

วัตถุประสงค์

Lab4 รหัส - ชื่อ นรากรณ์ ดีเย็น 65050446

A. เข้าใจโครงสร้าง หลักการ ของ abstract class

B. สามารถบอกความแตกต่างของผลการสืบทอดจาก class กับ จาก abstract class

C. เข้าใจโครงสร้าง หลักการ ของ interface

กิจกรรมที่ 1

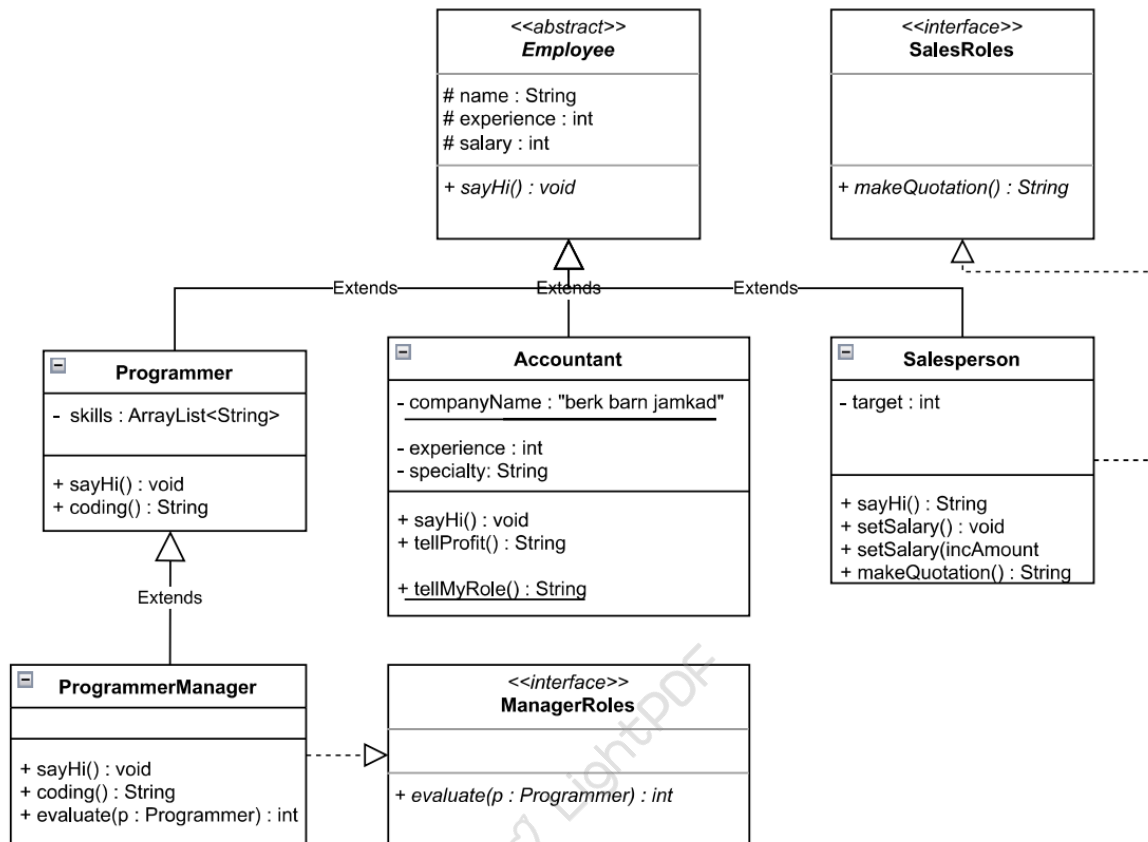
1.1 กำหนด interface CanSwimIntf และผลการเรียกใช้

Lec4Fish และ Lec4Duck ดังแสดง

1.2 เขียน Lec4Fish.java และ Lec4Duck.java

```
1 package packA;  
2 public interface CanSwimIntf {  
3     public void swim();  
4 }
```

```
11 static void q1() {  
12     Lec4Fish fish = new Lec4Fish();  
13     fish.swim();  
14     fish.fishMethod();  
15     // flexing my tail back and forth  
16     // do i?  
17     Lec4Duck duck = new Lec4Duck();  
18     duck.swim();  
19     duck.duckMethod();  
20     // waddling  
21     // quack  
22 }
```



2.1

ลบ EmpTmp.class และ rename EmpTmp.java เป็น Lab3EmpTmp.java

(เพื่อป้องกันการฟลัด รัน งานนี้ผ่าน)

2.2 สร้าง Employee.java ให้เป็น abstract class ให้ sayHi() เป็น abstract

2.3 แปลง Programmer, Salesperson, Accountant ให้สืบทอดจาก Employee นี้แทน (ดังนั้น ทั้ง 3 จะ implement sayHi()) และต่างมี method เฉพาะคลาสตนเอง

2.4 เราสามารถสร้าง interface เช่น SalesRoles เพื่อกำหนด methods สำหรับ class ที่เป็นตำแหน่งที่เกี่ยวข้องกับการขาย ต้องมี ...แต่ในที่นี้บังเอิญซ้ำกับที่เราเคยมีใน Salesperson ไปแล้ว ดังนั้นการระบุ implements SalesRoles ก็ไม่ต้องเปลี่ยนแปลงอะไรกับ Salesperson.java

2.5 สร้าง interface ManagerRoles.java แล้วสร้าง class ProgrammerManager ตาม diagram

2.6 public int evaluate(Programmer p) ให้เพิ่ม salary ของ p 15% (int) และแสดง new salary นั้น

```

5     public static void main(String[] args) {
6         q1();
7         System.out.println(x: "-----");
8         q2();
9     }
10
11    static void q1() {
12        ProgrammerManager pmanager = new ProgrammerManager(n: "CodeReviewer", exp: 9, sal: 550);
13        System.out.println(pmanager);
14        Employee e = pmanager;
15        e.sayHi();
16    }
17
18    static void q2() {
19        ProgrammerManager pmanager = new ProgrammerManager(n: "CodeReviewer", exp: 9, sal: 550);
20        ArrayList<Programmer> aList = new ArrayList<>();
21        aList.add(new Programmer(n: "Keng", exp: 2, sal: 300));
22        aList.add(new Programmer(n: "Somsri", exp: 3, sal: 400));
23        aList.add(new Programmer(n: "haha", exp: 4, sal: 600));
24        for (Programmer p : aList) {
25            int newSalary = pmanager.evaluate(p);
26            System.out.println(newSalary);
27        }
28    }

```

```

ManagerProgrammer [name=CodeReviewer, experience=9salary=550]
Coding in [solidity, typescript]
-----
345
460
690

```

สรุปหลักการ interface พอสังเขป

inter น่าจะมาจาก intersection ซึ่งในทางคณิตศาสตร์หมายถึงสิ่งที่มีร่วมกัน ดังนั้น interface จึงหมายถึง สิ่งที่มีร่วมกันของ class นั้นๆ ของ method ซึ่งในที่นี้คือ ชื่อ method ที่มีเหมือนกันแต่การทำงานอาจต่างกันตาม class นั้นๆ โดยการสร้าง class interface และสร้าง method ที่ต้องการโดยไม่มี body เมื่อต้องการเรียกใช้ให้ใช้ implements แล้วระบบจะบังคับให้ เราต้องเขียน method ตามที่ interface ได้กำหนดไว้ โดยใช้ interface ถือเป็น override แบบหนึ่ง

กำหนดส่ง TBA