



NAMA : Karmila Novi Arfiana

NIM : 2041720073

KELAS : TI-2C

ABSEN : 11

2. 1 Percobaan 1

Pertanyaan :

1. Class apa sajakah yang merupakan turunan dari class **Employee**?

Jawab : yang merupakan class turunan class employee adalah class **internShipEmployee** dan class **PermanentEmployee**

2. Class apa sajakah yang implements ke interface **Payable**?

Jawab : class **PermanentEmployee** dan **ElectricityBill**

3. Perhatikan class **Tester1**, baris ke-10 dan 11. Mengapa **e**, bisa diisi dengan objek **pEmp** (merupakan objek dari class **PermanentEmployee**) dan objek **iEmp** (merupakan objek dari class **InternshipEmployee**) ?

Jawab : karena class **permanentEmployee** dan **InternshipEmployee** merupakan turunan dari class **employee** sehingga terhubung

4. Perhatikan class **Tester1**, baris ke-12 dan 13. Mengapa **p**, bisa diisi dengan objek **pEmp** (merupakan objek dari class **PermanentEmployee**) dan objek **eBill** (merupakan objek dari class **ElectricityBill**) ?

Jawab : karena class **permanentEmployee** dan **electricityBill** mengimplementasikan class interface **payable** sehingga terhubung

5. Coba tambahkan sintaks: **p = iEmp;**
e = eBill;

pada baris 14 dan 15 (baris terakhir dalam method **main**) ! Apa yang menyebabkan error?

Jawab : terjadi error karena class **InterShipEmployee** tidak mengimplementasikan class interface **payable** dan class **electricityBill** tidak mengextends class **Employee**

6. Ambil kesimpulan tentang konsep/bentuk dasar polimorfisme!

Jawab : Polimorfisme memiliki sebuah konsep di mana *class* memiliki banyak “bentuk” *method* yang berbeda, meskipun namanya sama. Maksud dari “bentuk” adalah isinya yang berbeda, namun tipe data dan parameternya berbeda



NAMA : Karmila Novi Arfiana

NIM : 2041720073

KELAS : TI-2C

ABSEN : 11

2.2 Percobaan 2

Pertanyaan :

1. Perhatikan class **Tester2** di atas, mengapa pemanggilan **e.getEmployeeInfo()** pada baris 8 dan **pEmp.getEmployeeInfo()** pada baris 10 menghasilkan hasil sama?

Jawab : karena **permanentEmployee** adalah turunan dari class **employee** maka pada baris 8 dan 10 sama-sama memanggil method **info** yang ada di class **permanentEmployee**. Dan pada baris 8 menggunakan pemanggilan method virtual

2. Mengapa pemanggilan method **e.getEmployeeInfo()** disebut sebagai pemanggilan method virtual (virtual method invocation), sedangkan **pEmp.getEmployeeInfo()** tidak?

Jawab : karena pada class **Employee** diinisialisasikan dengan **e**, dan **e = pEmp** yang dimana **Employee** memanggil instansiasi **pEmp** dari objek **PermanentEmployee**. Sedangkan **pEmp.getEmployeeInfo** inisialisai objeknya langsung memanggil methodnya sendiri

3. Jadi apakah yang dimaksud dari virtual method invocation? Mengapa disebut virtual?

Jawab : Virtual method invocation terjadi ketika ada pemanggilan overriding method dari suatu objek polimorfisme. Disebut virtual karena antara method yang dikenali oleh compiler dan method yang dijalankan oleh JVM berbeda



NAMA : Karmila Novi Arfiana

NIM : 2041720073

KELAS : TI-2C

ABSEN : 11

2.3 Percobaan 3

Pertanyaan :

1. Perhatikan array **e** pada baris ke-8, mengapa ia bisa diisi dengan objek objek dengan tipe yang berbeda, yaitu objek **pEmp** (objek dari **PermanentEmployee**) dan objek **iEmp** (objek dari **InternshipEmployee**) ?

Jawab : karena objek **pEmp** (objek dari **PermanentEmployee**) dan objek **iEmp** (objek dari **InternshipEmployee**) merupakan class turunan dari class **employee** (objek **e**)

2. Perhatikan juga baris ke-9, mengapa array **p** juga diisi dengan objek-objek dengan tipe yang berbeda, yaitu objek **pEmp** (objek dari **PermanentEmployee**) dan objek **eBill** (objek dari **ElectricityBilling**) ?

Jawab : karena pada array **p** adalah interface **Payable**, dimana objek **pEmp** (objek dari **PermanentEmployee**) dan pada objek **eBill** (objek dari **ElectricityBilling**) telah mengimplementasikan class interface **Payable**.

3. Perhatikan baris ke-10, mengapa terjadi error?

Jawab : karena **eBill** (objek dari **ElectricityBill**) tidak mengextendskan class **Employee**, jadi saat **Employee** mau memanggil **ebill** terjadi error, dan class **ElectricityBill** harus extends **Employee**



NAMA : Karmila Novi Arfiana

NIM : 2041720073

KELAS : TI-2C

ABSEN : 11

2.4 Percobaan 4

Pertanyaan :

1. Perhatikan class Tester4 baris ke-7 dan baris ke-11, mengapa pemanggilan ow.pay(eBill) dan ow.pay(pEmp) bisa dilakukan, padahal jika diperhatikan method pay() yang ada di dalam class Owner memiliki argument/parameter bertipe Payable? Jika diperhatikan lebih detil eBill merupakan objek dari ElectricityBill dan pEmp merupakan objek dari PermanentEmployee?

Jawab : Pemanggilan ow.pay(eBill) dan ow.pay(pEmp) dapat dilakukan karena Class dari kedua objek tersebut merupakan implementasi dari Class Interface Payment

2. Jadi apakah tujuan membuat argument bertipe Payable pada method pay() yang ada di dalam class Owner?

Jawab : untuk menginstansiasi class payable kedalam method pay

3. Coba pada baris terakhir method main() yang ada di dalam class Tester4 ditambahkan perintah ow.pay(iEmp);

```
3 public class Tester4 {
4     public static void main(String[] args) {
5         Owner ow = new Owner();
6         ElectricityBill eBill = new ElectricityBill(5, "R-1");
7         ow.pay(eBill); //pay for electricity bill
8         System.out.println("-----");
9
10        PermanentEmployee pEmp = new PermanentEmployee("Dedik", 500);
11        ow.pay(pEmp); //pay for permanent employee
12        System.out.println("-----");
13
14        InternshipEmployee iEmp = new InternshipEmployee("Sunarto", 5);
15        ow.showMyEmployee(pEmp); //show permanent employee info
16        System.out.println("-----");
17        ow.showMyEmployee(iEmp); //show internship employee info
18
19        ow.pay(iEmp);
20    }
21 }
```

Mengapa terjadi error?

Jawab : Terjadi error karena ow.pay(iEmp) objek dari class ini tidak mengimplementasikan Class Interface Payable dan karena pada class interShipEmployee tidak didefinisikan pada method pay (tidak ada di instanceof)

4. Perhatikan class Owner, diperlukan untuk apakah sintaks p instanceof ElectricityBill pada baris ke-6 ?

Jawab : untuk mendeklarasikan/menginstansiasi Method yang ada didalam sebuah Class ElectricityBill

5. Perhatikan kembali class Owner baris ke-7, untuk apakah casting objek disana (ElectricityBill eb = (ElectricityBill) p) diperlukan ? Mengapa objek p yang bertipe Payable harus di-casting ke dalam objek eb yang bertipe ElectricityBill ?

Jawab : Karena downcasting yang memanggil kelas turunan dari kelas induk



NAMA : Karmila Novi Arfiana
NIM : 2041720073
KELAS : TI-2C
ABSEN : 11

Hasil Tugas

```
run:
Walking Zombie Data =
Health = 100
Level = 1

Jumping Zombie Data =
Health = 100
Level = 2

Barrier Strenght = 100

Walking Zombie Data =
Health = 42
Level = 1

Jumping Zombie Data =
Health = 66
Level = 2

Barrier Strenght = 64

BUILD SUCCESSFUL (total time: 0 seconds)
```