Object Oriented Programming

Agenda

1	Arrays
2	Enums
3	Strings

Intro

Intro

- **Sebuah program aplikasi bisa jadi membutuhkan sejumlah variabel.**
- **Deklarasi dan inisialisasi variabel-variabel dalam sebuah program membuat program tersebut menjadi kompleks.**
- **Selain itu, akan terasa lebih sulit untuk melakukan track pada variabel yang jumlahnya banyak.**
- **Untuk menanggulangi hal ini, kita bisa menggunakan variabel tunggal yang bisa menampung banyak data serupa.**
- **Inilah yang disebut dengan Array.**

Intro

- **Array merupakan sebuah kontainer/ variabel yang dapat menampung banyak data dengan tipe yang sama.**
- **Selain Array, kita juga sewaktu-waktu membutuhkan restriction action untuk user agar tidak bisa melihat data dari set value yang sudah tetap dalam pendefinisaannya.**
- **Untuk mencapai hal itu, kita akan menggunakan enum yang pada Java bisa digunakan untuk mendefinisikan konstan yang sudah fix.**
- **Selain itu, kita juga bias menyimpan rangkaian karakter seperti nama karyawan. Kita bisa menggunakan String untuk hal ini.**

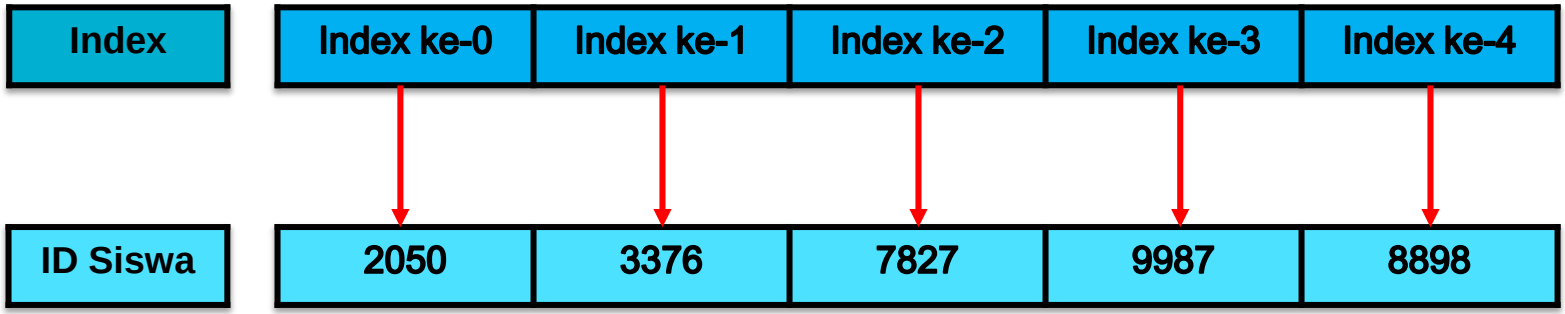
Array

Array

- Array merupakan kumpulan nilai yang disimpan disebuah variabel dan memiliki tipe data yang sama.
- Kita dapat mengakses nilai array dengan mengspesifikasikan nama array dan nomor index array tersebut.
- Elemen pertama dalam sebuah array disebut juga dengan index pertama.
- Index pertama ini diwakili oleh angka 0, bukan 1.
- Jadi, index di array dimulai dari angka 0.

Array

Contoh Ilustrasi:

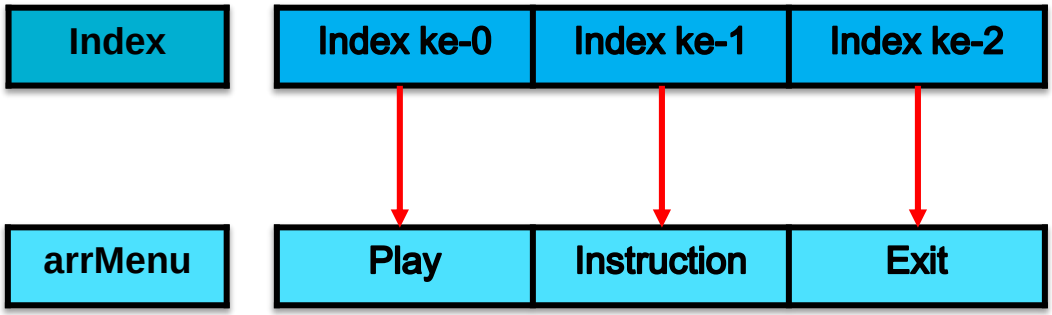


Array

- **Terdapat dua jenis array yang bisa digunakan, yakni Array satu dimensi dan Array multi-dimensi.**
  - **Array satu dimensi: merupakan koleksi elemen dengan nilai index tunggal.**
  - **Proses pembuatan dimulai dengan mendeklarasikan array, dilanjutkan dengan meng-assign nilai kedalam array.**
  - **Deklaras array satu dimendi menggunakan sintaks** arrayType arrayName[] = new arrayType[size];
  - **Contoh:** String arrMenu[] = new String[3];

Array

- **Untuk meng-assign nilai kedalam array, kita bisa menggunakan sintaks** arrMenu[0] = “Play”;
- **Atau:** String arrMenu[] = {“Play”, “Instruction”, “Exit”};
- **Ilustrasi array:**

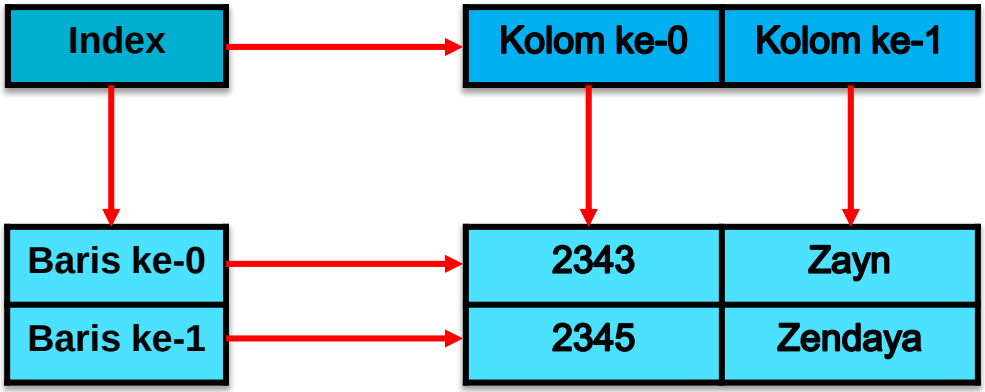


Array

- Array multi-dimensi merupakan array dari array, memiliki dimensi lebih dari satu.
- Tipe array multi-dimensi yang paling sering digunakan adalah dua dimensi dimana array memiliki baris dan kolom.
- Sebagai contoh, array yang akan menampung ID Siswa dan Nama Siswa.
- Array memiliki index baris dan index kolom.
- Index kolom maupun baris dimulai dari 0.

Array

- **Ilustrasi:**

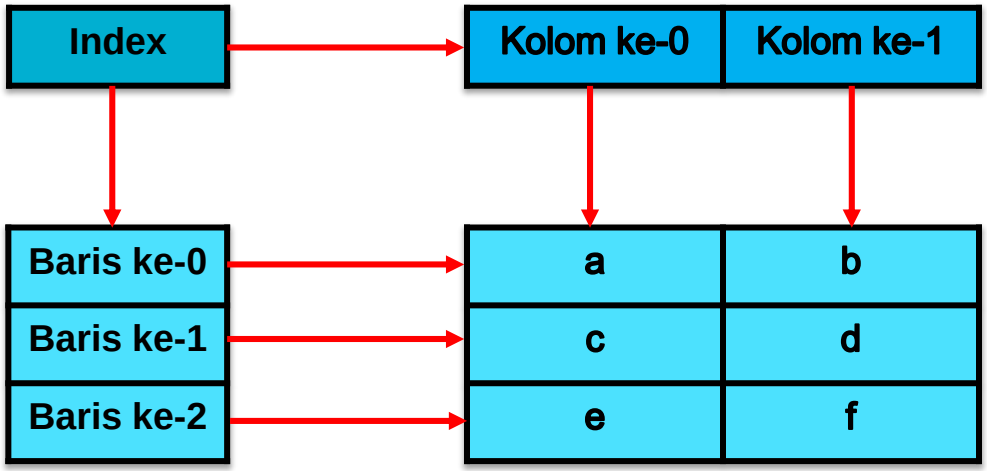


Array

- **Untuk membuat array dua dimensi, hal yang sama kita lakukan seperti pada pembuatan array satu dimensi, yakni dimulai dengan deklarasi lalu diikuti oleh assign nilai.**
- **Deklarasi array dua dimensi menggunakan sintaks** arrayType arrayName[][] = new arrayType[rowSize][columnSize];
- **Contoh:** String[][] words = new String[4][2];
- **Untuk assigning nilai, kita akan menggunakan sintaks** words[0][0] = “a”; words[0][1] = “b”;

Array

- Cara lain untuk assign nilai ke array dua dimensi adalah dengan cara String[][] words = new String[][]{{“a”, “b”}, {“c”, “d”}, {“e”, “f”}};
- Ilustrasi



Array

- **Untuk mengakses array, baik yang satu dimensi maupun dua dimensi, kita bisa menggunakan sintaks berikut:**
- **Satu dimensi:**

Cara 1:

```
System.out.println(arrMenu[0]);
```

- **Output:** Play

Cara 2:

```
for(int i=0; i<arrMenu.length; i++){
    System.out.println(arrMenu[i]);
  }
```

- **Output:**

```
Play
Instruction
Exit
```

Array

Cara 3:

```
for(String i : arrMenu){
        System.out.println(i);
    }
```

- **Output:**

```
Play
Instruction
Exit
```

Array

- **Dua dimensi:**

Cara 1:

```
System.out.println(words[0][0]);
```

- **Output:** a

Cara 2:

```
for(int i = 0; i <= 2; i++){
    for(int j = 0; j<2; j++){
        System.out.print(words[i][j]);
    }
}
```

- **Output:** abcdef

Array

Cara 3:

```
for(int i = 0; i < words.length; i++){
    for(int j = 0; j<words[i].length; j++){
        System.out.print(words[i][j]);
    }
}
```

- **Output:** abcdef

Cara 4:

```
for(String[] I : words){
    for(String j : i){
        System.out.println(j);
    }
}
```

- **Output:** abcdef

Enum

Enum

- **Enum adalah sebuah tipe data yang nilainya hanya terbatas dari pilihan nilai-nilai yang telah didefinisikan terlebih dahulu.**
- **Deklarasi enum bisa menggunakan sintaks berikut:**

  enum enum_name{constant1, constant2, …, constantN};
- **Contoh:**

  enum Buah{Mangga, Apel, Pisang};
- **Di Java, enum mirip dengan class, dimana enum dapat memiliki konstruktor, method dan instance variables. Namun, kita tidak bisa membuat instance dari enum dengan keyword "new".**

Enum

- **Konstruktor enum akan dipanggil ketika enum constant dibuat, constant ini berlaku sebagai objek, seperti contoh berikut:**

MacamBuah.java

Source | History

```
1   package ArrayEnumString.Enum;
2
3   class MacamBuah {
4       enum Buah {
5           Apel(10), Mangga(7), Pisang(8);
6           private int harga;
7
8           Buah(int h) {
9               harga = h;
10          }
11
12          int getHarga() {
13              return harga;
14          }
15      }
16  }
```

Enum

- **Ketika kita telah mendeklarasikan enum, kita dapat mengaksesnya dengan menggunakan nama enum tersebut atau referensinya.**
- **Cara pembuatan referensi enum sama dengan pembuatan variabel seperti berikut:**

  enum_name.enum_constant

  Atau

  enum_reference.enum_constant
- **Contoh:** Buah h = Buah.Apel;

Enum

- **Ketika kita telah mendeklarasikan enum, kita dapat mengaksesnya dengan menggunakan nama enum tersebut atau referensinya.**
- **Cara pembuatan referensi enum sama dengan pembuatan variabel seperti berikut:**

  enum_name.enum_constant

  Atau

  enum_reference.enum_constant

- **Contoh:** Buah h = Buah.Apel; **Pada contoh ini constant Apel kita akses dengan menggunakan nama enum kita yakni Buah. Selanjutnya, constant tersebut kita masukkan kedalam reference enum kita yakni "p".**

Enum

- **Untuk mengakses keseluruhan nilai yang kita simpan di enum, Java menyediakan sebuah fungsi bernama** values()**.**
- **Contoh:**

MacamBuah.java

Source | History

```
1  package ArrayEnumString.Enum;
2
3  enum Buah{Apel, Mangga, Pisang};
4  class MacamBuah{
5      public static void main(String[] args) {
6          for(Buah h:Buah.values())
7              System.out.println(h);
8      }
9  }
```

Enum

- **Selain itu, kita juga bisa mengakses fungsi yang ada di dalam enum dengan cara berikut:**

MacamBuah.java

Source | History

```
1  package ArrayEnumString.Enum;
2
3  enum Buah {
4      Apel(9000), Mangga(5000), Pisang(3000);
5      private int harga;
6
7      Buah(int h) {
8          harga = h;
9      }
10     int getHarga(){
11         return harga;
12     }
13 }
14
15 class MacamBuah {
16     public static void main(String[] args) {
17         Buah b = Buah.Apel;
18         System.out.println(b.getHarga());
19     }
20 }
```

String

String

- **Untuk memanipulasi string, kita bisa menggunakan beberapa class seperti String, StringBuffer dan StringBuilder yang disediakan oleh Java.**
- **Kita dapat menggunakan sintaks berikut untuk membuat objek String:**

  String nama = new String("Lucas");
- **Kita juga dapat membuat objek String dengan cara berikut:**

  String nama = "Lucas";
- **Pada kedua contoh diatas, kita mengisikan nama Lucas kedalam variabel String "nama".**

String

- **Terdapat beberapa fungsi yang bisa kita gunakan untuk memanipulasi class String, berikut yang paling umum digunakan:**
  - **int length(): Merupakan fungsi yang akan mengembalikan panjang objek String. Misal, “Lucas” memiliki panjang 5.**
  - **char charAt(int index): Akan mengembalikan karakter/ huruf yang ada di posisi index terntentu, index dihitung dari angka 0.**

String

- **void getChars(int srcBegin int srcEnd, char[] dst, int dstBegin):** Fungsi ini akan meng-copy karakter/ huruf dari sumber objek string kedalam karakter array. Karakter pertama yang di-copy akan berada di index **srcBegin,** karakter terakhir yang di-copy akan berada di index **srcEnd-1,** karakter di-copykan dimasukkan kedalam **dst,** dan karakter yang di-copy dimulai dari index ke-**destBegin.**
- **boolean equals(object obj): Digunakan untuk membandingkan objek string yang ada dengan objek string yang lain, lalu mengembalikan nilai TRUE/ FALSE.**

String

- **int compareTo(String str): Membandingkan string objek dengan string objek lain, jika kedua string sama, maka akan dikembalikan nilai 0, jika tidak akan dikembalikan selain 0.**
- **boolean startsWith(String prefix): Mengecek apakah sebuah string diawali oleh prefix tertentu atau tidak dan mengembalikan nilai TRUE/ FALSE.**
- **boolean endsWith(String suffix): Mengecek apakah sebuah string diakhiri oleh suffix tertentu atau tidak dan mengembalikan nilai TRUE/ FALSE.**

String

- int indexOf(int ch): Mengembalikan index dari kemunculan pertama karakter yang dicari.
- int lastIndexOf(int ch): Mengembalikan index dari kemunculan terakhir dari karakter yang dicari.
- String subString(int beginindex): Mengembalikan substring dari sebuah string.
- String concat(String str): Menyambungkan sebuah string ke belakang objek string lain.

String

- String replace(char oldChar, char newChar): Mengganti karakter tertentu dengan karakter baru yang kita tentukan.
- String toUpperCase(): Mengkonversi string ke bentuk uppercase.
- String toLowerCase(): Mengkonversi string ke bentuk lowercase.
- String trim(): menghilangkan spasi diawal dan akhir sebuah kalimat.
- char[] toCharArray(): Mengembalikan sebuah array yang berisikan string yang sudah di deklarasikan, ukuran array akan menyesuaikan dengan panjang string.

String

- **String valueOf(Object obj): Mengembalikan nilai string dari argument tertentu.**
- **boolean equalsIgnoreCase(String anotherString): Membandingkan satu string denga string lain tanpa mengecek case dari kedua string (mengabaikan huruf kapital dan huruf kecilnya, hanya mengecek apakah sama atau tidak).**

String

- **Selain beberapa fungsi tersebut, kita juga bisa menggunakan class StringBuilder dan StringBuffer.**
- **Kedua kelas ini merupakan kelas muteable.**
- **Kita bisa menggunakan sintaks berikut untuk melakukan inisialisasi**

  StringBuilder sl = new StringBuilder("Hello World");

- **Baik StringBuilder maupun StringBuffer sebenarnya sama, hanya saja StringBuilder tidak disinkronisasi dan StringBuffer disinkronisasi.**
- **Sederhananya, StringBuilder lebih cepat dibanding StringBuffer.**

String

- **Beberapa fungsi yang bisa kita gunakan dari class ini adalah:**
  - **StringBuilder append(String obj): Menambahkan sebuah argumen ke string builder.**
  - **StringBuilder delete(int start, int end): Menghapus karakter dari index tertentu ke index tertentu.**
  - **StringBuilder insert(int offset, String obj): Memasukkan argumen kedua ke sebuah string builder.**
  - **StringBuilder reverse(): Membalikan sebuah string.**

Thank You!

I Am Who I'm Believe I Am