NAMA : YUDAS MALABI
NIM : 2041720054
KELAS : TI 2C
MATERI : PRAKTIKUM PBO

4.2 Pertanyaan

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

   Jawab :

   Yang merupakan class turunan Employee adalah class InternshipEmployee dan class PermanentEmployee

2. Class apa sajakah yang implements ke interface Payable?

   Jawab :

   Class PermanenEmployee dan class 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 InternshipEmploye) ?

   Jawab :

   Karena e merupakan inisialisasi dari class Employee, dimana class Employee merupakan parent class dari PermanentEmployee (pEmp) dan juga class InternshipEmployee (iEmp)

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

   Jawab :

   Karena pada class PermanentEmployee dan class Electricity mengimplements suatu interface yang bernama Payable.

5. Coba tambahkan sintaks: p = iEmp; e = eBill; pada baris 14 dan 15 (baris terakhir dalam method main) ! Apa yang menyebabkan error?

   Jawab :

   Error karena pada class InternshipEmployee (iEmp) tidak mengimplements interface Payable, dan class Electricity tidak mengekstends class Employee

6. Ambil kesimpulan tentang konsep/bentuk dasar polimorfisme!

   Jawab :

NAMA : YUDAS MALABI
NIM : 2041720054
KELAS : TI 2C
MATERI : PRAKTIKUM PBO

Kesimpulannya yaitu bahwa setiap objek atau class memiliki bentuk method yang berbeda meskipun namanya sama

5.2 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 objek Employee merupakan parent class dari objek PermanentEmployee, sehingga ketika di inisiasikan e = pEmp, hasil outputnya akan sama saja.

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

Jawab :

Karena terdapat metode overriding , dimana getEmployeeInfo() pada class Employee di override pada subclass PermanentEmployee

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

Jawab :

Virtual method invocation adalah metode yang terjadi karena ada kondisi overriding method dari suatu objek polimorfisme. Disebut virtual karena antara method yang dikenali oleh compiler dan method yang dijalankan oleh JVM berbeda.

NAMA : YUDAS MALABI
NIM : 2041720054
KELAS : TI 2C
MATERI : PRAKTIKUM PBO

6.2 Pertanyaan

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

Jawab :

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

Jawab :

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

Jawab :

Pada baris ke 10 terjadi error karena pada class ElectricityBill tidak mengimplemen kelas Employee sehingga pada saat ingin di convert atau di casting pada objek Employee, objek ElectricityBill akan error

NAMA : YUDAS MALABI
NIM : 2041720054
KELAS : TI 2C
MATERI : PRAKTIKUM PBO

7.2 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) bisa dilakukan karena pada class ElectricityBill dan juga pada class PermanentEmployee mengimplements suatu interface Payable

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

Jawab :

Tujuannya yaitu agar pada method pay tersebut dapat dipassing parameter berupa class yang telah mengimplementasikan interface Payable

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

Jawab :

Terjadi error karena pada class InternshipEmployee tidak melakukan implements terhadap interface Payable.

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

Jawab :

Diperlukan untuk melakukan pengecekan tipe objek yg di passing pada parameter, karena terdapat perbedaan bentuk dan konsep walaupun sama-sama mengimplements interface Payable

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 pada parameter method pay bertipe objek Payable , dimana Payable itu interface nya, sehingga untuk bisa mengakses ke objek spesifiknya harus dilakukan casting. Ketika sudah dilakukan casting baru kita bisa mengakses method pada objek yg di casting tersebut.