan cara menambahkan tanda [] sebanyak dimensi yang ingin dibuat. Sebagai contoh adalah sebagai berikut:

```
int x[][] = new int[3][4];
```

Baris *statement* diatas berarti kita ingin membuat array berdimensi 2, dengan 3 elemen di dimensi ke-1 dan 4 elemen di dimensi ke-2.

- Untuk mengetahui panjang dari suatu array yang telah kita buat, kita dapat memakai properti *length*. Adapun format untuk menggunakan *length* adalah sebagai berikut:

*var\_array.length* → total elemen array pada dimensi 1

*var\_array[i].length* → total elemen array pada dimensi 2 untuk indeks ke-i pada dimensi 1

*var\_array[i][j].length* → total elemen array pada dimensi 3 untuk indeks ke-i pada dimensi 1 dan indeks ke-j pada dimensi 2  
dan seterusnya.

- Isi dari suatu array dapat kita kopi pada array yang lain dengan memanfaatkan method *arraycopy()* pada class *System*. Format penulisannya sebagai berikut :

```
System.arraycopy(array1, p1, array2, p2, n);
```

dimana : array1 = array asal/sumber pengkopian

array2 = array tujuan pengkopian

p1 = posisi indeks awal pengkopian pada array asal

p2 = posisi indeks awal pengkopian pada array tujuan

n = banyaknya elemen array yang akan dikopi

- Suatu array juga dapat me-refer (merujuk) ke array yang lain, dengan kata lain merujuk pada alamat memori yang sama. Sebagai contoh adalah program berikut ini:

```
int nilai[] = {10, 20, 30};  
int result[];  
result = nilai;
```

Di baris ketiga, kita meng-*assign* array nilai ke array result. Akibatnya, array result akan me-refer (merujuk) pada array nilai, sehingga kedua array tersebut merujuk alamat memori yang sama.

## Percobaan

### **Percobaan 1 : Mengakses elemen array**

```
public class Array1 {  
    public static void main(String args[]) {  
        int nilai[]=new int[3];  
        nilai[0]=70;  
        nilai[1]=80;  
        nilai[2]=65;  
  
        double ratarata=0.0;  
        for(int i=0; i<nilai.length; i++) ratarata+=nilai[i];  
        ratarata/=nilai.length;  
  
        System.out.println("Nilai rata-rata = " + ratarata);  
    }  
}
```

### **Percobaan 2 : Mengakses elemen array berdimensi 2**

```
import java.text.NumberFormat;  
  
public class Array2 {  
    public static void main(String args[]) {  
        NumberFormat nf=NumberFormat.getInstance();  
        nf.setMaximumFractionDigits(3);  
  
        int nilai[][]=new int[2][3];  
        nilai[0][0]=85;
```

```

nilai[0][1]=81;
nilai[0][2]=78;
nilai[1][0]=65;
nilai[1][1]=73;
nilai[1][2]=71;

String MK[]={"RPL", "PBO"};
double ratarataMK[]=new double[nilai.length];

for (int i=0; i<nilai.length; i++) {
    for (int j=0; j<nilai[0].length; j++) {
        ratarataMK[i]+=nilai[i][j];
    }
    ratarataMK[i]/=nilai[0].length;
}

System.out.println("Nilai Mata Kuliah\n");
System.out.println("MK\tMinggu1\tMinggu2\tMinggu3\tRata-Rata");

for (int i=0; i<nilai.length; i++) {
    System.out.print(MK[i] + "\t");
    for (int j=0; j<nilai[0].length; j++) {
        System.out.print(nilai[i][j] + "\t");
    }
    System.out.print(nf.format(ratarataMK[i])+"\n");
}
}
}

```

### Percobaan 3 : Mendapatkan informasi panjang elemen array multi dimensi

```

public class CariPanjangElemen {
    public static void main(String args[]) {
        int x[][][][]=new int[2][][][];
        x[0]=new int[1][][];
        x[0][0]=new int[2][];
        x[0][0][0]=new int[3];
        x[0][0][1]=new int[2];

        x[1]=new int[2][][];
        x[1][0]=new int[1][];
        x[1][0][0]=new int[2];
        x[1][1]=new int[2][];
        x[1][1][0]=new int[1];
        x[1][1][1]=new int[3];

        System.out.println(x.length);
        System.out.println(x[0].length);
        System.out.println(x[0][0].length);
        System.out.println(x[0][0][0].length);
        System.out.println(x[0][0][1].length);
    }
}

```

```
        System.out.println();
        System.out.println(x[1].length);
        System.out.println(x[1][0].length);
        System.out.println(x[1][0][0].length);
        System.out.println(x[1][1].length);
        System.out.println(x[1][1][0].length);
        System.out.println(x[1][1][1].length);
    }
}
```

#### Percobaan 4 : Menangkap daftar argumen

```
public class GetArguments {
    public static void main(String args[]) {
        System.out.println("Tanggal : " + args[0]);
        System.out.println("Bulan : " + args[1]);
        System.out.println("Tahun : " + args[2]);
    }
}
```

#### Percobaan 5 : Melakukan pengkopian array

```
public class CopyArray {
    public static void main(String args[]) {
        int[] array1 = { 7, 4, 8, 1, 4, 1, 4 };
        int[] array2 = new int[3];
        System.arraycopy(array1, 0, array2, 0, 3);

        System.out.print("Array1 : ");
        for (int i=0; i<array1.length; i++)
            System.out.print(array1[i] + " ");
        System.out.println();

        System.out.print("Array2 : ");
        for (int i=0; i<array2.length; i++)
            System.out.print(array2[i] + " ");
    }
}
```

### Latihan

#### Latihan 1 : Mencari nilai rata-rata mata kuliah dari daftar nilai siswa

Diketahui daftar nilai siswa sebagai berikut:

<b>NRP</b>	<b>Nama Mhs</b>	<b>RPL</b>	<b>BD</b>	<b>PBO</b>
1	Ahmad	81	90	62
2	Adang	50	83	87
3	Dani	89	55	65
4	Edi	77	70	92

Buatlah program untuk menampilkan laporan sebagai berikut:

NRP	Rata-rata
1	77.67
2	73.33
3	69.67
4	79.67

### **Latihan 2 : Menampilkan deret Fibonacci dengan array**

Deret fibonanci adalah deret dimana dimulai dengan dua angka, dimana bernilai 1 dan 1, kemudian deret ketiga ditentukan dari penjumlahan kedua angka tersebut, sedangkan deret keempat ditentukan dari dua angka sebelumnya begitu seterusnya. Sehingga didapatkan deret fibonanci sebagai berikut: 1 1 2 3 5 8 13 21 dan seterusnya. Buatlah program untuk menampilkan bilangan Fibonacci yang banyaknya sesuai dengan input dan harus menggunakan array.

Contoh tampilan:

```
Masukkan jumlah deretan Fibonacci? 8
1 1 2 3 5 8 13 21
```

```
Masukkan jumlah deretan Fibonacci? 10
1 1 2 3 5 8 13 21 34 55
```

### **Tugas**

#### **Mendeteksi bilangan prima**

Buatlah suatu program untuk mendeteksi suatu bilangan itu termasuk bilangan prima

atau bukan.

Contoh tampilan:

```
Masukkan bilangan? 8  
8 bukan termasuk bilangan prima
```

```
Masukkan bilangan? 11  
11 adalah bilangan prima
```

# Bab 8

---

## Dasar-dasar

### Pemrograman Berbasis Obyek

---

#### POKOK BAHASAN

- Information hiding
- Enkapsulasi
- Constructor
- Overloading constructor

#### TUJUAN BELAJAR

Dengan praktikum ini mahasiswa diharapkan dapat:

- Menerapkan konsep enkapsulasi pada class
- Mendeklarasikan suatu constructor

#### Dasar Teori

- Kita dapat menyembunyikan information dari suatu class sehingga anggota-anggota class tersebut tidak dapat diakses dari luar. Adapun caranya adalah cukup dengan memberikan akses kontrol private ketika mendeklarasikan suatu atribut atau method. Contoh:

```
private int nrp;
```

- Encapsulation (Enkapsulasi) adalah suatu cara untuk menyembunyikan implementasi detail dari suatu class. Enkapsulasi mempunyai dua hal mendasar, yaitu:
  - information hiding
  - menyediakan suatu perantara (method) untuk pengaksesan data

Contoh:

```
public class Siswa {
    private int nrp;

    public void setNrp(int n) {
        nrp=n;
    }
}
```

- Constructor (konstruktor) adalah suatu method yang pertama kali dijalankan pada saat pembuatan suatu obyek. Konstruktor mempunyai ciri yaitu:
  - mempunyai nama yang sama dengan nama class
  - tidak mempunyai return type (seperti void, int, double, dan lain-lain)

Contoh:

```
public class Siswa {
    private int nrp;
    private String nama;

    public Siswa(int n, String m) {
        nrp=n;
        nama=m;
    }
}
```

- Suatu class dapat mempunyai lebih dari 1 konstruktor dengan syarat daftar parameternya tidak boleh ada yang sama. Contoh:

```
public class Siswa {
    private int nrp;
    private String nama;

    public Siswa(String m) {
        nrp=0;
        nama="";
    }
}
```

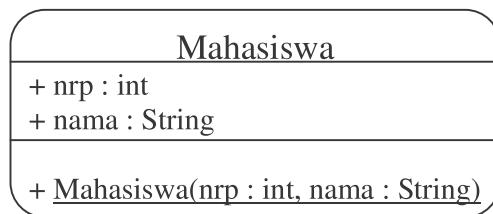
```

public Siswa(int n, String m) {
    nrp=n;
    nama=m;
}
}

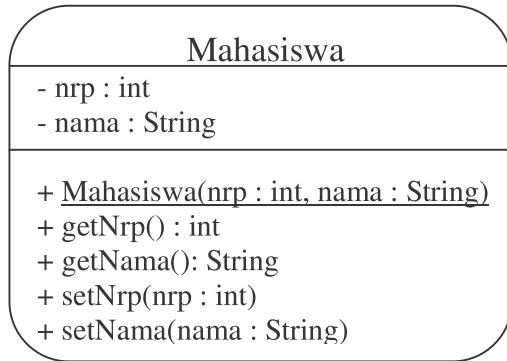
```

## Percobaan

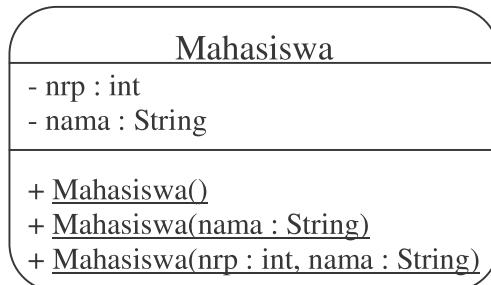
### Percobaan 1 : Melakukan enkapsulasi pada suatu class



Jika enkapsulasi dilakukan pada class diagram diatas, maka akan berubah menjadi:



### Percobaan 2 : Melakukan overloading constructor

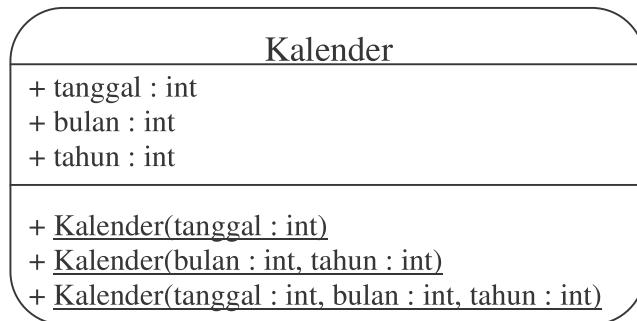


Dari class diagram tersebut, dapat diimplementasikan ke dalam program sebagai berikut:

```
public class Mahasiswa {  
    private int npn;  
    private String nama;  
  
    public Mahasiswa() {  
        npn=0;  
        nama="";  
    }  
  
    public Mahasiswa(String nama) {  
        npn=0;  
        this.nama=nama;  
    }  
  
    public Mahasiswa(int npn, String nama) {  
        this.npn=npn;  
        this.nama=nama;  
    }  
}
```

### Latihan

Mengimplementasikan UML class diagram dalam program untuk class Kalender



Dari class diagram diatas, desainlah suatu class yang memenuhi konsep enkapsulasi. Untuk nilai inisialisasi, dipakai 1-1-2000. Pakailah kata kunci *this* untuk mempersingkat pengkodean. Tulislah listing program berikut ini sebagai pengetesan.

```
public class TesLatihan {  
    public static String getTime(Kalender kal) {  
        String tmp;  
        tmp=kal.getTanggal() + "-" +  
            kal.getBulan() + "-" +  
            kal.getTahun();  
        return tmp;  
    }  
  
    public static void main(String args[]) {  
        Kalender kal=new Kalender(8);  
        System.out.println("Waktu awal : " + getTime(kal));  
        kal.setTanggal(9);  
        System.out.println("1 hari setelah waktu awal : " + getTime(kal));  
        kal=new Kalender(6,2003);  
        System.out.println("Waktu berubah : " + getTime(kal));  
        kal.setBulan(7);  
        System.out.println("1 bulan setelah itu : " + getTime(kal));  
        kal=new Kalender(20,10,2004);  
        System.out.println("Waktu berubah : " + getTime(kal));  
        kal.setTahun(2005);  
        System.out.println("1 tahun setelah itu : " + getTime(kal));  
    }  
}
```

Lakukan kompilasi pada program diatas dan jalankan. Jika tampilan di layar tampak seperti dibawah ini, maka program anda sudah benar. Jika tidak sama, benahi kembali program anda.

Waktu awal : 8-1-2000 1 hari setelah waktu awal : 9-1-2000 Waktu berubah : 1-6-2003 1 bulan setelah itu : 1-7-2003 Waktu berubah : 20-10-2004 1 tahun setelah itu : 20-10-2005
---