Pertemuan V

ARRAY

5.1. Pengenalan Array

Pada pertemuan sebelumnya kita telah membahas cara mendeklarasikan berbagai macam variabel dengan menggunakan berbagai tipe data. Dalam mendeklarasikan variabel, kita sering menggunakan sebuah tipe data beserta nama variabel atau *identifier* yang unik. Jika kita ingin menggunakan variabel tersebut, kita akan memanggil dengan nama variabel atau *identifier*-nya.

Sebagai contoh, kita memiliki tiga variabel dengan tipe data int yang memiliki *identifier* berbeda untuk tiap variabel.

```
int Angka1;
int Angka2;
int Angka3;

Angka1 = 1;
Angka2 = 2;
Angka3 = 3;
```

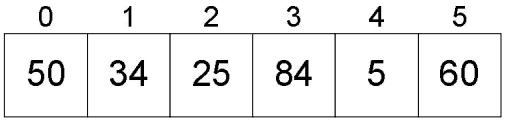
Seperti yang dapat kita lihat pada contoh diatas, kode tersebut akan sia-sia karena harus menginisialisasi dan menggunakan setiap variabel padahal sebenarnya variabel-variabel tersebut digunakan untuk tujuan yang sama. Apalagi jika data dalam variabel tersebut merupakan data yang berurutan, maka kita harus mengakses satu persatu nama variabelnya.

Pada bahasa pemrograman Java maupun di bahasa pemrograman yang lain, terdapat sebuah kemampuan untuk menggunakan satu variabel yang dapat menyimpan beberapa data dan memanipulasinya dengan lebih efektif. Tipe variabel inilah yang disebut sebagai array.

Array adalah suatu tipe yang dibentuk dari tipe data primitif untuk menyimpan sejumlah item yang bertipe sama. Array merupakan konsep yang penting dalam pemrograman, karena array memungkinkan untuk menyimpan data maupun referensi obyek dalam jumlah banyak dan terindeks. Array menggunakan indeks integer untuk menentukan urutan elemen-elemennya, dimana elemen pertamanya dimulai dari indeks 0, elemen kedua memiliki indeks 1, dan seterusnya.

Sebuah array akan menyimpan beberapa item data yang memiliki tipe data sama didalam sebuah blok memori yang berdekatan yang kemudian dibagai menjadi beberapa ruang. Array adalah sebuah variabel/sebuah lokasi tertentu yang memiliki satu nama sebagai *identifier*, namun *identifier* ini dapat menyimpan lebih dari sebuah nilai.

Penyimpanan data dalam array dapat digambarkan seperti pada gambar 3.7 dibawah ini :



Gambar 5.7. Blok penyimpanan dalam array bertipe integer

5.2. Mendeklarasikan Array

Array harus dideklarasikan seperti layaknya sebuah variabel. Pada saat mendeklarasikan array, kita harus membuat sebuah daftar dari tipe data, yang diikuti oleh sepasang tanda kurung siku "[]", lalu diikuti oleh nama *identifier*-nya. Sebagai contoh :

```
int[] ages;
```

atau kita dapat menempatkan sepasang tanda kurung siku [] sesudah nama *identifier*. Sebagai contoh :

```
int ages[];
```

Setelah mendeklarasikan array, kita harus membuat array dan menentukan berapa panjangnya dengan sebuah konstruktor. Proses ini di Java disebut sebagai *instantiation* (istilah dalam Java yang berarti membuat). Untuk meng-*instantiate* sebuah obyek, kita membutuhkan sebuah konstruktor *new*. Sebagai catatan bahwa ukuran dari array tidak dapat diubah setelah anda menginisialisasinya. Sebagai contoh :

```
//deklarasi
int ages[];
//instantiate obyek
ages = new int[100];
```

atau bisa juga ditulis dengan :

```
//deklarasi dan instantiate obyek
int ages[] = new int[100];
```

Pada contoh diatas, pendeklarasian tersebut akan memberitahukan kepada compiler Java, bahwa identifier *ages* akan digunakan sebagai nama array yang berisi data bertipe *integer*, dan dilanjutkan dengan membuat atau meng-*instantiate* sebuah array baru yang terdiri dari 100 elemen.

Selain menggunakan sebuah pernyataan *new* untuk meng-instantiate array, Anda juga dapat mendeklarasikan, membangun, kemudian memberikan sebuah nilai pada array sekaligus dalam sebuah pernyataan. Sebagai contoh,

```
//Membuat sebuah array yang terdiri dari penginisialisasian
//4variabel double bagi value {100,90,80,75}
double []grades = {100, 90, 80, 75};
```

5.3. Mengakses Elemen Array

Untuk mengakses sebuah elemen dalam array, atau mengakses sebagian dari array, kita harus menggunakan sebuah angka atau yang disebut sebagai indeks atau *subscript*.

Pada saat memasukkan nilai ke dalam array, sebuah nomor indeks atau *subscript* telah diberikan kepada tiap anggota array, sehingga program dan programmer dapat mengakses setiap nilai pada array apabila dibutuhkan. Nilai indeks selalu dalam tipe integer, dimulai dari angka nol dan dilanjutkan ke angka berikutnya sampai akhir array. Sebagai catatan bahwa indeks di dalam array dimulai dari 0 sampai dengan ukuran array dikurangi 1.

Sebagai contoh, pada array yang kita deklarasikan tadi, kita memberi nilai atau membaca nilai array sebagai berikut :

```
 //memberikan nilai 10 kepada elemen array pertama
ages[0] = 10;
//memberikan nilai 30 kepada elemen array indeks ke-90
ages[90] = 10;
//mencetak elemen array yang terakhir
System.out.print(ages[99]);
//mencetak elemen array indeks ke-90
System.out.print(ages[90]);
```

Perlu diperhatikan bahwa sekali array dideklarasikan dan dikonstruksi, nilai yang disimpan dalam setiap anggota array akan diinisialisasi sebagai nol. Oleh karena itu, apabila kita menggunakan tipe data seperti String, array tidak akan diinisalisasi menjadi string kosong "". Untuk itu Anda tetap harus membuat String array secara eksplisit.

Berikut ini adalah contoh kode untuk mencetak seluruh elemen didalam array. Dalam contoh ini digunakanlah pernyataan *for loop*, sehingga kode kita menjadi lebih pendek.

```
public class ArraySample{
   public static void main( String[] args ){
      int[] ages = new int[100];
      for( int i=0; i<100; i++ ){
         System.out.print( ages[i] );
      }
   }
}
```

5.4. Panjang Array

Untuk mengetahui berapa banyak elemen didalam sebuah array, kita dapat menggunakan atribut *length* dari array. Atribut ini akan mengembalikan ukuran dari array itu sendiri. Sebagai contoh :

```
arrayName.length
```

Pada contoh sebelumnya, kita dapat menuliskannya kembali seperti berikut ini :

```
public class ArraySample{
   public static void main( String[] args ){
      int[] ages = new int[100];
      for( int i=0; i<ages.length; i++ ){
         System.out.print( ages[i] );
      }
   }
}
```

5.5. Array Multidimensi

Array multidimensi diimplementasikan sebagai array yang terletak di dalam array. Array multidimensi dideklarasikan dengan menambahkan jumlah tanda kurung setelah nama array.

Sebagai contoh array multidimensi dapat dilihat pada kode program dibawah ini :

```
 // Elemen 512 x 128 dari integer array
 int[][] twoD = new int[512][128];
 // karakter array 8 x 16 x 24
 char[][][] threeD = new char[8][16][24];
 // String array 4 baris x 2 kolom
 String[][] dogs = {{"terry","brown"},{"Kristin","white"},{"toby",
"gray"},{"fido","black"}};
```

Untuk mengakses sebuah elemen didalam array multidimensi, sama saja dengan mengakses array satu dimensi. Misalnya saja, untuk mengakses elemen pertama dari baris pertama didalam array dogs, kita akan menulis :

```
 System.out.print( dogs[0][0] );
```

Kode diatas akan mencetak String “terry” di layar.

5.6. Latihan

Modifikasi program dibawah ini agar dapat menampilkan nilai maksimal, nilai minimal dan mengurutkan data dalam array.

```
 import java.io.BufferedReader;
 import java.io.InputStreamReader;
 import java.io.IOException;

 public class LatihanArray{
    public static void main( String[] args ){
      BufferedReader dataIn = new BufferedReader(new
InputStreamReader(System.in));
      String BacaInput = "";
      Byte JmlData = 0;
      Byte DataArray[];

      System.out.print("Jumlah data :");

      try{
        BacaInput = dataIn.readLine();
      }
      catch( IOException e ){
        System.out.println("Ada kesalahan !");
      }

      JmlData = new Byte (BacaInput);
      DataArray = new Byte[JmlData];

      //Membaca data dari keyboard
      System.out.println();
      for (Byte i = 0;i<JmlData;i++){
        System.out.print("DataArray["+i+"] = ");

        try{
          BacaInput = dataIn.readLine();
        }
        catch( IOException e ){
          System.out.println("Ada kesalahan !");
        }
        DataArray[i] = new Byte (BacaInput);
      }
```

```
      //Menampilkan data dari array
      System.out.println();
      for (Byte i = 0;i<JmlData;i++){
         System.out.println("DataArray["+i+"] = "+DataArray[i]);
      }
   }
}
```

Referensi:

1. Hariyanto, Bambang, (2007), *Esensi-esensiBahasaPemrograman Java*, Edisi 2, Informatika Bandung, November 2007.
2. Tim Pengembang JENI, JENI 1-6, Depdiknas, 2007
3. Utomo, EkoPriyo, (2009), *PanduanMudahMengenalBahasa Java*, YramaWidya, Juni 2009.