

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 diintervensi 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;
15    }
16 }
```

```
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    }
```

```

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!

```
// 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);
}
}
```

```
36 // no 3
37 public class Test {
38     private String name;
39     private int mark;
40
41     public void setName(String n) {
42         name = n;
43     }
44
45     public String getName() {
46         return name;
47     }
48
49     public void setMark(int m) {
50         mark = m;
51     }
52
53     public int getMark() {
54         return mark;
55     }
56 }
57
```

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("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));
    }
}
```

```
// 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 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.
 - b. Tambahkan source code 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 .

```

51 // no 3
52 public class Vehicle1 {
53     public double load, maxLoad;
54
55     public Vehicle1(double max) {
56         this.maxLoad = max;
57     }
58
59     public double getLoad() {
60         return this.load;
61     }
62
63     public double getMaxLoad() {
64         return this.maxLoad;
65     }
66
67     public boolean addBox(double weight) {
68         double temp = 0.00;
69         temp = this.load + weight;
70         if (temp <= maxLoad) {
71             this.load = this.load + weight;
72             return true;
73         } else {
74             return false;
75         }
76     }
77 }
78
79 // no 3
80 public class TestVehicle1 {
81     Run | Debug
82     public static void main(String[] args) {
83         Vehicle1 vehicle = new Vehicle1(max:1000);
84         System.out.println("Add load(100kg) : " + (vehicle.load = 500));
85     }
86 }
87
88
89
90
91
92
93

```

B.


```

79 // no 3b
80 public class Vehicle1 {
81     public double load, maxLoad;
82
83     public Vehicle1(double max) {
84         this.maxLoad = max;
85     }
86
87     public double getLoad() {
88         return this.load;
89     }
90
91     public double getMaxLoad() {
92         return this.maxLoad;
93     }
94
95     public boolean addBox(double weight) {
96         double temp = this.load + weight;
97         if (temp <= maxLoad) {
98             this.load += weight;
99             return true;
100         } else {
101             return false;
102         }
103     }
104
105     Run | Debug
106     public static void main(String[] args) {
107         Vehicle1 vehicle = new Vehicle1(max:1000);
108         System.out.println("Add load(100kg) : " + (vehicle.load = 500));
109     }

```

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

```
110
111 // no 4
112 public class Vehicle1 {
113     protected double load, maxLoad;
114
115     public Vehicle1(double max) {
116         this.maxLoad = max;
117     }
118
119     public double getLoad() {
120         return this.load;
121     }
122
123     public double getMaxLoad() {
124         return this.maxLoad;
125     }
126
127     public boolean addBox(double weight) {
128         double temp = this.load + weight;
129         if (temp <= maxLoad) {
130             this.load += weight;
131             return true;
132         } else {
133             return false;
134         }
135     }
136 }
```

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

```

138 // no 5
139 class Vehicle1 {
140     double load, maxLoad;
141
142     public Vehicle1(double max) {
143         this.maxLoad = max;
144     }
145
146     public double getLoad() {
147         return this.load;
148     }
149
150     public double getMaxLoad() {
151         return this.maxLoad;
152     }
153
154     public boolean addBox(double weight) {
155         double temp = this.load + weight;
156         if (temp <= maxLoad) {
157             this.load += weight;
158             return true;
159         } else {
160             return false;
161         }
162     }
163
164     Run | Debug
165     public static void main(String[] args) {
166         Vehicle1 vehicle = new Vehicle1(max:1000);
167         System.out.println("Add load(100kg) : " + (vehicle.load = 500));
168     }
169 }
170
171 // no 5
172 public class TestVehicle1 {
173     Run | Debug
174     public static void main(String[] args) {
175         Vehicle1 vehicle = new Vehicle1(max:1000);
176         System.out.println("Add load(100kg) : " + (vehicle.load = 500));
177     }
178 }
179
180
181

```

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.

```

1  class Pelanggan {
2      private int nomorPelanggan;
3      private String nama;
4      private double saldo;
5
6      public Pelanggan(int nomorPelanggan, String nama, double saldo) {
7          this.nomorPelanggan = nomorPelanggan;
8          this.nama = nama;
9          this.saldo = saldo;
10     }
11
12     public int getNomorPelanggan() {
13         return nomorPelanggan;
14     }
15
16     public String getNama() {
17         return nama;
18     }
19
20     public double getSaldo() {
21         return saldo;
22     }
23
24     public void tambahSaldo(double jumlah) {
25         saldo += jumlah;
26     }
27
28     public boolean tarikSaldo(double jumlah) {
29         if (saldo >= jumlah) {
30             saldo -= jumlah;
31             return true;
32         } else {
33             System.out.println("Saldo tidak mencukupi.");
34             return false;
35         }
36     }
37 }

```

```

1 // NO 1
2 public class MainPelanggan {
3     Run | Debug
4     public static void main(String[] args) {
5         Pelanggan pelanggan1 = new Pelanggan(nomorPelanggan:123456, nama:"John Doe", saldo:1000.0);
6         Transaksi transaksi1 = new Transaksi(pelanggan1);
7
8         transaksi1.prosesTransaksi(jenisTransaksi:"tambah", jumlah:500.0);
9         transaksi1.prosesTransaksi(jenisTransaksi:"tarik", jumlah:200.0);
10        transaksi1.prosesTransaksi(jenisTransaksi:"tarik", jumlah:2000.0);
11
12        System.out.println("Saldo akhir: " + pelanggan1.getSaldo());
13    }
14
15    static class Transaksi {
16        private Pelanggan pelanggan;
17
18        public Transaksi(Pelanggan pelanggan) {
19            this.pelanggan = pelanggan;
20        }
21
22        public void prosesTransaksi(String jenisTransaksi, double jumlah) {
23            if (jenisTransaksi.equals(anObject:"tambah")) {
24                pelanggan.tambahSaldo(jumlah);
25                System.out.println(x:"Saldo berhasil ditambahkan.");
26            } else if (jenisTransaksi.equals(anObject:"tarik")) {
27                if (pelanggan.tarikSaldo(jumlah)) {
28                    System.out.println(x:"Penarikan saldo berhasil.");
29                } else {
30                    System.out.println(x:"Penarikan saldo gagal.");
31                }
32            } else {
33                System.out.println(x:"Jenis transaksi tidak valid.");
34            }
35        }
36    }
37 }

```

Activate Windows
Go to Settings to activate Windows.

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

```

40 // no 2
41 public class Pelanggan {
42     private long nomorPelanggan;
43     private String nama;
44     private double saldo;
45
46     public Pelanggan(long nomorPelanggan, String nama, double saldo) {
47         this.nomorPelanggan = nomorPelanggan;
48         this.nama = nama;
49         this.saldo = saldo;
50     }
51
52     public long getNomorPelanggan() {
53         return nomorPelanggan;
54     }
55
56     public String getNama() {
57         return nama;
58     }
59
60     public double getSaldo() {
61         return saldo;
62     }
63
64     public void tambahSaldo(double jumlah) {
65         saldo += jumlah;
66         int jenisRekening = (int) (nomorPelanggan / 100000000);
67         if (jenisRekening == 38 && jumlah > 1000000) {
68             double cashback = jumlah * 0.05;
69             saldo += cashback;
70             System.out.println("Anda mendapat cashback sebesar " + cashback);
71         }
72     }
73
74     public boolean tarikSaldo(double jumlah) {
75         if (saldo >= jumlah) {
76             saldo -= jumlah;
77             return true;
78         } else {
79             System.out.println("Saldo tidak mencukupi.");
80             return false;
81         }
82     }
83 }
84

```

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

```

85 // no 3
86 public class Pelanggan {
87     private long nomorPelanggan;
88     private String nama;
89     private double saldo;
90
91     public Pelanggan(long nomorPelanggan, String nama, double saldo) {
92         this.nomorPelanggan = nomorPelanggan;
93         this.nama = nama;
94         this.saldo = saldo;
95     }
96
97     public long getNomorPelanggan() {
98         return nomorPelanggan;
99     }
100
101     public String getNama() {
102         return nama;
103     }
104
105     public double getSaldo() {
106         return saldo;
107     }
108
109     public void tambahSaldo(double jumlah) {
110         saldo += jumlah;
111         int jenisRekening = (int) (nomorPelanggan / 100000000);
112         if (jenisRekening == 38) {
113             if (jumlah > 1000000) {
114                 double cashback = jumlah * 0.05;
115                 saldo += cashback;
116                 System.out.println("Anda mendapat cashback sebesar " + cashback);
117             }
118         }
119         else if (jenisRekening == 56) {
120             if (jumlah > 1000000) {
121                 double cashback = jumlah * 0.07;
122                 saldo += cashback;
123                 System.out.println("Anda mendapat cashback sebesar " + cashback);
124             }
125             else {
126                 double cashback = jumlah * 0.02;
127                 saldo += cashback;
128                 System.out.println("Anda mendapat cashback sebesar " + cashback);
129             }
130         }
131     }
132
133     public boolean tarikSaldo(double jumlah) {
134         if (saldo >= jumlah) {
135             saldo -= jumlah;
136             return true;
137         } else {
138             System.out.println("Saldo tidak mencukupi.");
139             return false;
140         }
141     }
142 }

```

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

```
143 public class Pelanggan {
144     private long nomorPelanggan; // Menggunakan long untuk menampung nomor pelanggan 10 digit
145     private String nama;
146     private double saldo;
147
148     public Pelanggan(long nomorPelanggan, String nama, double saldo) {
149         this.nomorPelanggan = nomorPelanggan;
150         this.nama = nama;
151         this.saldo = saldo;
152     }
153
154     public long getNomorPelanggan() {
155         return nomorPelanggan;
156     }
157
158     public String getNama() {
159         return nama;
160     }
161
162     public double getSaldo() {
163         return saldo;
164     }
165
166     public void tambahSaldo(double jumlah) {
167         saldo += jumlah;
168
169         // Menentukan jenis rekening berdasarkan dua digit pertama nomor pelanggan
170         int jenisRekening = (int) (nomorPelanggan / 100000000); // Ambil dua digit pertama
171
172         // Cek jenis rekening
173         switch (jenisRekening) {
174             case 38: // Silver
175                 if (jumlah > 1000000) {
176                     double cashbackSilver = jumlah * 0.05;
177                     saldo += cashbackSilver;
178                     System.out.println("Anda mendapat cashback sebesar " + cashbackSilver);
179                 }
180                 break;
181             case 56: // Gold
182                 if (jumlah > 1000000) {
183                     double cashbackGold = jumlah * 0.07;
184                     saldo += cashbackGold;
185                     System.out.println("Anda mendapat cashback sebesar " + cashbackGold);
186                 } else {
187                     double cashbackGold = jumlah * 0.02;
188                     saldo += cashbackGold;
189                     System.out.println("Anda mendapat cashback sebesar " + cashbackGold);
190                 }
191                 break;
192         }
193     }
194 }
```



```

192         case 74: // Platinum
193             if (jumlah > 1000000) {
194                 double cashbackPlatinum = jumlah * 0.10;
195                 saldo += cashbackPlatinum;
196                 System.out.println("Anda mendapat cashback sebesar " + cashbackPlatinum);
197             } else {
198                 double cashbackPlatinum = jumlah * 0.05;
199                 saldo += cashbackPlatinum;
200                 System.out.println("Anda mendapat cashback sebesar " + cashbackPlatinum);
201             }
202             break;
203         default:
204             System.out.println("Jenis rekening tidak valid.");
205     }
206 }
207
208 public boolean tarikSaldo(double jumlah) {
209     if (saldo >= jumlah) {
210         saldo -= jumlah;
211         return true;
212     } else {
213         System.out.println("Saldo tidak mencukupi.");
214         return false;
215     }
216 }
217 }
218

```

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

```

220 // no 3
221 public class Pelanggan {
222     private long nomorPelanggan;
223     private String nama;
224     private double saldo;
225     private final double SALDO_MINIMAL = 10000;
226
227     public Pelanggan(long nomorPelanggan, String nama, double saldo) {
228         this.nomorPelanggan = nomorPelanggan;
229         this.nama = nama;
230         this.saldo = saldo;
231     }
232
233     public long getNomorPelanggan() {
234         return nomorPelanggan;
235     }
236
237     public String getNama() {
238         return nama;
239     }
240
241     public double getSaldo() {
242         return saldo;
243     }
244

```

```

245 public void tambahSaldo(double jumlah) {
246     saldo += jumlah;
247     int jenisRekening = (int) (nomorPelanggan / 100000000);
248     switch (jenisRekening) {
249         case 38:
250             if (jumlah > 1000000) {
251                 double cashbackSilver = jumlah * 0.05;
252                 saldo += cashbackSilver;
253                 System.out.println("Anda mendapat cashback sebesar " + cashbackSilver);
254             }
255             break;
256         case 56:
257             if (jumlah > 1000000) {
258                 double cashbackGold = jumlah * 0.07;
259                 saldo += cashbackGold;
260                 System.out.println("Anda mendapat cashback sebesar " + cashbackGold);
261             } else {
262                 double cashbackGold = jumlah * 0.02;
263                 saldo += cashbackGold;
264                 System.out.println("Anda mendapat cashback sebesar " + cashbackGold);
265             }
266             break;
267         case 74:
268             if (jumlah > 1000000) {
269                 double cashbackPlatinum = jumlah * 0.10;
270                 saldo += cashbackPlatinum;
271                 System.out.println("Anda mendapat cashback sebesar " + cashbackPlatinum);
272             } else {
273                 double cashbackPlatinum = jumlah * 0.05;
274                 saldo += cashbackPlatinum;
275                 System.out.println("Anda mendapat cashback sebesar " + cashbackPlatinum);
276             }
277             break;
278         default:
279             System.out.println("Jenis rekening tidak valid.");
280     }
281 }
282
283 public boolean tarikSaldo(double jumlah) {
284     if (saldo - jumlah >= SALDO_MINIMAL) {
285         saldo -= jumlah;
286         return true;
287     } else {
288         System.out.println("Transaksi gagal. Saldo minimal Rp10.000.");
289         return false;
290     }
291 }
292 }
293

```

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.

```

347 // no 4
348 public class Pelanggan {
349     private int nomorPelanggan;
350     private String nama;
351     private double saldo;
352     private int pin;
353
354     public Pelanggan(int nomorPelanggan, String nama, double saldo, int pin) {
355         this.nomorPelanggan = nomorPelanggan;
356         this.nama = nama;
357         this.saldo = saldo;
358         this.pin = pin;
359     }
360
361     public int getNomorPelanggan() {
362         return nomorPelanggan;
363     }
364
365     public String getNama() {
366         return nama;
367     }
368
369     public double getSaldo() {
370         return saldo;
371     }
372
373     public int getPIN() {
374         return pin;
375     }
376
377     public void tambahSaldo(double jumlah) {
378         this.saldo += jumlah;
379     }
380 }
381
38 // no 4
39 public class MainPelanggan {
40     Run | Debug
41     public static void main(String[] args) {
42         // Membuat objek pelanggan
43         Pelanggan pelanggan1 = new Pelanggan(nomorPelanggan:123456, nama:"John Doe", saldo:1000.0, pin:1234);
44
45         // Menampilkan informasi pelanggan
46         System.out.println("Informasi Pelanggan:");
47         System.out.println("Nomor Pelanggan: " + pelanggan1.getNomorPelanggan());
48         System.out.println("Nama: " + pelanggan1.getNama());
49         System.out.println("Saldo: " + pelanggan1.getSaldo());
50
51         // Melakukan penambahan saldo
52         pelanggan1.tambahSaldo(jumlah:500.0);
53
54         // Menampilkan saldo setelah penambahan
55         System.out.println("Saldo setelah penambahan: " + pelanggan1.getSaldo());
56     }
57 }

```

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

```

382 // no 5
383 public class Pelanggan {
384     private int nomorPelanggan;
385     private String nama;
386     private double saldo;
387     private int pin;
388     private int kesalahanAutentifikasi;
389
390     private boolean terblokir;
391
392     public Pelanggan(int nomorPelanggan, String nama, double saldo, int pin) {
393         this.nomorPelanggan = nomorPelanggan;
394         this.nama = nama;
395         this.saldo = saldo;
396         this.pin = pin;
397         this.kesalahanAutentifikasi = 0;
398         this.terblokir = false;
399     }
400
401     public int getNomorPelanggan() {
402         return nomorPelanggan;
403     }
404
405     public String getNama() {
406         return nama;
407     }
408
409     public double getSaldo() {
410         return saldo;
411     }
412
413     public int getPIN() {
414         return pin;
415     }
416
417     public boolean isTerblokir() {
418         return terblokir;
419     }
420
421     public void tambahSaldo(double jumlah) {
422         this.saldo += jumlah;
423     }
424
425     public boolean verifikasiPIN(int pin) {
426         if (!terblokir && this.pin == pin) {
427             kesalahanAutentifikasi = 0;
428             return true;
429         } else {
430             kesalahanAutentifikasi++;
431             if (kesalahanAutentifikasi >= 3) {
432                 terblokir = true;
433                 System.out.println(x:"Akun terblokir. Silakan hubungi layanan pelanggan.");
434             }
435             return false;
436         }
437     }
438 }
439

```

```
58 // no 5
59 public class MainPelanggan {
60     public static void main(String[] args) {
61         Pelanggan pelanggan1 = new Pelanggan(nomorPelanggan:123456, nama:"John Doe", saldo:1000.0, pin:1234);
62         System.out.println(x:"Informasi Pelanggan:");
63         System.out.println("Nomor Pelanggan: " + pelanggan1.getNomorPelanggan());
64         System.out.println("Nama: " + pelanggan1.getNama());
65         System.out.println("Saldo: " + pelanggan1.getSaldo());
66
67         for (int i = 0; i < 3; i++) {
68             boolean berhasil = pelanggan1.verifikasiPIN(pin:9999);
69             if (!berhasil) {
70                 System.out.println("Kesalahan ke-" + (i + 1) + " dalam autentifikasi.");
71             }
72
73             boolean berhasil = pelanggan1.verifikasiPIN(pin:1234);
74             if (!berhasil) {
75                 System.out.println(x:"Kesalahan dalam autentifikasi. Akun terblokir.");
76             } else {
77                 pelanggan1.tambahSaldo(jumlah:500.0);
78                 System.out.println("Saldo setelah penambahan: " + pelanggan1.getSaldo());
79             }
80         }
81     }
82 }
```