BAB 4
ENCAPSULATION

Tujuan

1. Praktikan mampu memahami konsep encapsulation (enkapsulasi) yang ada di java
2. Mampu memahami dan mengimplementasikan encapsulation

Ringkasan Materi

A. Encapsulation

Enkapsulasi adalah suatu cara untuk menyembunyikan informasi detail dari suatu class. Dalam enkapsulasi terdapat hak akses *public, protected, dan private*. Hak akses *public* memungkinkan semua kelas dapat mengakses meskipun berada pada paket yang berbeda, hak akses *protected* hanya diberikan kepada kelasnya sendiri dan turunannya, serta kelas-kelas dalam satu paket. Sedangkan *private* hanya boleh diakses oleh kelasnya sendiri.

Access Modifier	Class tersebut	Package	Subclass	Root / Network
Private	v			
Default	v	v		
Protected	v	v	v	
Public	v	v	v	v

Enkapsulasi bertujuan untuk menjaga suatu proses program agar tidak dapat diakses secara sembarangan atau di intervensi oleh program lain. Konsep enkapsulasi sangat penting dilakukan untuk menjaga kebutuhan program agar dapat diakses sewaktu-waktu, sekaligus menjaga program tersebut. Dua hal yang mendasar dalam enkapsulasi yakni :

A.1 Information Hiding

Sebelumnya, kita dapat mengakses anggota class baik berupa atribut maupun method secara langsung dengan menggunakan objek yang telah kita buat. Hal ini dikarenakan akses kontrol yang diberikan kepada atribut maupun method yang ada di dalam class tersebut adalah 'public'. Kita dapat menyembunyikan informasi dari suatu class sehingga anggota class tersebut tidak dapat diakses dari luar, caranya adalah hanya dengan memberikan akses kontrol 'private' ketika mendeklarasikan atribut atau method. Proses ini disebut dengan information hiding.

A.2 Interface to Access Data

Jika kita telah melakukan information hiding terhadap suatu atribut pada suatu class, lalu bagaimana melakukan perubahan terhadap atribut yang kita sembunyikan tersebut. Caranya adalah dengan membuat suatu interface berupa method untuk menginisialisasi atau merubah nilai dari suatu atribut tersebut. Manfaat utama teknik encapsulation adalah kita mampu memodifikasi kode tanpa merusak kode yang telah digunakan pada class lain. Enkapsulasi memiliki manfaat sebagai berikut:

- Modularitas

Source code dari sebuah class dapat dikelola secara independen dari source code class yang lain. Perubahan internal pada sebuah class tidak akan berpengaruh bagi class yang menggunakannya.

- Information Hiding

Penyembunyian informasi yang tidak perlu diketahui objek lain.

B. Accessor

Untuk mengimplementasikan enkapsulasi, kita tidak menginginkan sembarang object dapat mengakses data kapan saja. Untuk itu, kita deklarasikan atribut dari class sebagai private. Namun, ada kalanya dimana kita menginginkan object lain untuk dapat mengakses data private. Dalam hal ini kita gunakan accessor methods.

Accessor Methods digunakan untuk membaca nilai variabel pada class, baik berupa instance maupun static. Sebuah accessor method umumnya dimulai dengan penulisan *get<namaInstanceVariable>*. Method ini juga mempunyai sebuah return value. Sebagai contoh, kita ingin menggunakan accessor method untuk dapat membaca nama, alamat, nilai bahasa Inggris, Matematika, dan ilmu pasti dari siswa. Mari kita perhatikan salah satu contoh implementasi accessor method.

```
public class StudentRecord {
  private String name;
  :
  :
  public String getName(){
     return name;
  }
}
```

C. Mutator

Method yang dapat memberi atau mengubah nilai variable dalam class, baik itu berupa instance maupun static. Method semacam ini disebut dengan mutator methods. Sebuah mutator method umumnya tertulis *set<namaInstanceVariabel>*. Mari kita perhatikan salah satu dari implementasi mutator method.

```
public class StudentRecord{
  private String name;
  :
  :
  public void setName( String temp ){
      name = temp;
  }
}
```

Pelaksanaan Percobaan

A. Encapsulation 1

Ketikkan program di bawah ini

```
1   public class Student {
2       private String name;
3       private int mark;
4       public void setName(String n){
5          name=n;
6       }
7       public String getName(){
8          return name;
9       }
10      public void setMark(int m){
11         mark=m;
12      }
13      public int getMark(){
14         return mark;
        }
    }
```

```
1   public class Test {
2       public static void main(String [] args) {
3           Student s1=new Student();
4           s1.setName("Enkapsulasi");
5           s1.setMark("90");
6           System.out.println("s1Name is "+s1.setName());
7           System.out.println("s1Mark is "+s1.setMark());
8           System.out.println("name dan mark "+name+" "+mark);
9       }
10  }
```

B. Encapsulation 2

Buatlah class Vehicle1

```
1   public class Vehicle1
2   {
3   private double load, maxLoad;
4
5   public Vehicle1 (double max){
6         this.maxLoad = max;
7
8   }
9   public double getLoad(){
10        return this.load;
11  }
12  public double getMaxLoad(){
13        return this.maxLoad;
14  }
15  public boolean addBox(double weight){
16        double temp = 0.0D;
17        temp = this.load + weight;
18        if(temp <= maxLoad){
19        this.load = this.load + weight;
20        return true;
21        }
```

```
22        else
23        {
24        return false;
25        }
26  }
27  }
```

```
1   public class TestVehicle1{
2         public static void main(String[] args){
3               System.out.println("Creating a vehicle with a 10,000
4   kg maximumload.");
5               Vehicle1 vehicle = new Vehicle1(10000);
6               System.out.println("Add box #1 (500kg) : " +
7   vehicle.addBox(500));
8               System.out.println("Add box #2 (250kg) : " +
9   vehicle.addBox(250));
10              System.out.println("Add box #3 (5000kg) : " +
11  vehicle.addBox(5000));
12              System.out.println("Add box #4 (4000kg) : " +
13  vehicle.addBox(4000));
14              System.out.println("Add box #5 (300kg) : " +
15  vehicle.addBox(300));
16              System.out.println("Vehicle load is "
17  +vehicle.getLoad() + "kg");
18        }
19  }
```

Data dan Analisis hasil percobaan

A. Encapsulation 1

Pertanyaan

1. Lakukan percobaan diatas dan benahi jika menemukan kesalahan!
   Di dalam kelas student tidak menemukan adanya kesalahan
   - Jika di dalam kelas main student terdapat kesalahan Menggunakan nilai integer (90) saat memanggil setMark() karena parameter yang diharapkan adalah tipe data int, bukan String.
   - Menggunakan method getName() dan getMark() untuk mendapatkan nilai yang telah diatur, dan kemudian menampilkannya.
   - Menggunakan variabel lokal studentName dan studentMark untuk menyimpan nama dan nilai siswa, dan kemudian menampilkannya.

2. Jika pada baris 6 *s1.setName* diubah menjadi *s1.getName* apa yang terjadi? jelaskan!
   Jika pada baris 6 s1.setName diubah menjadi s1.getName, maka ini akan menyebabkan kesalahan dalam kode. Pada baris ini, saya mencoba untuk mencetak nilai yang dikembalikan oleh pemanggilan s1.setName(). Namun, setName() adalah sebuah method yang tidak mengembalikan nilai apapun (berupa void), sehingga tidak ada nilai yang dapat dicetak.

3. Setelah diperbaiki, ubahlah hak akses pada baris 4 (pada class Student) menjadi *private* apa yang terjadi jika class Test dijalankan? Jelaskan!

```java
// no 3
public class Student {
    private String name;
    private int mark;
    public void setName(String n) {
        name = n;
    }
    public String getName() {
        return name;
    }
    public void setMark(int m) {
        mark = m;
    }
    public int getMark() {
        return mark;
    }

    Run | Debug
    public static void main(String[] args) {
        Student s1 = new Student();

        s1.setName(n:"Enkapsulasi");
        s1.setMark(m:90);

        String studentName = s1.getName();
        int studentMark = s1.getMark();

        System.out.println("s1Name is " + studentName);
        System.out.println("s1Mark is " + studentMark);
    }
}
```

```java
// no 3
public class Test {
    private String name;
    private int mark;

    public void setName(String n) {
        name = n;
    }

    public String getName() {
        return name;
    }

    public void setMark(int m) {
        mark = m;
    }

    public int getMark() {
        return mark;
    }
}
```

Jika hak akses pada baris 4 (pada class Student) diubah menjadi private, maka variabel name dan mark akan menjadi private, yang berarti hanya dapat diakses di dalam kelas Student itu sendiri. Oleh karena itu, kode Test akan menghasilkan kesalahan kompilasi karena mencoba mengakses variabel yang tidak dapat diakses dari luar kelas Student

4. Jika kedua kelas diatas terdapat dalam package yang sama apakah konsep enkapsulasi tetap berfungsi? jelaskan!

Ya, konsep enkapsulasi tetap berfungsi meskipun kedua kelas tersebut berada dalam package yang sama. Konsep enkapsulasi dalam pemrograman Java berkaitan dengan menyembunyikan detail implementasi internal suatu kelas dan hanya memperlihatkan fungsionalitasnya kepada kelas-kelas lain Dalam kasus ini, meskipun kelas Test dan Student berada dalam package yang sama, kelas Student masih dapat menerapkan enkapsulasi dengan baik. Variabel name dan mark masih bersifat private, yang berarti hanya dapat diakses langsung oleh kelas Student itu sendiri. Metode-metode setName, getName, setMark, dan getMark berfungsi sebagai antarmuka untuk mengakses dan memanipulasi variabel-variabel tersebut dari luar kelas Student.

B. Encapsulation 2

Pertanyaan

1. Method apakah yang menjadi accessor (getter) ?

Accessor (getter) adalah metode yang digunakan untuk mengambil nilai dari suatu variabel anggota (field) kelas. Dalam kode yang diberikan, terdapat dua metode yang berperan sebagai accessor (getter):

- getLoad(): Metode ini mengembalikan nilai dari variabel load, yang merupakan beban saat ini pada kendaraan.
- getMaxLoad(): Metode ini mengembalikan nilai dari variabel maxLoad, yang merupakan beban maksimum yang dapat ditangani oleh kendaraan.

2. Tambahkan source code berikut dibawah baris ke 6 pada class TestVehicle1.

   System.out.println("Add load(100kg) : " + (vehicle.load=500));

   Jalankan program, apakah output dari program tersebut?

   Kembalikan program seperti semula.

```
// no 2
public class TestVehicle1 {
    Run | Debug
    public static void main(String[] args) {
        System.out.println(x:"Creating a vehicle with a 10,000 kg maximum load.");
        Vehicle1 vehicle = new Vehicle1(max:10000.0);
        System.out.println("Add box #1 (500kg) : " + vehicle.addBox(weight:500.0));
        System.out.println("Add box #2 (250kg) : " + vehicle.addBox(weight:250.0));
        System.out.println("Add box #3 (5000kg) : " + vehicle.addBox(weight:5000.0));
        System.out.println("Add box #4 (4000kg) : " + vehicle.addBox(weight:4000.0));
        System.out.println("Add box #5 (300kg) : " + vehicle.addBox(weight:300.0));
        System.out.println("Vehicle load is " + vehicle.getLoad() + "kg");
        System.out.println("Add load(100kg) : " + (vehicle.getLoad() + 100));
    }
}
```

```java
// no 2
public class Vehicle1 {
    private double load, maxLoad;

    public Vehicle1(double max) {
        this.maxLoad = max;
    }

    public double getLoad() {
        return this.load;
    }

    public double getMaxLoad() {
        return this.maxLoad;
    }

    public boolean addBox(double weight) {
        double temp = 0.0D;
        temp = this.load + weight;
        if (temp <= maxLoad) {
            this.load = this.load + weight;
            return true;
        } else {
            return false;
        }
    }
}
```

3. Ubahlah tipe data pada atribut load dan maxload pada class Vehicle1 menjadi **public**. Jalankan program, apakah output dari program tersebut?
   a. Tambahkan source kode berikut dibawah baris ke 6 pada class TestVehicle1.
      System.out.println("Add load(100kg) : " + (vehicle.load=500));
      Jalankan program, apakah output dari program tersebut?
      Kembalikan program seperti semula.
   b. Tambahkan source kode berikut dibawah baris ke 12 pada class TestVehicle1.
      System.out.println("Add load(100kg) : " + (vehicle.load=500));
      Jalankan program, apakah output dari program tersebut?
      Kembalikan program seperti semula.

A .

```java
// no 3
public class Vehicle1 {
    public double load, maxLoad;

    public Vehicle1(double max) {
        this.maxLoad = max;
    }

    public double getLoad() {
        return this.load;
    }

    public double getMaxLoad() {
        return this.maxLoad;
    }

    public boolean addBox(double weight) {
        double temp = 0.0D;
        temp = this.load + weight;
        if (temp <= maxLoad) {
            this.load = this.load + weight;
            return true;
        } else {
            return false;
        }
    }
}
```

```java
// no 3
public class TestVehicle1 {
    Run | Debug
    public static void main(String[] args) {
        Vehicle1 vehicle = new Vehicle1(max:1000);
        System.out.println("Add load(100kg) : " + (vehicle.load = 500));
    }
}
```

B .

```java
// no 3b
public class Vehicle1 {
    public double load, maxLoad;

    public Vehicle1(double max) {
        this.maxLoad = max;
    }

    public double getLoad() {
        return this.load;
    }

    public double getMaxLoad() {
        return this.maxLoad;
    }

    public boolean addBox(double weight) {
        double temp = this.load + weight;
        if (temp <= maxLoad) {
            this.load += weight;
            return true;
        } else {
            return false;
        }
    }

    Run | Debug
    public static void main(String[] args) {
        Vehicle1 vehicle = new Vehicle1(max:1000);
        System.out.println("Add load(100kg) : " + (vehicle.load = 500));
    }
}
```

4. Ulangi instruksi pada nomer 4 dengan mengubah tipe data pada atribut load dan maxload pada class Vehicle1 menjadi **protected.**

```java
// no 4
public class Vehicle1 {
    protected double load, maxLoad;

    public Vehicle1(double max) {
        this.maxLoad = max;
    }

    public double getLoad() {
        return this.load;
    }

    public double getMaxLoad() {
        return this.maxLoad;
    }

    public boolean addBox(double weight) {
        double temp = this.load + weight;
        if (temp <= maxLoad) {
            this.load += weight;
            return true;
        } else {
            return false;
        }
    }
}
```

5. Ulangi instruksi pada nomer 4 dengan mengubah tipe data pada atribut load dan maxload pada class Vehicle1 menjadi **default.**

```java
// no 5
class Vehicle1 {
    double load, maxLoad;

    public Vehicle1(double max) {
        this.maxLoad = max;
    }

    public double getLoad() {
        return this.load;
    }

    public double getMaxLoad() {
        return this.maxLoad;
    }

    public boolean addBox(double weight) {
        double temp = this.load + weight;
        if (temp <= maxLoad) {
            this.load += weight;
            return true;
        } else {
            return false;
        }
    }

    Run | Debug
    public static void main(String[] args) {
        Vehicle1 vehicle = new Vehicle1(max:1000);
        System.out.println("Add load(100kg) : " + (vehicle.load = 500));
    }
}

// no 5
public class TestVehicle1 {
    Run | Debug
    public static void main(String[] args) {
        Vehicle1 vehicle = new Vehicle1(max:1000);
        System.out.println("Add load(100kg) : " + (vehicle.load = 500));
    }
}
```

Tugas Praktikum

Anda dan tim anda mendapat sebuah proyek untuk merancang sistem transaksi pada sebuah swalayan Tiny. Anda ditugasi oleh tim untuk membuat programnya berdasarkan hasil analisis tim anda :

1. Informasi akun seorang pelanggan (saldo, nomor pelanggan, nama) tidak bias diubah oleh pelanggan secara langsung.

```java
class Pelanggan {
    private int nomorPelanggan;
    private String nama;
    private double saldo;

    public Pelanggan(int nomorPelanggan, String nama, double saldo) {
        this.nomorPelanggan = nomorPelanggan;
        this.nama = nama;
        this.saldo = saldo;
    }

    public int getNomorPelanggan() {
        return nomorPelanggan;
    }

    public String getNama() {
        return nama;
    }

    public double getSaldo() {
        return saldo;
    }

    public void tambahSaldo(double jumlah) {
        saldo += jumlah;
    }


    public boolean tarikSaldo(double jumlah) {
        if (saldo >= jumlah) {
            saldo -= jumlah;
            return true;
        } else {
            System.out.println("Saldo tidak mencukupi.");
            return false;
        }
    }
}
```

p

```java
// NO 1
public class MainPelanggan {
    Run | Debug
    public static void main(String[] args) {
        Pelanggan pelanggan1 = new Pelanggan(nomorPelanggan:123456, nama:"John Doe", saldo:1000.0);
        Transaksi transaksi1 = new Transaksi(pelanggan1);

        transaksi1.prosesTransaksi(jenisTransaksi:"tambah", jumlah:500.0);
        transaksi1.prosesTransaksi(jenisTransaksi:"tarik", jumlah:200.0);
        transaksi1.prosesTransaksi(jenisTransaksi:"tarik", jumlah:2000.0);

        System.out.println("Saldo akhir: " + pelanggan1.getSaldo());
    }

    static class Transaksi {
        private Pelanggan pelanggan;

        public Transaksi(Pelanggan pelanggan) {
            this.pelanggan = pelanggan;
        }

        public void prosesTransaksi(String jenisTransaksi, double jumlah) {
            if (jenisTransaksi.equals(anObject:"tambah")) {
                pelanggan.tambahSaldo(jumlah);
                System.out.println(x:"Saldo berhasil ditambahkan.");
            } else if (jenisTransaksi.equals(anObject:"tarik")) {
                if (pelanggan.tarikSaldo(jumlah)) {
                    System.out.println(x:"Penarikan saldo berhasil.");
                } else {
                    System.out.println(x:"Penarikan saldo gagal.");
                }
            } else {
                    System.out.println(x:"Jenis transaksi tidak valid.");
                }
            }
        }
    }

```


2. Nomor pelanggan terdiri dari 10 digit, dimana 2 digit awal adalah jenis rekening
    - 38 : Pelanggan jenis silver; setiap pembelian diatas 1 jt maka mendapat cashback sebesar 5%

```java
// no 2
public class Pelanggan {
    private long nomorPelanggan;
    private String nama;
    private double saldo;

    public Pelanggan(long nomorPelanggan, String nama, double saldo) {
        this.nomorPelanggan = nomorPelanggan;
        this.nama = nama;
        this.saldo = saldo;
    }

    public long getNomorPelanggan() {
        return nomorPelanggan;
    }

    public String getNama() {
        return nama;
    }

    public double getSaldo() {
        return saldo;
    }

    public void tambahSaldo(double jumlah) {
        saldo += jumlah;
        int jenisRekening = (int) (nomorPelanggan / 100000000);
        if (jenisRekening == 38 && jumlah > 1000000) {
            double cashback = jumlah * 0.05;
            saldo += cashback;
            System.out.println("Anda mendapat cashback sebesar " + cashback);
        }
    }

    public boolean tarikSaldo(double jumlah) {
        if (saldo >= jumlah) {
            saldo -= jumlah;
            return true;
        } else {
            System.out.println("Saldo tidak mencukupi.");
            return false;
        }
    }
}

```

- 56 : Pelanggan jenis gold; setiap pembelian diatas 1 jt maka mendapat cashback sebesar 7%, selain itu cashback 2% (cashback kembali ke saldo)

```java
// no 3
public class Pelanggan {
    private long nomorPelanggan;
    private String nama;
    private double saldo;

    public Pelanggan(long nomorPelanggan, String nama, double saldo) {
        this.nomorPelanggan = nomorPelanggan;
        this.nama = nama;
        this.saldo = saldo;
    }

    public long getNomorPelanggan() {
        return nomorPelanggan;
    }

    public String getNama() {
        return nama;
    }

    public double getSaldo() {
        return saldo;
    }

    public void tambahSaldo(double jumlah) {
        saldo += jumlah;
        int jenisRekening = (int) (nomorPelanggan / 100000000);
        if (jenisRekening == 38) {
            if (jumlah > 1000000) {
                double cashback = jumlah * 0.05;
                saldo += cashback;
                System.out.println("Anda mendapat cashback sebesar " + cashback);
            }
        }
        else if (jenisRekening == 56) {
            if (jumlah > 1000000) {
                double cashback = jumlah * 0.07;
                saldo += cashback;
                System.out.println("Anda mendapat cashback sebesar " + cashback);
            } else {
                double cashback = jumlah * 0.02;
                saldo += cashback;
                System.out.println("Anda mendapat cashback sebesar " + cashback);
            }
        }
    }

    public boolean tarikSaldo(double jumlah) {
        if (saldo >= jumlah) {
            saldo -= jumlah;
            return true;
        } else {
            System.out.println("Saldo tidak mencukupi.");
            return false;
        }
    }
}

```

- 74 : Pelanggan jenis platinum; setiap pembelian diatas 1 jt maka mendapat cashback sebesar 10%, selain itu cashback 5% (cashback kembali ke saldo)

```java
public class Pelanggan {
    private long nomorPelanggan; // Menggunakan long untuk menampung nomor pelanggan 10 digit
    private String nama;
    private double saldo;

    public Pelanggan(long nomorPelanggan, String nama, double saldo) {
        this.nomorPelanggan = nomorPelanggan;
        this.nama = nama;
        this.saldo = saldo;
    }

    public long getNomorPelanggan() {
        return nomorPelanggan;
    }

    public String getNama() {
        return nama;
    }

    public double getSaldo() {
        return saldo;
    }

    public void tambahSaldo(double jumlah) {
        saldo += jumlah;

        // Menentukan jenis rekening berdasarkan dua digit pertama nomor pelanggan
        int jenisRekening = (int) (nomorPelanggan / 100000000); // Ambil dua digit pertama

        // Cek jenis rekening
        switch (jenisRekening) {
            case 38: // Silver
                if (jumlah > 1000000) {
                    double cashbackSilver = jumlah * 0.05;
                    saldo += cashbackSilver;
                    System.out.println("Anda mendapat cashback sebesar " + cashbackSilver);
                }
                break;
            case 56: // Gold
                if (jumlah > 1000000) {
                    double cashbackGold = jumlah * 0.07;
                    saldo += cashbackGold;
                    System.out.println("Anda mendapat cashback sebesar " + cashbackGold);
                } else {
                    double cashbackGold = jumlah * 0.02;
                    saldo += cashbackGold;
                    System.out.println("Anda mendapat cashback sebesar " + cashbackGold);
                }
                break;
```

```
            case 74: // Platinum
                if (jumlah > 1000000) {
                    double cashbackPlatinum = jumlah * 0.10;
                    saldo += cashbackPlatinum;
                    System.out.println("Anda mendapat cashback sebesar " + cashbackPlatinum);
                } else {
                    double cashbackPlatinum = jumlah * 0.05;
                    saldo += cashbackPlatinum;
                    System.out.println("Anda mendapat cashback sebesar " + cashbackPlatinum);
                }
                break;
            default:
                System.out.println("Jenis rekening tidak valid.");
        }
    }

    public boolean tarikSaldo(double jumlah) {
        if (saldo >= jumlah) {
            saldo -= jumlah;
            return true;
        } else {
            System.out.println("Saldo tidak mencukupi.");
            return false;
        }
    }
}
```

3. Pelanggan harus memiliki saldo minimal Rp10.000, jika saldo pasca transaksi kurang dari batas minimal tadi, maka transaksi pembelian dianggap gagal

```
// no 3
public class Pelanggan {
    private long nomorPelanggan;
    private String nama;
    private double saldo;
    private final double SALDO_MINIMAL = 10000;

    public Pelanggan(long nomorPelanggan, String nama, double saldo) {
        this.nomorPelanggan = nomorPelanggan;
        this.nama = nama;
        this.saldo = saldo;
    }

    public long getNomorPelanggan() {
        return nomorPelanggan;
    }

    public String getNama() {
        return nama;
    }

    public double getSaldo() {
        return saldo;
    }

```

```java
    public void tambahSaldo(double jumlah) {
        saldo += jumlah;
        int jenisRekening = (int) (nomorPelanggan / 100000000);
        switch (jenisRekening) {
            case 38:
                if (jumlah > 1000000) {
                    double cashbackSilver = jumlah * 0.05;
                    saldo += cashbackSilver;
                    System.out.println("Anda mendapat cashback sebesar " + cashbackSilver);
                }
                break;
            case 56:
                if (jumlah > 1000000) {
                    double cashbackGold = jumlah * 0.07;
                    saldo += cashbackGold;
                    System.out.println("Anda mendapat cashback sebesar " + cashbackGold);
                } else {
                    double cashbackGold = jumlah * 0.02;
                    saldo += cashbackGold;
                    System.out.println("Anda mendapat cashback sebesar " + cashbackGold);
                }
                break;
            case 74:
                if (jumlah > 1000000) {
                    double cashbackPlatinum = jumlah * 0.10;
                    saldo += cashbackPlatinum;
                    System.out.println("Anda mendapat cashback sebesar " + cashbackPlatinum);
                } else {
                    double cashbackPlatinum = jumlah * 0.05;
                    saldo += cashbackPlatinum;
                    System.out.println("Anda mendapat cashback sebesar " + cashbackPlatinum);
                }
                break;
            default:
                System.out.println("Jenis rekening tidak valid.");
        }
    }

    public boolean tarikSaldo(double jumlah) {
        if (saldo - jumlah >= SALDO_MINIMAL) {
            saldo -= jumlah;
            return true;
        } else {
            System.out.println("Transaksi gagal. Saldo minimal Rp10.000.");
            return false;
        }
    }
}
```

4. Buatlah sistem transaksi swalayan ini terbatas pada pembelian dan top up saja dan menggunakan PIN dan nomor pelanggan sebagai syarat transaksi pembelian atau top up.

```java
// no 4
public class Pelanggan {
    private int nomorPelanggan;
    private String nama;
    private double saldo;
    private int pin;

    public Pelanggan(int nomorPelanggan, String nama, double saldo, int pin) {
        this.nomorPelanggan = nomorPelanggan;
        this.nama = nama;
        this.saldo = saldo;
        this.pin = pin;
    }

    public int getNomorPelanggan() {
        return nomorPelanggan;
    }

    public String getNama() {
        return nama;
    }

    public double getSaldo() {
        return saldo;
    }

    public int getPIN() {
        return pin;
    }

    public void tambahSaldo(double jumlah) {
        this.saldo += jumlah;
    }
}
```

```java
// no 4
public class MainPelanggan {
    Run | Debug
    public static void main(String[] args) {
        // Membuat objek pelanggan
        Pelanggan pelanggan1 = new Pelanggan(nomorPelanggan:123456, nama:"John Doe", saldo:1000.0, pin:1234);

        // Menampilkan informasi pelanggan
        System.out.println(x:"Informasi Pelanggan:");
        System.out.println("Nomor Pelanggan: " + pelanggan1.getNomorPelanggan());
        System.out.println("Nama: " + pelanggan1.getNama());
        System.out.println("Saldo: " + pelanggan1.getSaldo());

        // Melakukan penambahan saldo
        pelanggan1.tambahSaldo(jumlah:500.0);

        // Menampilkan saldo setelah penambahan
        System.out.println("Saldo setelah penambahan: " + pelanggan1.getSaldo());
    }
}
```

5. Apabila pelanggan melakukan 3x kesalahan dalam autentifikasi, maka akun pelanggan akan defreeze / diblokir sehingga tidak bisa digunakan lagi.

```java
// no 5
public class Pelanggan {
    private int nomorPelanggan;
    private String nama;
    private double saldo;
    private int pin;
    private int kesalahanAutentifikasi;

    private boolean terblokir;

    public Pelanggan(int nomorPelanggan, String nama, double saldo, int pin) {
        this.nomorPelanggan = nomorPelanggan;
        this.nama = nama;
        this.saldo = saldo;
        this.pin = pin;
        this.kesalahanAutentifikasi = 0;
        this.terblokir = false;
    }

    public int getNomorPelanggan() {
        return nomorPelanggan;
    }

    public String getNama() {
        return nama;
    }

    public double getSaldo() {
        return saldo;
    }

    public int getPIN() {
        return pin;
    }

    public boolean isTerblokir() {
        return terblokir;
    }

    public void tambahSaldo(double jumlah) {
        this.saldo += jumlah;
    }

    public boolean verifikasiPIN(int pin) {
        if (!terblokir && this.pin == pin) {
            kesalahanAutentifikasi = 0;
            return true;
        } else {
            kesalahanAutentifikasi++;
            if (kesalahanAutentifikasi >= 3) {
                terblokir = true;
                System.out.println(x:"Akun terblokir. Silakan hubungi layanan pelanggan.");
            }
            return false;
        }
    }
}
```

```java
// no 5
public class MainPelanggan {
    Run | Debug
    public static void main(String[] args) {
        Pelanggan pelanggan1 = new Pelanggan(nomorPelanggan:123456, nama:"John Doe", saldo:1000.0, pin:1234);
        System.out.println(x:"Informasi Pelanggan:");
        System.out.println("Nomor Pelanggan: " + pelanggan1.getNomorPelanggan());
        System.out.println("Nama: " + pelanggan1.getNama());
        System.out.println("Saldo: " + pelanggan1.getSaldo());

        for (int i = 0; i < 3; i++) {
            boolean berhasil = pelanggan1.verifikasiPIN(pin:9999);
            if (!berhasil) {
                System.out.println("Kesalahan ke-" + (i + 1) + " dalam autentifikasi.");
            }
        }
        boolean berhasil = pelanggan1.verifikasiPIN(pin:1234);
        if (!berhasil) {
            System.out.println(x:"Kesalahan dalam autentifikasi. Akun terblokir.");
        } else {
            pelanggan1.tambahSaldo(jumlah:500.0);
            System.out.println("Saldo setelah penambahan: " + pelanggan1.getSaldo());
        }
    }
}
```