PRAKTIKUM 16

TYPE ENUM

A. TUJUAN PEMBELAJARAN

1. Memahami mengenai konsep Enum
2. Memahami bentuk-bentuk penggunaan Enum
3. Memahami fungsi-fungsi yang terdapat di Enum

B. DASAR TEORI

Sebelum J2SE 5.0, solusi untuk menangani masalah sekumpulan nilai konstanta, dicontohkan seperti di bawah ini: (jenis credit card yang bisa diterima oleh aplikasi)

- public static final int VISA = 1;
- public static final int MASTER_CARD = 2;
- public static final int AMERICAN_EXPRESS = 3;

Permasalahan yang muncul adalah tidak ada yang mengikat ketiga nilai menjadi semacam set dan kita bisa memberikan nilai yang salah pada variabel-variabel tersebut. Kondisi seperti ini disebut *not type safe* (tipe yang tidak aman). Kondisi ini dapat diperbaiki dengan membuat tipe yang relatif aman (tipe safe) dengan mendefinisikan suatu class, yaitu class AllowedCreditCard. Class tersebut mendefinisikan :

– konstanta-konstanta di dalam kelas
– variabel untuk menyatakan state object dari class tersebut.
– Constructor private untuk mengeset state.

```
public class AllowedCreditCard {
    protected final String card ;
    public final static AllowedCreditCard VISA = new
AllowedCreditCard("VISA");
    public final static AllowedCreditCard MASTER_CARD = new
AllowedCreditCard("MASTER CARD");
    public final static AllowedCreditCard AMERICAN_EXPRESS = new
AllowedCreditCard("AMERICAN_EXPRESS");
    private AllowedCreditCard(String str){
        card = str;
```

```
    }
    public String getName(){
        return card ;
    }
}
```

Pada pendekatan ini, terdapat 3 state dari class AllowedCreditCard yang dinyatakan oleh tiga object yang dicreate dengan memberikan nilai yang berbeda pada variabel card. Karena konstruktor private, sehingga tidak bisa create object diluar class. Design seperti ini masih dianggap type safe. Tapi pada J2SE 5.0 terdapat solusi yang bagus dengan mengenalkan tipe baru yaitu enum.

Tipe data enum dikenalkan di J2SE 5.0 berguna untuk variabel yang berisi sekumpulan nilai. Cara mendefinisikan variabel enum:

– Mendefinisikan tipe enum dengan sekumpulan nilai.
– Mendefinisikan variabel yang menyimpan satu dari nilai-nilai tersebut.

Contoh 1 : Class AllowedCreditCard dirubah menjadi bentuk enum.

```
enum AllowedCreditCard {VISA, MASTER_CARD, AMERICAN_EXPRESS};
AllowedCreditCard visa = AllowedCreditCard.VISA;
```

Contoh 2 : Mendeklarasikan enum di luar class

```
enum CoffeeSize {BIG, HUGE, OVERWHELMING}

public class Coffee {
    CoffeeSize size ;
}

class CoffeeTest1{
    public static void main(String[] args) {
        Coffee drink = new Coffee();
        drink.size = CoffeeSize.BIG ;
    }
}
```

Contoh 3 : Mendeklarasikan enum di dalam class

```
public class Coffee2 {
    enum CoffeeSize {BIG, HUGE, OVERWHELMING}
    CoffeeSize size ;
}
class CoffeeTest2{
    public static void main(String[] args) {
        Coffee2 drink = new Coffee2();
```

```
        drink.size = Coffee2.CoffeeSize.BIG ;
    }
}
```

Contoh 4 : Tidak bisa mendeklarasikan enum di dalam method

```
//Tidak legal
public class CoffeeTest3 {
    public static void main(String[] args) {
        enum CoffeeSize {BIG, HUGE, OVERWHELMING}
        Coffee drink = new Coffee();
        drink.size = CoffeeSize.BIG ;
    }
}
```

Menentukan sendiri nilai konstanta dari enum.

- Cara termudah dengan memberikan nilai enum (misal : BIG, HUGE, dan OVERWHELMING). Nilai enum sebagai object yang masing-masing mempunyai nilai instance variabel sendiri-sendiri.
- Nilai ini diberikan pada saat enum diinisialisasi, dengan memberikan nilai pada konstruktor enum.

```
enum CoffeeSize2{
    BIG(8), HUGE(10), OVERWHELMING(16);
    CoffeeSize2(int ounces){
        this.ounces = ounces ;
    }
    private int ounces ;
    public int getOunces() {
        return ounces;
    }
}
```

Konstruktor enum dijalankan secara otomatis. Contoh BIG(8) menjalankan konstruktor CoffeSize yang menerima parameter berupa int, dengan nilai 8. Konstruktor pada enum bisa lebih dari satu.

Fungsi pada Enum

- public static Apple[] values()

  Mengembalikan array yang berisi konstanta dari tipe enum, urutan sesuai pada saat pendeklarasian enum.
- public static Apple valueOf(String name)

Mengembalikan konstanta enum sesuai dengan inputan dari parameter String

- public final int ordinal()

  Mengembalikan ordinal dari enum konstanta (dimulai dari 0)

- public final int compareTo(E o)

  Membandingkan object enum dengan object enum lainnya berdasarkan urutan.

  Mengembalikan nilai negatif (object enum 1 < object enum 2), 0 (object enum 1 = object enum 2) dan positif (object enum 1 > object enum 2).

```
enum Apple {
  A(10), B(9), C, D(15), E(8);

  private int price; // price of each apple

  // Constructor
  Apple(int p) { price = p; }

  // Overloaded constructor
  Apple() { price = -1; }

  int getPrice() { return price; }
}
```

C. TUGAS PENDAHULUAN

Buatlah review mengenai

- Definisi enum
- Merubah bentuk konstanta(terdapat lebih dari satu konstanta) menjadi bentuk enum
- Letak yang diperbolehkan dan tidak diperbolehkan untuk pendeklarasian enum

D. PERCOBAAN

Percobaan 1 : Class AllowedCreditCard dirubah menjadi bentuk enum.

```
public class AllowedCreditCard {
    protected final String card ;
    public final static AllowedCreditCard VISA = new
AllowedCreditCard("VISA");
    public final static AllowedCreditCard MASTER_CARD = new
AllowedCreditCard("MASTER CARD");
    public final static AllowedCreditCard AMERICAN_EXPRESS = new
AllowedCreditCard("AMERICAN_EXPRESS");
    private AllowedCreditCard(String str){
        card = str;
    }
    public String getName(){
        return card ;
```

```
    }
}
```

```
enum AllowedCreditCard {VISA, MASTER_CARD, AMERICAN_EXPRESS};
AllowedCreditCard visa = AllowedCreditCard.VISA;
```

Percobaan 2 : Mendeklarasikan enum di luar class

```
enum CoffeeSize {BIG, HUGE, OVERWHELMING}

public class Coffee {
    CoffeeSize size ;
}

class CoffeeTest1{
    public static void main(String[] args) {
        Coffee drink = new Coffee();
        drink.size = CoffeeSize.BIG ;
    }
}
```

Percobaan 3 : Mendeklarasikan enum di dalam class

```
public class Coffee2 {
    enum CoffeeSize {BIG, HUGE, OVERWHELMING}
    CoffeeSize size ;
}
class CoffeeTest2{
    public static void main(String[] args) {
        Coffee2 drink = new Coffee2();
        drink.size = Coffee2.CoffeeSize.BIG ;
    }
}
```

Percobaan 4 : Tidak bisa mendeklarasikan enum di dalam method

```
//Tidak legal
public class CoffeeTest3 {
    public static void main(String[] args) {
        enum CoffeeSize {BIG, HUGE, OVERWHELMING}
        Coffee drink = new Coffee();
        drink.size = CoffeeSize.BIG ;
    }
}
```

Percobaan 5 : Menentukan sendiri nilai konstanta dari enum

```
enum CoffeeSize2{
    BIG(8), HUGE(10), OVERWHELMING(16);
    CoffeeSize2(int ounces){
        this.ounces = ounces ;
```

```
    }
    private int ounces ;
    public int getOunces() {
        return ounces;
    }
}
```

Percobaan 6 : Enum dengan statement switch

```
enum OperatingSystems {
    windows, unix, linux, macintosh
}

public class EnumExample1 {
    public static void main(String args[])
    {
        OperatingSystems os;

        os = OperatingSystems.windows;
        switch(os) {
            case windows:
                System.out.println("You chose Windows!");
                break;
            case unix:
                System.out.println("You chose Unix!");
                break;
            case linux:
                System.out.println("You chose Linux!");
                break;
            case macintosh:
                System.out.println("You chose Macintosh!");
                break;
            default:
                System.out.println("I don't know your OS.");
                break;
        }
    }
}
```

Percobaan 7 : Fungsi pada Enum

```
enum Apple {
  A(10), B(9), C, D(15), E(8);

  private int price; // price of each apple

  // Constructor
  Apple(int p) { price = p; }

  // Overloaded constructor
  Apple() { price = -1; }

  int getPrice() { return price; }
}
```

Percobaan 8 : Melakukan enumerasi pada Enum

```
enum Media {

  book, music_cd, music_vinyl, movie_vhs, movie_dvd;

}
public class MediaFactory {

  public static void main(String[] args) {

    System.out.println(MediaFactory.getMedia("Book"));
  }
  public static Media getMedia(String s) {
    return Enum.valueOf(Media.class, s.toLowerCase());
  }
  public static Media getMedia(int n){
    return Media.values()[n];
  }
}
```

E. LATIHAN

1. Consider the following code fragment:

```
1. class EnumTest{
2.    enum Size{small, medium, large, Xlarge};
3.    public static void main(String [] args) {
4.       for( Size s : Size.values()) {
5.          if (s == Size.small)
6.                System.out.print("small ");
7.          else if (Size.medium.equals(s))
8.                System.out.println("medium ");
9.          else if (s == Size.large)
10.               System.out.println("large ");
11.         else if (s.equals("Xlarge "))
12.               System.out.println("Xlarge ");
13.         else if ( s == "Xlarge ");
14.               System.out.println("Xlarge ");
15.       }
16.    }
17. }
```

What is the result of this code?

A. small medium large Xlarge Xlarge

B. small medium large Xlarge

C. small medium large

D. Compiler error at line 11

E. Compiler error at line 13

F. Throws exception at runtime

2. Consider the following code fragment:

```
1. enum Colors {BLUE, GREEN, YELLOW, RED}
2. class Picture {
3.    public static void main(String [] args) {
4.          int x = 0;
5.          Colors c = Colors.GREEN;
6.          switch(c) {
7.            case BLUE:
8.                System.out.print(c);
9.            case GREEN:
10.               System.out.print(c);
11.           case YELLOW:
12.               System.out.print(c);
13.           default:
14.               System.out.print(" BlackWhite ");
15.               break;
16.           case RED:
17.               System.out.print(c);
18.         }
19.         System.out.println(" PicturePerfect");
20.    }
21. }
```

What is the result?

A. GREEN PicturePerfect

B. GREENGREEN PicturePerfect

C. GREENGREEN BlackWhite PicturePerfect

D. GREENYELLOW BlackWhite PicturePerfect

E. GREENGREEN BlackWhite Red PicturePerfect

F. Compilation fails at line 6.

G. Compilation fails at line 13.

3. Consider the following code fragment:

```
1. enum Colors {BLUE, GREEN, RED}
2. class Picture {
3.    public static void main(String [] args) {
```

```
4.       int x = 0;
5.        Colors c = Colors.GREEN;
6.       switch(c) {
7.          case BLUE:
8.            System.out.print(c);
9.          case GREEN:
10.           System.out.print(c);
11.         case YELLOW:
12.           System.out.print(c);
13.         default:
14.           System.out.print(" BlackWhite ");
15.           break;
16.         case RED:
17.           System.out.print(c);
18.      }
19.       System.out.println(" PicturePerfect");
20.   }
21. }
```

What is the result?

A. GREEN PicturePerfect

B. GREENYELLOW PicturePerfect

C. GREENGREEN BlackWhite PicturePerfect

D. GREEN BlackWhite Red PicturePerfect

E. Compilation fails.

F. An exception is thrown at runtime.

4. Consider the following code:

```
1. enum Villages {Pharwala, Gohawar, Phagwara, Goraya}
2. public class MyEnumTest {
3.    public static enum Colors{RED, BLUE, GREEN, YELLOW, ORANGE};
4.    private enum weekend { Saturday, Sunday};
5.    public static void main(String[] args) {
6.      enum Currency {Dollars, Rupees, Franc, Euro};
7.      System.out.println("Hello");
8.   }
9. }
```

What is the result?

A. Hello

B. Compilation fails at line 1.

C. Compilation fails at line 3.

D. Compilation fails at line 6.

E. An exception is thrown at runtime.

5. Given:

```
1. enum Animals {
2.   DOG ("woof"), CAT ("meow"), FISH ("burble");
3.   String sound;
4.   Animals(String s) { sound = s; }
5. }
6. class TestEnum {
7.   static Animals a;
8.   public static void main(String[] args) {
9.     System.out.println(a.DOG.sound + " " + a.FISH.sound);
10.  }
11. }
```

What is the result?

A. woof burble
B. Multiple compilation errors
C. Compilation fails due to an error on line 2
D. Compilation fails due to an error on line 3
E. Compilation fails due to an error on line 4
F. Compilation fails due to an error on line 9

F. TUGAS

Tugas 1 :

Pada supermarket terdapat beberapa macam jenis buah Apel yaitu Apel Malang, Granny Smith, Pink Lady, Golden Delicious, Gala dan Red Delicious. Buatlah `enum Apel` dengan berbagai jenis apel dan harganya. Berikan deskripsi dari apel tersebut pada method `getDeskripsi()`. Selanjutnya tampilkan data semua Apel, harga dan deskripsi pada Supermarket tersebut.


```
Enum Apel{
        MALANG(//harga),...
        public String getDeskripsi() {…}
}
```

G. LAPORAN RESMI

Kerjakan hasil percobaan(D), latihan(E) dan tugas(F) di atas dan tambahkan analisa.