

**LAPORAN PRAKTIKUM
PEMOGRAMAN BERORIENTASI OBJEK**

**MODUL IV
“INHERITANCE & POLYMORPHISM”**



**Disusun oleh:
Laras Wahyu Adiningsih
21102214**

**Dosen Pengampu:
Dedy Agung Prabowo, S.Kom., M.Kom**

**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS INFORMATIKA
INSTITUT TEKNOLOGI TELKOM PURWOKERTO
PURWOKERTO
2023**

BAB I

TUJUAN PRAKTIKUM

- a. Mengerti dan memahami konsep Inheritance (Pewarisan)
- b. Mampu menerapkan Inheritance dalam pemrograman java

BAB II

DASAR TEORI

A. Abstraction

Sederhananya, **Abstraction** adalah menggunakan hal-hal sederhana untuk mewakili hal-hal rumit atau kompleks. **Abstraksi** adalah Menemukan hal-hal yang penting pada suatu objek dan mengabaikan hal-hal yang sifatnya insidental.

B. Enkapsulasi

Enkapsulasi (encapsulation) merupakan cara untuk melindungi **property** (atribut) / method tertentu dari sebuah kelas agar tidak sembarangan diakses dan dimodifikasi oleh suatu bagian program. Cara untuk melindungi data yaitu dengan menggunakan access modifiers (hak akses). Ada 4 hak akses yang tersedia, yaitu default, public, protected, private.

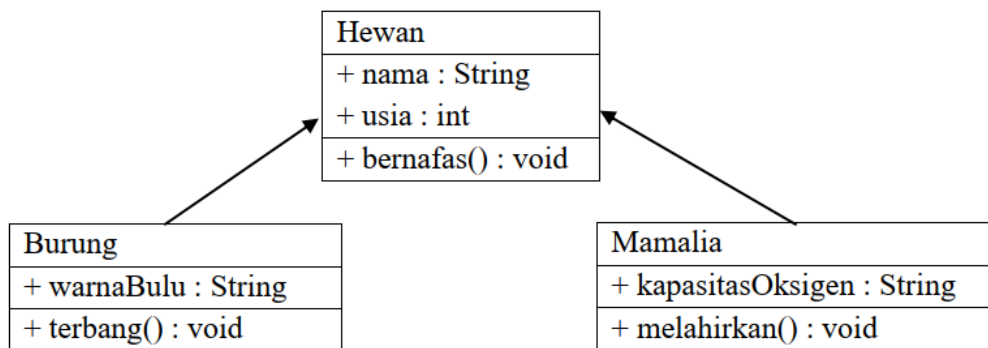
No	Modifier	Pada class dan interface	Pada method dan variabel
1	Default (tidak ada modifier)	Dapat diakses oleh yang sepaket	Diwarisi oleh subkelas dipaket yang sama, dapat diakses oleh method-method yang sepaket
2	Public	Dapat diakses dimanapun	Diwarisi oleh subkelasnya, dapat diakses dimanapun
3	Protected	Tidak bisa diterapkan	Diwarisi oleh subkelasnya, dapat diakses oleh method-method yang sepaket
4	private	Tidak bisa diterapkan	Tidak dapat diakses dimanapun kecuali oleh method-method yang ada dalam kelas itu sendiri

Aksesabilitas	public	private	protected	default
Dari kelas yang sama	Ya	Ya	Ya	Ya
Dari sembarang kelas dalam paket yang sama	Ya	Tidak	Ya	Ya
Dari sembarang kelas di luar paket	Ya	Tidak	Tidak	Tidak
Dari subkelas dalam paket yang sama	Ya	Tidak	Ya	Ya
Dari subkelas di luar paket	Ya	Tidak	Ya	Tidak

C. Inheritance

Konsep yang mendasari Inheritance adalah Generalization. Generalization digambarkan sebagai hubungan dari subclass ke superclass, sedangkan Inheritance adalah proses pewarisan data dan behaviour (method) dari superclass ke subclass. Contohnya, hewan adalah generalization dari singa, tumbuhan adalah generalization dari rumput. Sebaliknya, singa adalah inherits dari hewan dan rumput inherits dari tumbuhan.

Pewarisan (**Inheritance**) merupakan proses pembentukan kelas baru dari kelas yang sudah ada (**reusability**). Kelas yang mewariskan disebut Kelas Super (Kelas Induk), sedangkan kelas yang diwariskan disebut Sub Kelas (Kelas Anak). Pewarisan ini bersifat menyeluruh, sehingga semua data dan method yang dimiliki oleh kelas asalnya akan diturunkan kepada kelas baru.



D. Polymorphism

Polimorfisme dapat berarti “mempunyai banyak bentuk” sehingga dapat disimpulkan, **Polimorfisme** adalah kemampuan untuk meminta objek yang berbeda untuk melaksanakan tugas yang sama dan membuat objek tahu bagaimana untuk mencapainya dengan caranya sendiri. **Polimorfisme** menunjukkan kemampuan untuk menangani dua atau lebih bentuk obyek yang berlainan saat eksekusi berlangsung.

Polimorfisme dapat diilustrasikan sebagai berikut, perhatikanlah penggunaan kata “mentah” dalam beberapa kalimat. “Sayuran itu masih mentah, belum dimasak”, “Pukulan petinju itu berhasil dimentahkan oleh lawannya”, “Gagasan ini masih mentah sehingga perlu di bahas kembali”. Kata “mentah” pada contoh di atas dapat diaplikasikan pada berbagai objek dan dapat di-interpretasikan ke dalam beberapa makna.

BAB III

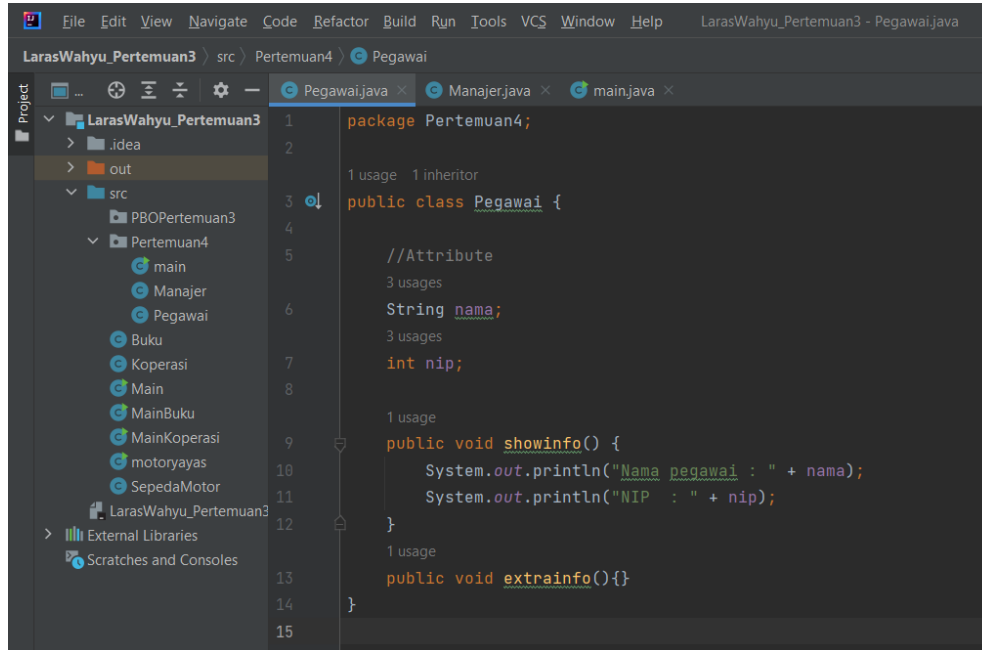
GUIDED

1. Guided 1

Uji coba Inheritance

a. Buat class baru dengan nama “**Pegawai**” kemudian masukkan script berikut :

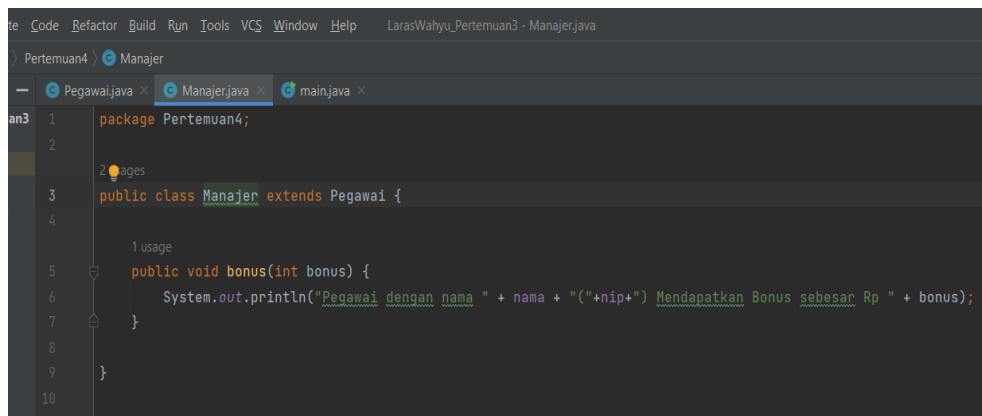
Source code



```
1 package Pertemuan4;
2
3 public class Pegawai {
4
5     //Attribute
6     String nama;
7     int nip;
8
9     public void showinfo() {
10         System.out.println("Nama pegawai : " + nama);
11         System.out.println("NIP : " + nip);
12     }
13
14     public void extrainfo(){}
15 }
```

b. Setelah itu, buat class baru dengan nama “**Manajer**”

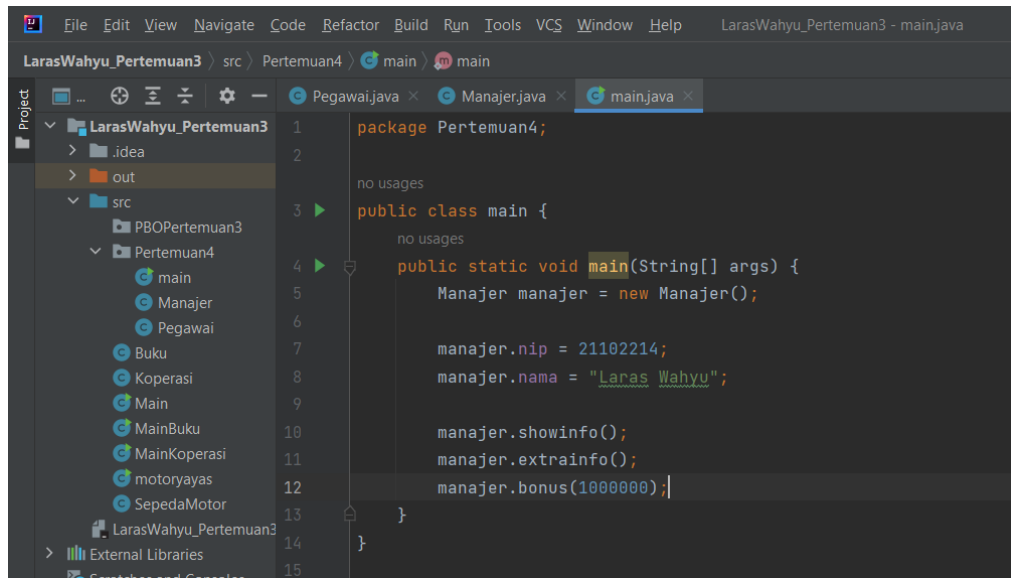
Source code



```
1 package Pertemuan4;
2
3 public class Manajer extends Pegawai {
4
5     public void bonus(int bonus) {
6         System.out.println("Pegawai dengan nama " + nama + "(" + nip + ") Mendapatkan Bonus sebesar Rp " + bonus);
7     }
8
9 }
10
```

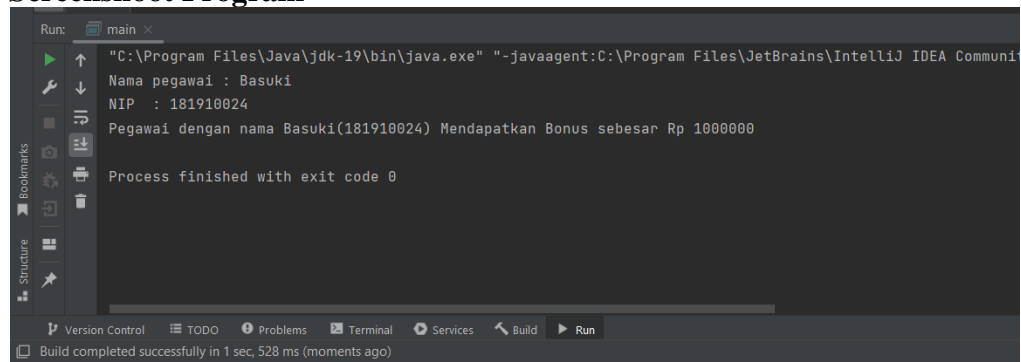
- c. Agar program dapat dijalankan, sekarang buatlah java class baru dengan nama “Main” lalu ketik script berikut :

Source code



```
1 package Pertemuan4;
2
3 public class main {
4     no usages
5     public static void main(String[] args) {
6         Manajer manajer = new Manajer();
7
8         manajer.nip = 21102214;
9         manajer.nama = "Laras Wahyu";
10
11         manajer.showinfo();
12         manajer.extrainfo();
13         manajer.bonus(1000000);
14     }
15 }
```

Screenshoot Program

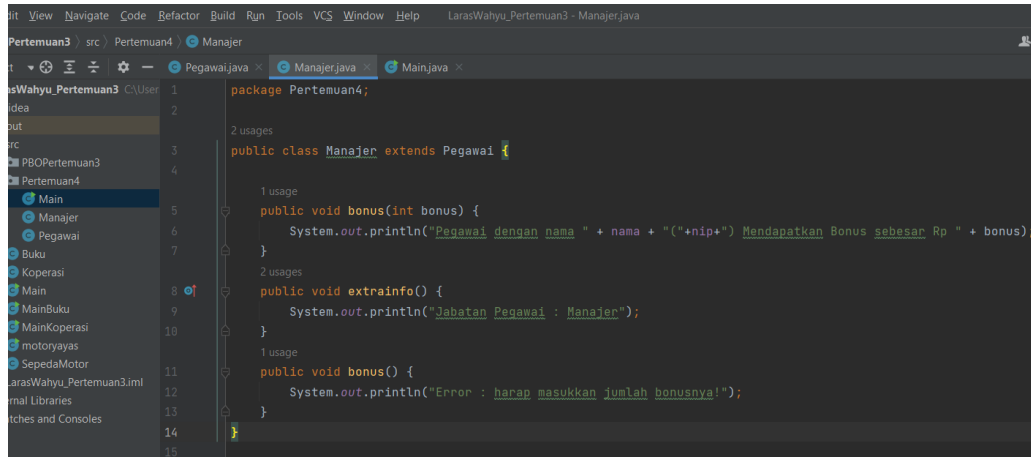


```
Run: main x
"\"C:\Program Files\Java\jdk-19\bin\java.exe\" \"-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Communi
Nama pegawai : Basuki
NIP : 181910024
Pegawai dengan nama Basuki(181910024) Mendapatkan Bonus sebesar Rp 1000000
Process finished with exit code 0
Build completed successfully in 1 sec, 528 ms (moments ago)
```

Uji Coba Polymorphism

a. Modifikasi class “**Manajer**”, tambahkan script berikut :

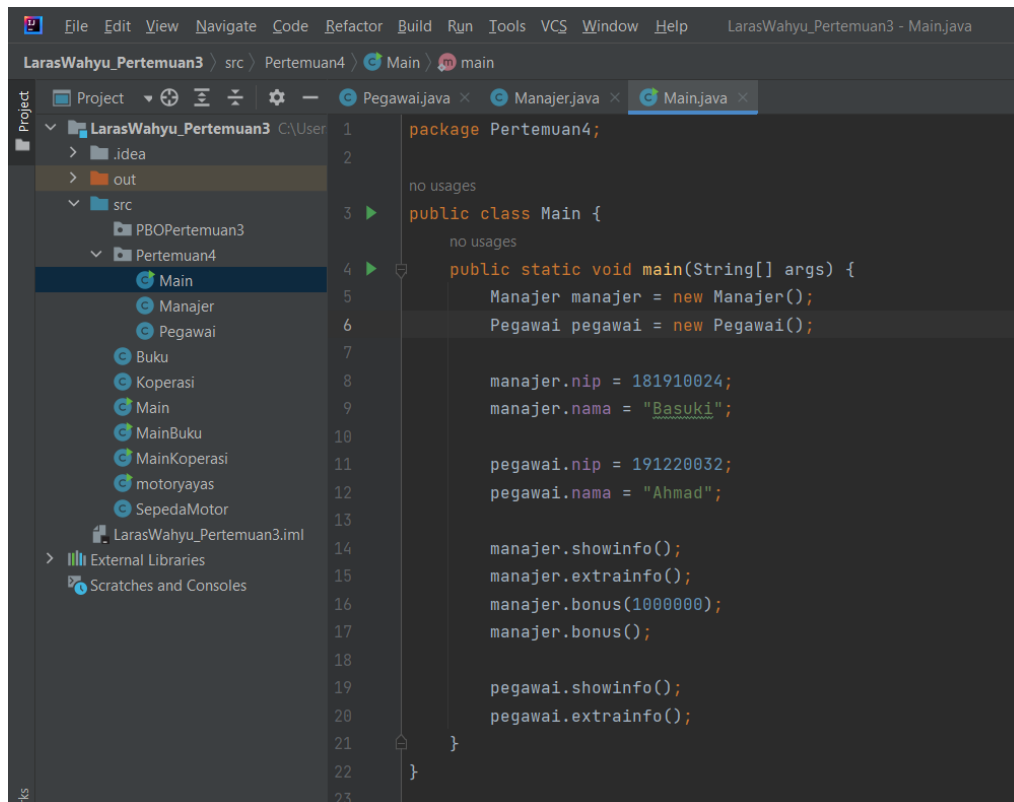
Source code



```
1 package Pertemuan4;
2
3 public class Manajer extends Pegawai {
4
5     1 usage
6     public void bonus(int bonus) {
7         System.out.println("Pegawai dengan nama " + nama + "(" + nip + ") Mendapatkan Bonus sebesar Rp " + bonus);
8     }
9     2 usages
10    public void extrainfo() {
11        System.out.println("Jabatan Pegawai : Manajer");
12    }
13    1 usage
14    public void bonus() {
15        System.out.println("Error : harap masukkan jumlah bonusnya!");
16    }
17 }
```

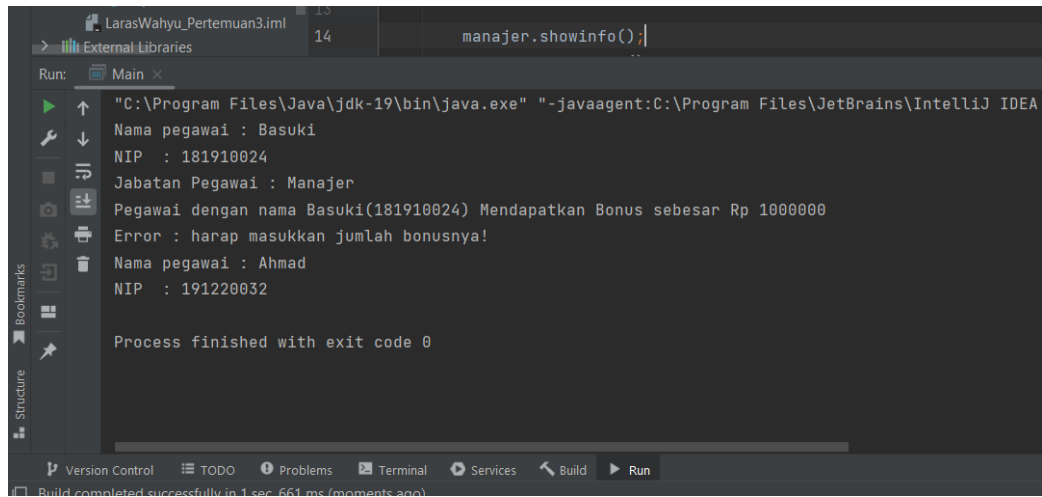
b. Modifikasi class “**Main**” dengan script berikut :

Source code



```
1 package Pertemuan4;
2
3 no usages
4 public class Main {
5     no usages
6     public static void main(String[] args) {
7
8         Manajer manajer = new Manajer();
9         Pegawai pegawai = new Pegawai();
10
11         manajer.nip = 181910024;
12         manajer.nama = "Basuki";
13
14         pegawai.nip = 191220032;
15         pegawai.nama = "Ahmad";
16
17         manajer.showinfo();
18         manajer.extrainfo();
19         manajer.bonus(1000000);
20         manajer.bonus();
21
22         pegawai.showinfo();
23         pegawai.extrainfo();
24     }
25 }
```

Screenshoot Program



```
manajer.showinfo();
```

Run: Main

```
"C:\Program Files\Java\jdk-19\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA
Nama pegawai : Basuki
NIP : 181910024
Jabatan Pegawai : Manajer
Pegawai dengan nama Basuki(181910024) Mendapatkan Bonus sebesar Rp 1000000
Error : harap masukkan jumlah bonusnya!
Nama pegawai : Ahmad
NIP : 191220032

Process finished with exit code 0
```

Build completed successfully in 1 sec 661 ms (moments ago)

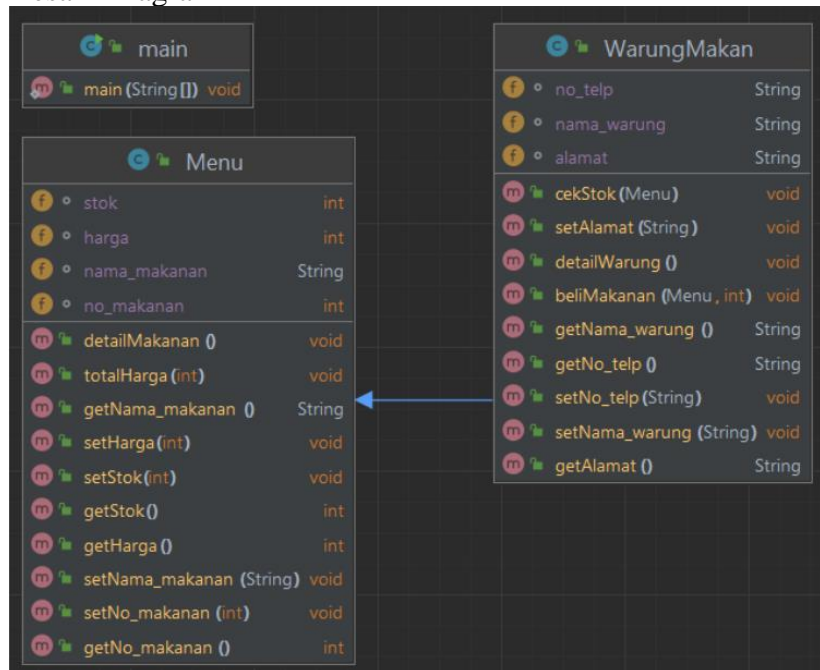
UNGUIDED

1. Unguided 1

Buatlah sebuah program warung makan sederhana dengan ketentuan sebagai berikut
Item yang dijual :

No Makanan	Nama Makanan	Harga	Stock
1	Nasi Goreng	10.000	10
2	Nasi Goreng Spesial	15.000	4
3	Nasi Goreng Spesial + Telur	20.000	20

Desain Diagram



Dengan ketentuan sebagai berikut pada Class WarungMakan.java :

- Terdapat method getter dan setter
- **Method detailWarung()** menampilkan detail seperti alamat nama warung dan no telp
- **Method beliMakanan()** menampilkan detail pembelian makanan dan melakukan pengecekan stock pada menu jika habis maka akan menampilkan stock tidak cukup dan jika stock tersedia, stock dikurangi jumlah makanan yang dibeli
- **Method cekStock()** menampilkan jumlah stock terkini setelah dilakukan pembelian dari menu yang dipilih
- Class WarungMakan extends class Menu

Dengan ketentuan sebagai berikut pada Class Makanan.java :

- Terdapat method getter dan setter
- **Method detailMakanan()** menampilkan detail seperti no makanan, nama makanan, harga dan stock
- **Method totalHarga()** menampilkan total harga dari hasil jumlah pembelian makanan

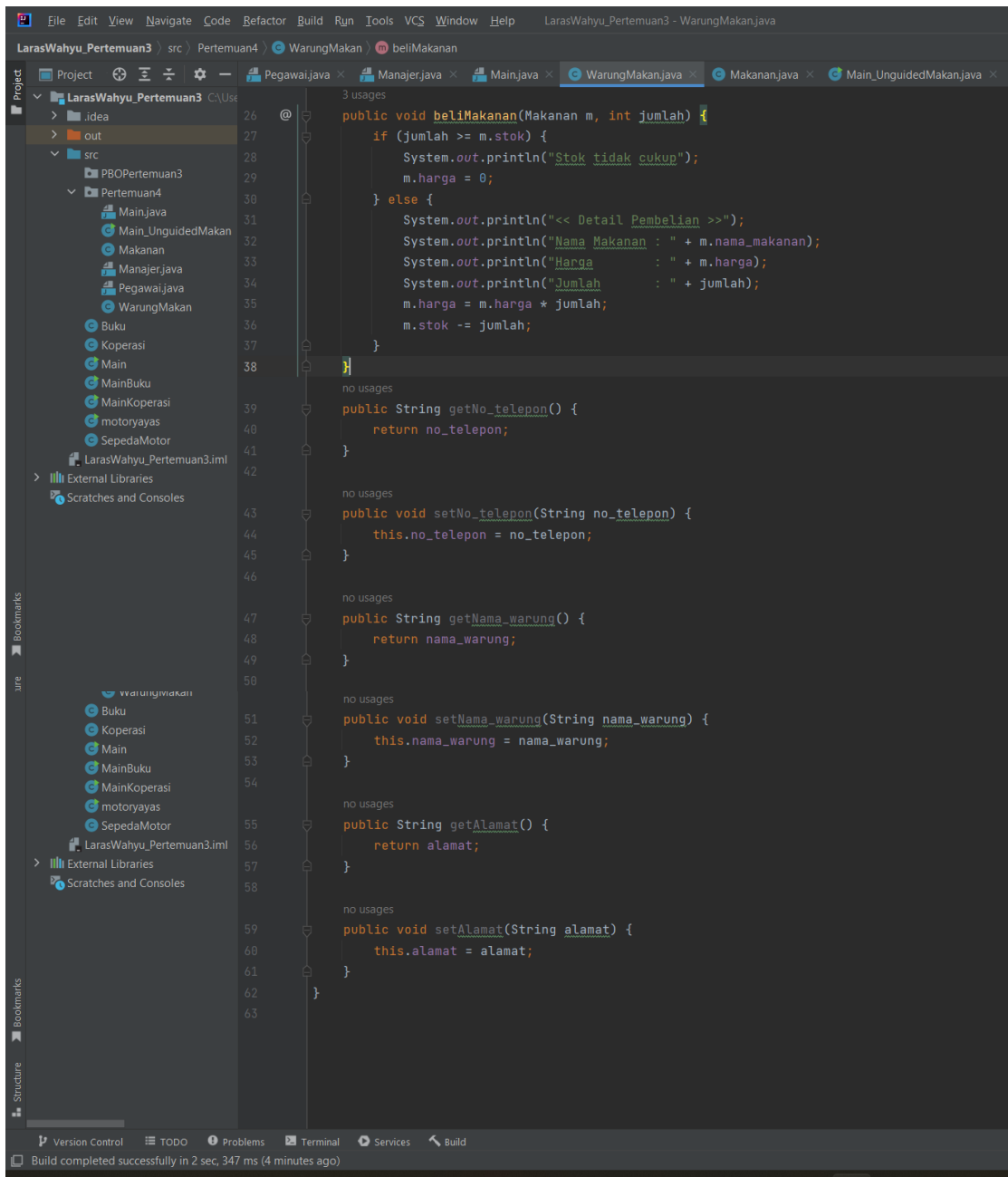
Buatlah Class Main.java di dalamnya dapat mengisi semua isian di Class WarungMakan dan dapat mengisi Menu makanan seperti detail tabel item diatas, kemudian tampilkan detail warung dan makanan yang telah diisi Lakukan operasi pembelian dengan method yang telah dibuat sebelumnya

Source code

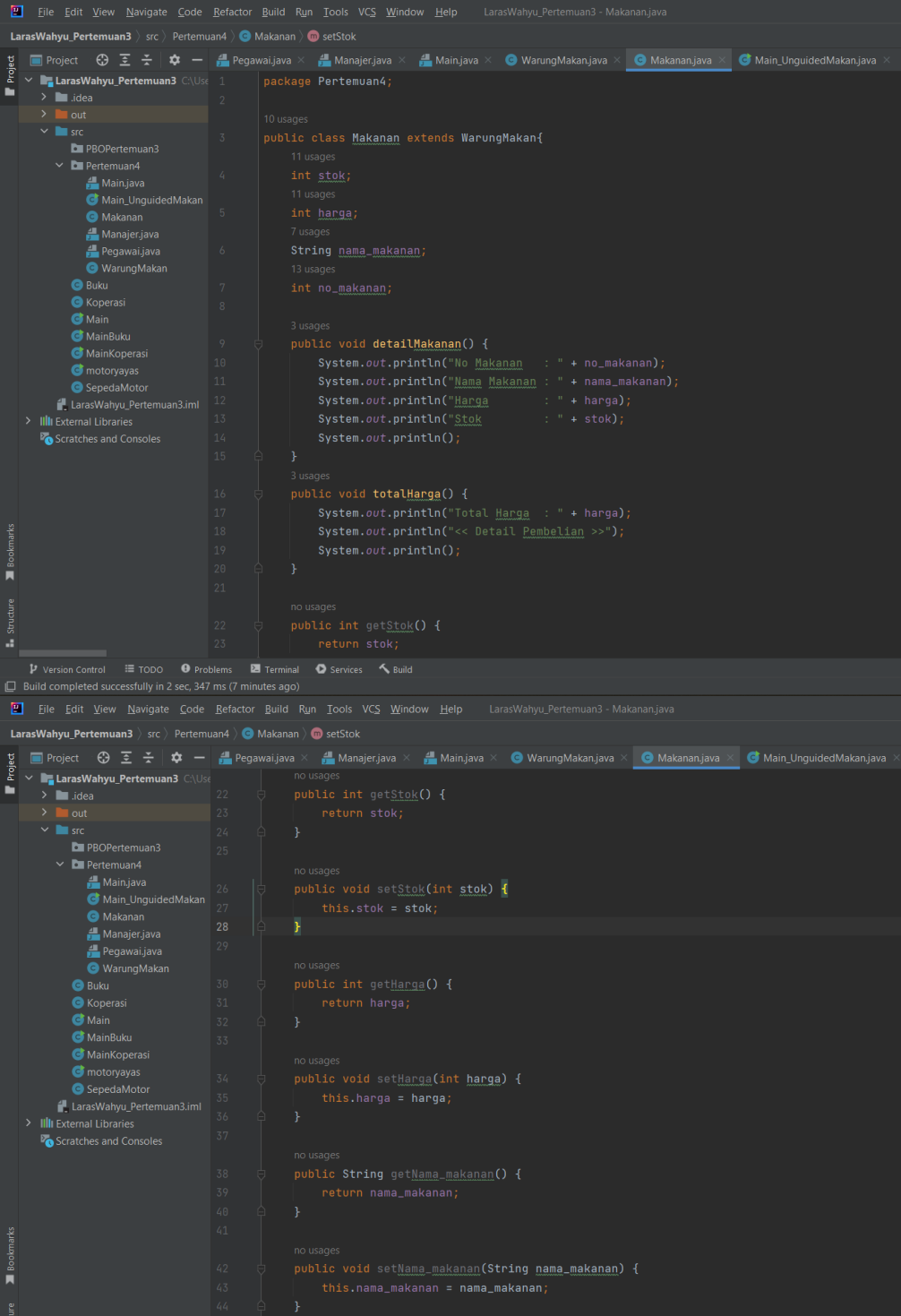
WarungMakan.Java

```

1 package Pertemuan4;
2
3 import javax.swing.*;
4
5 public class WarungMakan {
6     String no_telepon;
7     String nama_warung;
8     String alamat;
9
10    public void detailWarung() {
11        System.out.println("<< WARUNG MAKAN SEDERHANA >>");
12        System.out.println("Nama Warung : " + nama_warung);
13        System.out.println("Alamat : " + alamat);
14        System.out.println("No Telp : " + no_telepon);
15        System.out.println();
16    }
17
18    public void cekStok(Makanan m) {
19        if (m.no_makanan == 1) {
20            System.out.println("Stok Nasi Goreng : " + m.stok);
21        } else if (m.no_makanan == 2) {
22            System.out.println("Stok Nasi Goreng Spesial : " + m.stok);
23        } else if (m.no_makanan == 3) {
24            System.out.println("Stok Nasi Goreng Spesial + Telur : " + m.stok);
25        }
26        System.out.println();
27    }
28 }
  
```



Source code Makanan.Java

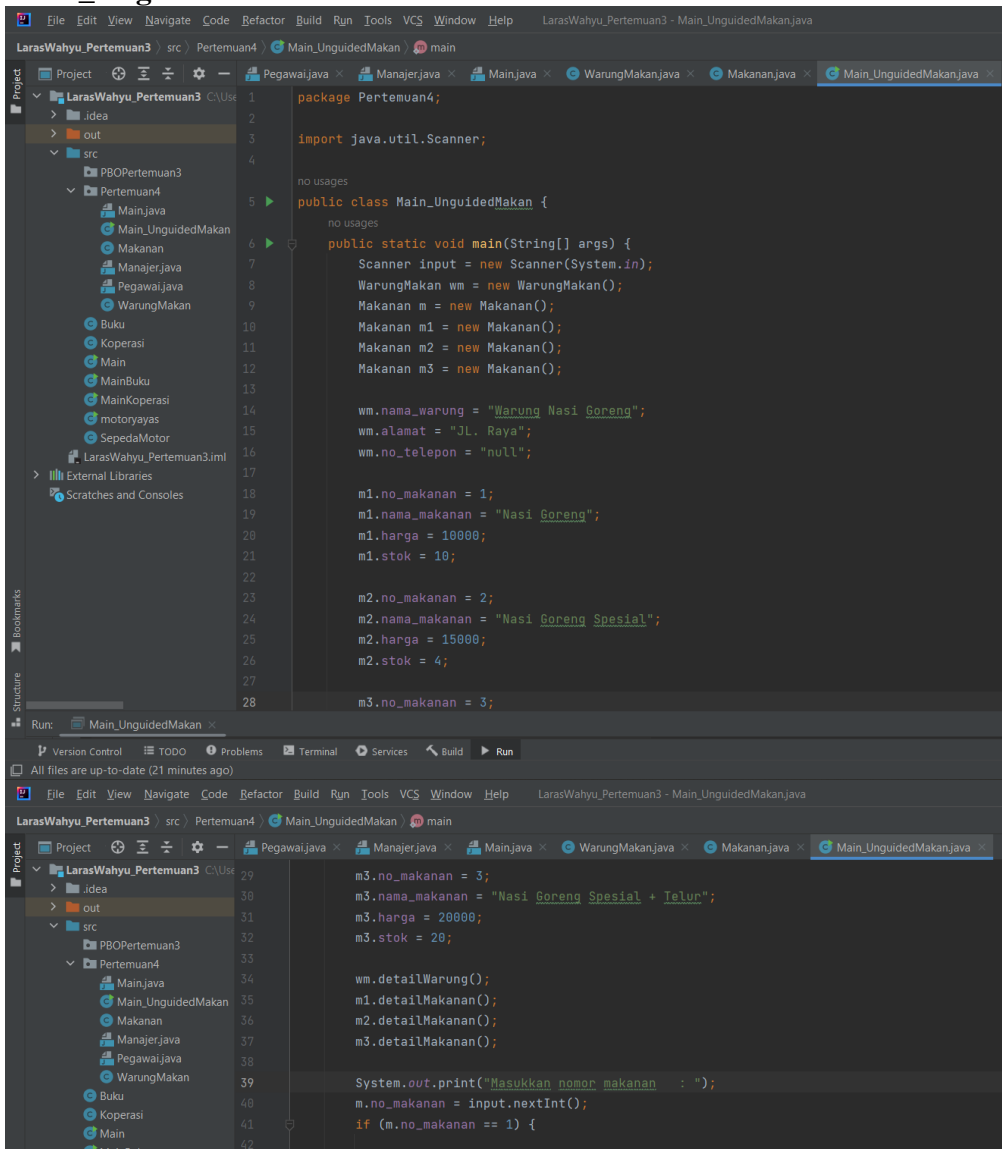


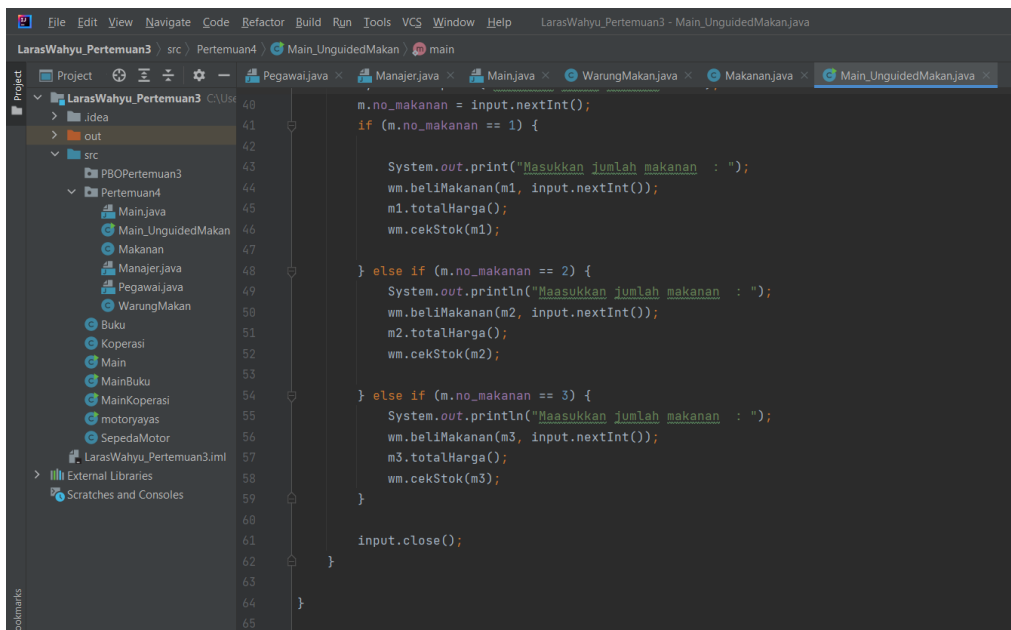
```
1 package Pertemuan4;
2
3 10 usages
4 public class Makanan extends WarungMakan{
5     11 usages
6     int stok;
7     11 usages
8     int harga;
9     7 usages
10    String nama_makanan;
11    13 usages
12    int no_makanan;
13
14    3 usages
15    public void detailMakanan() {
16        System.out.println("No Makanan : " + no_makanan);
17        System.out.println("Nama Makanan : " + nama_makanan);
18        System.out.println("Harga : " + harga);
19        System.out.println("Stok : " + stok);
20        System.out.println();
21    }
22
23    3 usages
24    public void totalHarga() {
25        System.out.println("Total Harga : " + harga);
26        System.out.println("<< Detail Pembelian >>");
27        System.out.println();
28    }
29
30    no usages
31    public int getStok() {
32        return stok;
33    }
34
35    no usages
36    public void setStok(int stok) {
37        this.stok = stok;
38    }
39
40    no usages
41    public int getHarga() {
42        return harga;
43    }
44
45    no usages
46    public void setHarga(int harga) {
47        this.harga = harga;
48    }
49
50    no usages
51    public String getNama_makanan() {
52        return nama_makanan;
53    }
54
55    no usages
56    public void setNama_makanan(String nama_makanan) {
57        this.nama_makanan = nama_makanan;
58    }
59 }
```



Source code

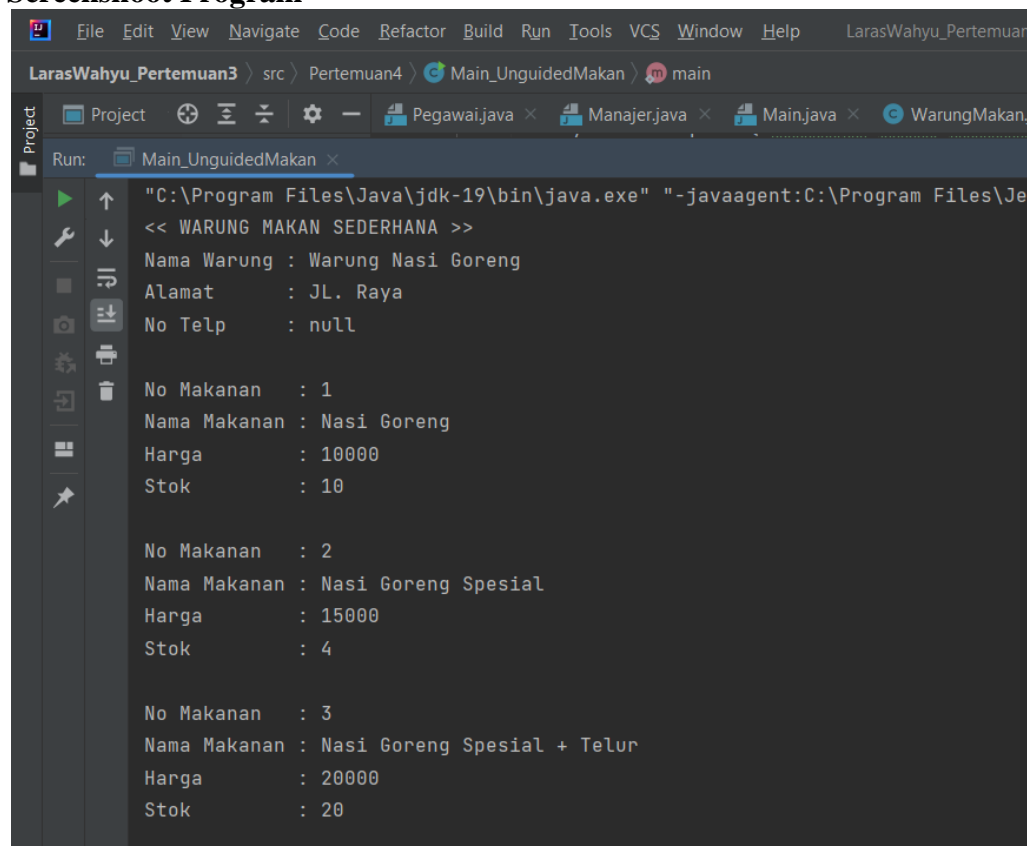
Main_UnguidedMakan.Java





```
40 m.no_makanan = input.nextInt();
41
42 if (m.no_makanan == 1) {
43
44     System.out.print("Masukkan jumlah makanan : ");
45     wm.beliMakanan(m1, input.nextInt());
46     m1.totalHarga();
47     wm.cekStok(m1);
48 }
49
50 } else if (m.no_makanan == 2) {
51     System.out.println("Masukkan jumlah makanan : ");
52     wm.beliMakanan(m2, input.nextInt());
53     m2.totalHarga();
54     wm.cekStok(m2);
55 }
56
57 } else if (m.no_makanan == 3) {
58     System.out.println("Masukkan jumlah makanan : ");
59     wm.beliMakanan(m3, input.nextInt());
60     m3.totalHarga();
61     wm.cekStok(m3);
62 }
63
64 input.close();
65 }
```

Screenshoot Program



```
Run: C:\Program Files\Java\jdk-19\bin\java.exe "-javaagent:C:\Program Files\Je
<< WARUNG MAKAN SEDERHANA >>
Nama Warung : Warung Nasi Goreng
Alamat      : JL. Raya
No Telp     : null

No Makanan  : 1
Nama Makanan : Nasi Goreng
Harga       : 10000
Stok        : 10

No Makanan  : 2
Nama Makanan : Nasi Goreng Spesial
Harga       : 15000
Stok        : 4

No Makanan  : 3
Nama Makanan : Nasi Goreng Spesial + Telur
Harga       : 20000
Stok        : 20
```

User dapat memilih makanan sesuai dengan nomor makanan

```
Run: Main_UnguidedMakan x
"C:\Program Files\Java\jdk-19\bin\java.exe" "-javaagent:C:\Program Files\Je
<< WARUNG MAKAN SEDERHANA >>
Nama Warung : Warung Nasi Goreng
Alamat      : JL. Raya
No Telp     : null

No Makanan  : 1
Nama Makanan : Nasi Goreng
Harga       : 10000
Stok        : 10

No Makanan  : 2
Nama Makanan : Nasi Goreng Spesial
Harga       : 15000
Stok        : 4

No Makanan  : 3
Nama Makanan : Nasi Goreng Spesial + Telur
Harga       : 20000
Stok        : 20

Masukkan nomor makanan : 1
Masukkan jumlah makanan : 2
<< Detail Pembelian >>
Nama Makanan : Nasi Goreng
Harga        : 10000
Jumlah       : 2
Total Harga  : 20000
<< Detail Pembelian >>

Stok Nasi Goreng : 8

Process finished with exit code 0
```

User memasukan jumlah makanan dan terlihat detail pembelian dan sisa stock setelah pembelian

BAB IV

KESIMPULAN

Kesimpulan pada Praktikum ini adalah kita belajar tentang Abstraksi, Enkapsulasi, Inheritance, dan Polimorfisme. Inheritance merupakan konsep OOP di mana kita bisa membentuk class baru yang “mewarisi” ataupun mempunyai bagian- bagian dari class yang telah terdapat tadinya. Konsep ini memakai sistem hirarki maupun bertingkat. Semacam suatu Drop- Down Menu yang terdapat di mayoritas web, di mana terus menjadi khusus submenunya, terus menjadi khusus pula kontennya. Demikian pula dengan Inheritance OOP di mana terus menjadi khusus subclassnya, terus menjadi sedikit pula komponen yang bisa diwarisi class tersebut.

Polymorphism merupakan konsep di mana sesuatu objek yang berbeda- beda bisa diakses lewat interface yang sama. Suatu objek yang polymorphic bisa menyesuaikan diri dengan tata cara apapun yang diimplementasikan pada objek tersebut, serta setiap class mempunyai interpretasinya tertentu terhadap interfacenya. Dalam Java, ada 2 tipe polymorphism adalah Static Polymorphism serta Dynamic Polymorphism. Static Polymorphism yang universal digunakan merupakan Method Overloading. Method Overloading mengizinkan kamu buat mempraktikkan sebagian implementasi tata cara yang berbeda dalam kelas yang sama tetapi dengan parameter yang berbeda- beda. Dalam Dynamic Polymorphism suatu subclass bisa mengenai tata cara dari superclassnya. Bila kamu mempraktikkan subclass tersebut, Java Virtual Machine hendak senantiasa memakai tata cara yang telah ditimpa.