

## LAPORAN PRAKTIKUM MINGGU 2

Topik: Class dan Object (Produk Pertanian)

### A. IDENTITAS

Nama	: Sri Wahyuningsih
Nim	: 240202844
Kelas	: 3IKRA

### B. TUJUAN

Praktikum ini memiliki tujuan sebagai berikut:

1. Mahasiswa memahami konsep class dan object dalam pemrograman berorientasi objek (OOP).
2. Mahasiswa dapat menerapkan enkapsulasi menggunakan access modifier dan getter/setter.
3. Mahasiswa mampu membuat class Produk untuk merepresentasikan produk pertanian.
4. Mahasiswa mampu menginstansiasi beberapa objek dan menampilkan informasinya di console.

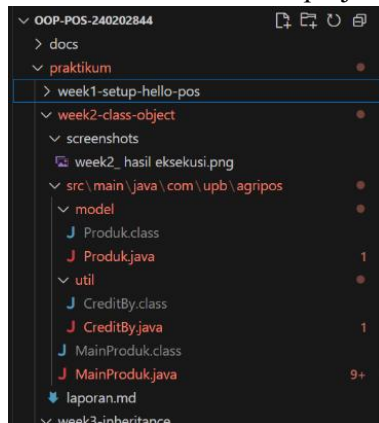
### C. DASAR TEORI

Adapun dasar teori yang mendasari praktikum ini, diantaranya:

1. Class adalah *blueprint* dari objek, berisi atribut dan method.
2. Object adalah instansiasi dari class yang merepresentasikan entitas nyata.
3. Enkapsulasi digunakan untuk melindungi data dengan membuat atribut bersifat *private*.
4. Getter dan Setter digunakan untuk mengakses dan mengubah nilai atribut secara aman. Dalam OOP, setiap objek dapat berinteraksi melalui method untuk menghasilkan perilaku tertentu.

### D. LANGKAH PRAKTIKUM

1. Membuat struktur folder project:



2. Buat class Produk dengan atribut kode, nama, harga, dan stok, serta method getter dan setter.
3. Buat class CreditBy pada package util yang menampilkan identitas mahasiswa dengan format credit by: <NIM> - <Nama>.
4. Instansiasi minimal tiga produk pertanian dan tampilkan informasinya di console, diakhiri dengan pemanggilan CreditBy.print().
5. Tambahkan method tambahStok(int jumlah) dan kurangiStok(int jumlah) untuk mengelola stok produk.

6. Jalankan perintah kompilasi dengan javac, lalu jalankan program dengan perintah java agar muncul hasil eksekusinya

```
a85c785..799e882 main -> main
PS C:\Users\sri61\Documents\oop-pos-240202844\praktikum\week2-class-object\src\main\java> javac com/upb/agripos/model/Produk.java
PS C:\Users\sri61\Documents\oop-pos-240202844\praktikum\week2-class-object\src\main\java> javac com/upb/agripos/util/CreditBy.java
PS C:\Users\sri61\Documents\oop-pos-240202844\praktikum\week2-class-object\src\main\java> javac com/upb/agripos/MainProduk.java
PS C:\Users\sri61\Documents\oop-pos-240202844\praktikum\week2-class-object\src\main\java> java com.upb.agripos.MainProduk
=== Info Awal Produk ===
```

7. Melakukan *commit* dan *push*

```
PS C:\Users\sri61\Documents\oop-pos-240202844\praktikum\week2-class-object\src\main\java> git add .
PS C:\Users\sri61\Documents\oop-pos-240202844\praktikum\week2-class-object\src\main\java> git commit -m "upload praktikum 2"
On branch main
Your branch is ahead of 'origin/main' by 2 commits.
(use "git push" to publish your local commits)

nothing to commit, working tree clean
PS C:\Users\sri61\Documents\oop-pos-240202844\praktikum\week2-class-object\src\main\java> git push origin main
Enumerating objects: 29, done.
Counting objects: 100% (29/29), done.
Delta compression using up to 2 threads
Compressing objects: 100% (12/12), done.
Writing objects: 100% (18/18), 1.86 KiB | 272.00 KiB/s, done.
Total 18 (delta 6), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (6/6), completed with 4 local objects.
To https://github.com/sriwaa/oop-202501-240202844.git
a85c785..799e882 main -> main
```

## E. KODE PROGRAM

1. Produk.java

```
praktikum > week2-class-object > src > main > java > com > upb > agripos > model > J Produk.java > Q kurangiStok()
1 package com.upb.agripos.model;
2
3 public class Produk {
4     private String kode;
5     private String nama;
6     private double harga;
7     private int stok;
8
9     public Produk(String kode, String nama, double harga, int stok) {
10         this.kode = kode;
11         this.nama = nama;
12         this.harga = harga;
13         this.stok = stok;
14     }
15
16     public String getKode() { return kode; }
17     public void setKode(String kode) { this.kode = kode; }
18     public String getNama() { return nama; }
19     public void setNama(String nama) { this.nama = nama; }
20
21     public double getHarga() { return harga; }
22     public void setHarga(double harga) { this.harga = harga; }
23
24     public int getStok() { return stok; }
25     public void setStok(int stok) { this.stok = stok; }
26
27     public void tambahStok(int jumlah) {
28         stok += jumlah;
29     }
30 }
```

```
praktikum > week2-class-object > src > main > java > com > upb > agripos > model > J Produk.java > Q kurangiStok()
1 public class Produk {
2
3     public int getStok() { return stok; }
4     public void setStok(int stok) { this.stok = stok; }
5
6     public void tambahStok(int jumlah) {
7         stok += jumlah;
8         System.out.println("Berhasil menambah stok " + jumlah + " unit. Stok sekarang: " + stok);
9     }
10
11     public void kurangiStok(int jumlah) {
12         if (stok >= jumlah) {
13             stok -= jumlah;
14             System.out.println("Berhasil mengurangi stok " + jumlah + " unit. Stok sekarang: " + stok);
15         } else {
16             System.out.println("Stok tidak cukup! Tidak bisa mengurangi " + jumlah + " unit.");
17         }
18     }
19
20     public void tampilkanInfo() {
21         System.out.println("Kode: " + kode + ", Nama: " + nama +
22             ", Harga: " + harga + ", Stok: " + stok);
23     }
24 }
```

2. CreditBy.java

```
J MainProduk.java 9+ J Produk.java 1 J CreditBy.java 1 X J CreditBy.class J Produk.class
praktikum > week2-class-object > src > main > java > com > upb > agripos > util > J CreditBy.java > ...
1 package com.upb.agripos.util;
2
3 public class CreditBy {
4     public static void print() {
5         System.out.println(x:"\ncredit by: 240202844 - sriwaa");
6     }
7 }
8
```

3. MainProduk.java

```
J MainProduk.java 9+ X J Produk.java 1 J CreditBy.java 1 J CreditBy.class J Produk.class
praktikum > week2-class-object > src > main > java > com > upb > agripos > J MainProduk.java > ...
1 package com.upb.agripos;
2
3 import com.upb.agripos.model.Produk;
4 import com.upb.agripos.util.CreditBy;
5
6 public class MainProduk {
7     Run (Debug)
8     public static void main(String[] args) {
9
10         Produk p1 = new Produk("M01-002", "Benih Stroberyry AW55", 80000.0, 100);
11         Produk p2 = new Produk("SSR-005", "Pupuk Hayati 25kg", 90000.0, 100);
12         Produk p3 = new Produk("SKW-025", "Sekop Tangan", 50000.0, 100);
13
14         System.out.println(x:"=== Info Awal Produk ===");
15         p1.tampilkanInfo();
16         p2.tampilkanInfo();
17         p3.tampilkanInfo();
18
19         System.out.println(x:"\n=== Menambah Stok Produk ===");
20         System.out.println(x:"Menambah stok Benih Stroberyry AW55 sebanyak 20");
21         p1.tambahStok(20);
22         p1.tampilkanInfo();
23
24         System.out.println(x:"\n=== Mengurangi Stok Produk ===");
25         System.out.println(x:"Mengurangi stok Sekop Tangan sebanyak 10");
26         p3.kurangiStok(10);
27         p3.tampilkanInfo();
28
29         CreditBy.print();
30     }
31 }
```

```
System.out.println(x:"=== Info Awal Produk ===");
p1.tampilkanInfo();
p2.tampilkanInfo();
p3.tampilkanInfo();

System.out.println(x:"\n=== Menambah Stok Produk ===");
System.out.println(x:"Menambah stok Benih Stroberyry AW55 sebanyak 20");
p1.tambahStok(20);
p1.tampilkanInfo();

System.out.println(x:"\n=== Mengurangi Stok Produk ===");
System.out.println(x:"Mengurangi stok Sekop Tangan sebanyak 10");
p3.kurangiStok(10);
p3.tampilkanInfo();

CreditBy.print();
}
```

## F. HASIL EKSEKUSI

```
PS C:\Users\sri61\Documents\oop-pos-240202844\praktikum\week2-class-object\src\main\java> java
=== Info Awal Produk ===
Kode: MWA-002, Nama: Benih Strobery Aw55, Harga: 80000.0, Stok: 100
Kode: SSR-005, Nama: Pupuk Hayati 25kg, Harga: 90000.0, Stok: 100
Kode: SRW-025, Nama: Sekop Tangan, Harga: 50000.0, Stok: 100

=== Menambah Stok Produk ===
Menambah stok Benih Strobery Aw55 sebanyak 20
Berhasil menambah stok 20 unit. Stok sekarang: 120
Kode: MWA-002, Nama: Benih Strobery Aw55, Harga: 80000.0, Stok: 120

=== Mengurangi Stok Produk ===
Mengurangi stok Sekop Tangan sebanyak 10
Berhasil mengurangi stok 10 unit. Stok sekarang: 90
Kode: SRW-025, Nama: Sekop Tangan, Harga: 50000.0, Stok: 90

credit by: 240202844 - sriwaa
PS C:\Users\sri61\Documents\oop-pos-240202844\praktikum\week2-class-object\src\main\java>
```

## G. ANALISIS

1. Alur kode berjalan:
  - a) Program dimulai dari MainProduk, yang membuat beberapa objek dari class Produk.
  - b) Setiap objek memiliki atribut seperti kode, nama, harga, dan stok.
  - c) Informasi produk ditampilkan menggunakan method *getter* dari class Produk.
  - d) Di akhir, class CreditBy digunakan untuk menampilkan identitas mahasiswa sebagai tanda pembuat program.
2. Perbedaan dengan minggu sebelumnya:
  - Minggu sebelumnya hanya membuat program sederhana dengan satu file (tanpa class terpisah).
  - Minggu ini menggunakan konsep *class* dan *object* dengan struktur package (model, util, main), sehingga program lebih terorganisir. Diterapkan prinsip *enkapsulasi* untuk melindungi data.
3. Kendala dan cara mengatasinya:

Saat menjalankan perintah git push, muncul error:

```
PS C:\Users\sri61\Documents\oop-pos-240202844\praktikum\week2-class-object\src\main\java> git push
To https://github.com/sriwaa/oop-202501-240202844.git
! [rejected]        main -> main (fetch first)
error: failed to push some refs to 'https://github.com/sriwaa/oop-202501-240202844.git'
hint: Updates were rejected because the remote contains work that you do not
hint: have locally. This is usually caused by another repository pushing to
hint: the same ref. If you want to integrate the remote changes, use
hint: 'git pull' before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
```

Penyebab:

Repository di GitHub sudah memiliki update terbaru yang belum ada di lokal.

Solusi: Jalankan perintah ini untuk menarik perubahan terbaru terlebih dahulu sebelum melakukan push ulang.

```
PS C:\Users\sri61\Documents\oop-pos-240202844\praktikum\week2-class-object\src\main\java> git pull origin main --rebase
remote: Enumerating objects: 10, done.
remote: Counting objects: 100% (10/10), done.
remote: Compressing objects: 100% (6/6), done.
remote: Total 6 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (6/6), 41.73 KiB | 601.00 KiB/s, done.
From https://github.com/sriwaa/oop-202501-240202844
* branch                main      -> FETCH_HEAD
42365ef..a85c785  main      -> origin/main
Successfully rebased and updated refs/heads/main.
PS C:\Users\sri61\Documents\oop-pos-240202844\praktikum\week2-class-object\src\main\java> git add .
PS C:\Users\sri61\Documents\oop-pos-240202844\praktikum\week2-class-object\src\main\java> git commit -m "upload praktikum 2"
On branch main
Your branch is ahead of 'origin/main' by 2 commits.
(use "git push" to publish your local commits)

nothing to commit, working tree clean
PS C:\Users\sri61\Documents\oop-pos-240202844\praktikum\week2-class-object\src\main\java> git push origin main
Enumerating objects: 29, done.
Counting objects: 100% (29/29), done.
Delta compression using up to 2 threads
Compressing objects: 100% (12/12), done.
Writing objects: 100% (18/18), 1.86 KiB | 272.00 KiB/s, done.
Total 18 (delta 6), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (6/6), completed with 4 local objects.
To https://github.com/sriwaa/oop-202501-240202844.git
main 240202844 - sriwaa
```

## H. KESIMPULAN

1. Mahasiswa memahami konsep dasar *class* dan *object* dalam OOP. Penerapan *enkapsulasi* dengan *getter* dan *setter* membantu menjaga keamanan data.
2. Program menjadi lebih modular dan mudah dikembangkan di masa depan.
3. Struktur package membuat kode lebih terorganisir dan profesional.
4. Praktikum ini menjadi dasar penting untuk pengembangan aplikasi POS yang lebih kompleks.

## QUIZ

1. Mengapa atribut sebaiknya dideklarasikan sebagai *private* dalam class?

**Jawaban:**

Atribut sebaiknya dideklarasikan sebagai *private* agar data di dalam class terlindungi dan tidak bisa diakses langsung dari luar class. Hal ini merupakan prinsip *enkapsulasi* yang bertujuan menjaga keamanan data serta mencegah perubahan nilai atribut secara sembarangan.

2. Apa fungsi *getter* dan *setter* dalam enkapsulasi?

**Jawaban:**

Fungsi *getter* adalah untuk mengambil atau membaca nilai dari atribut yang bersifat *private*, sedangkan *setter* digunakan untuk mengubah atau menetapkan nilai atribut tersebut secara terkontrol. Dengan *getter* dan *setter*, kita bisa menerapkan logika validasi sebelum nilai atribut diubah.

3. Bagaimana cara class Produk mendukung pengembangan aplikasi POS yang lebih kompleks?

**Jawaban:**

Class Produk menjadi fondasi utama dalam aplikasi POS karena mewakili data barang yang dijual. Dengan struktur atribut seperti kode, nama, harga, dan stok, serta method seperti *tambahStok()* dan *kurangiStok()*, class ini bisa dikembangkan lebih lanjut untuk fitur transaksi, laporan penjualan, dan manajemen inventori secara terintegrasi.

## CHECKLIST KEBERHASILAN

- |  |
|--|
| <ul style="list-style-type: none"><li>✓ Class Produk berhasil dibuat dengan atribut dan method yang lengkap.</li><li>✓ Class <i>CreditBy</i> berhasil dibuat dan dipanggil di program utama.</li><li>✓ Objek produk berhasil diinstansiasi dan ditampilkan.</li><li>✓ Enkapsulasi sudah diterapkan dengan benar.</li><li>✓ Commit dengan pesan sesuai instruksi berhasil dilakukan.</li><li>✓ Screenshot hasil eksekusi telah dilampirkan.</li><li>✓ Laporan singkat telah dibuat.</li></ul> |
|--|