Nama: Rara Deninda Hurianto

NIM: 2041720110

JOBSHEET 11

PERCOBAAN 1

Pertanyaan

1. Class apa sajakah yang merupakan turunan dari class Employee?
   Jawab:
   Ada 2 yaitu: class IntershipEmployee dan class PermanentEmployee

2. Class apa sajakah yang implements ke interface Payable?
   Jawab:
   Ada 2 yaitu: class PermanentEmployee 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 dari Employee merupakan parent class dari pEmp (PermanentEmployee) dan iEmp(IntershipEmployee)

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 p dari Payable merupakan interface dari pEmp(PermanentEmployee) dan eBill (ElectricityBill).

5. Coba tambahkan sintaks:
   p = iEmp;
   e = eBill;
   pada baris 14 dan 15 (baris terakhir dalam method main) ! Apa yang menyebabkan error?
   Jawab:
   P = iEmp error karena intershipEmployee tidak terhubung ke interface Payable, dan
   E = eBill error karena ElectricBill bukan class child dari Employee.

6. Ambil kesimpulan tentang konsep/bentuk dasar polimorfisme!
   Jawab:
   Polimorfisme adalah konsep dimana suatu objek dapat memiliki berbagai macam bentuk atau kondisi dan dapat diakses oleh interface yang sama.

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:
   Method getEmployeeInfo() yang dipanggil pEmp sama dengan e karena pEmp merupakan turunan dari class Employee

2. Mengapa pemanggilan method e.getEmployeeInfo() disebut sebagai pemanggilan method virtual (virtual method invication), sedangkan pEmp.getEmployeeInfo() tidak?
   Jawab:
   Karena method e.getEmployeeInfo() melakukan pemanggilan overriding method dari objek polimorfisme yaitu e dari Employee. Karena e = pEmp maka saat e.getEmplloyeeInfo() dijalankan oleh JVM maka method getEmployeeInfo() dari class PermanentEmployee lah yang akan terpanggil.

3. Jadi apakah yang dimaksud dari virtual method invocation? Mengapa disebut virtual?
   Jawab:
   Disebut virtual karena antara method yang dikenali oleh compiler dan method yang dijalankan oleh JVM berbeda.

PERCOBAAN 3

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:
   Karena pada kode baris ke-8 itu menggunakan konsep heterogen, yang mana didalam array bisa berisi berbagai mecam objek yang berbeda. Dan pada kode baris ke-8 tersebut berisi objek pEmp dan iEmp yang mana merupakan subclass dari class Employee.

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:
   Karena pada kode baris ke-9 itu menggunakan konsep heterogen, yang mana didalam array bisa berisi berbagai mecam objek yang berbeda. Pada kode baris ke-9 tersebut berisi objek pEmp dan eBill yang mana class dari ke2 objek tersebut adalah referensi dari class interface yaitu Payable dengan objek p.

3. Perhatikan baris ke-10, mengapa terjadi error?
   Jawab:
   Karena class dari objek eBill tidak memiliki relasi inheritance dengan Class Employee, sehingga untuk memperbaiki error kita perlu menambahkan keyword ‘extends Employee’ pada class ElectricityBill.

PERCOBAAN 4

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 PermanentEmploy
   Jawab:
   Karena pada ke2 baris tersebut menggunakan konsep polimorfisme argument sehingga ketika objek ow memanggil method pay yang memiliki parameter Payable dan class Payable ini memiliki subclass yaitu PermanentEmployee dan ElectricityBill, maka method pay tersebut bisa menerima argument berupa objek dari ke2 subclass yang disebutkan tadi.

2. Jadi apakah tujuan membuat argument bertipe Payable pada method pay() yang ada di dalam class Owner?
   Jawab:
   Agar method pay() bisa menerima nilai argument dari berbagai bentuk objek.

3. Coba pada baris terakhir method main() yang ada di dalam class Tester4 ditambahkan perintah ow.pay(iEmp); Mengapa terjadi error?
   Jawab:
   Karena parameternya bukan subclass dari Payable sehingga terjadilah error.

4. Perhatikan class Owner, diperlukan untuk apakah sintaks p instanceof ElectricityBill pada baris ke-6 ?
   Jawab:
   Untuk mengecek tipe objek, jika p = ElectricityBill maka argument didalamnya yg akan dijalankan.

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:
   Syntax tersebut untuk mengubah tipe data dari objek.

OUTPUT TUGAS