ชื่อ-นามสกุล นาย ธนัชชานน พรหมรักษ์ รหัสประจำตัวนักศึกษา 6604062636151 ปีการศึกษา 2567

แบบฝึกหัดปฏิบัติการคาบที่ 7: Polymorphism

คำสั่ง

1. ให้ศึกษาหลักการ Polymorphism ต่อไปนี้
    1. ความสัมพันธ์แบบการสืบทอดคุณสมบัติ อนุญาตให้คลาสสามารถสืบทอดคุณสมบัติจากคลาสแม่(Superclass) ได้
    2. คลาสลูก(Subclass) สามารถเพิ่มคุณสมบัติใหม่ ๆ ได้ โดย Subclass จะเป็นคลาสที่เจาะจงลงไปจากคลาสแม่
    3. ทุก ๆ วัตถุที่สร้างจาก Subclass จะถือว่าเป็น instance ของ superclass ด้วย เช่น
        a. ทุก ๆ วัตถุที่สร้างจากคลาส circle ถือเป็นวัตถุของคลาส Object
        b. แต่ทุก ๆ วัตถุของคลาส Object ไม่ได้เป็นวัตถุชนิด circle
    4. กรณีที่ในเมธอดมีการประกาศรับพารามิเตอร์เป็นแบบ instance ของ superclass เราสามารถส่ง instance ของ subclass ไปเป็นพารามิเตอร์ของ superclass ได้
    5. ตัวแปรอ้างอิงของซูเปอร์คลาสสามารถชี้อินสแตนท์ของซับคลาส
    6. แต่ตัวแปรอ้างอิงของซับคลาสไม่สามารถชี้อินสแตนท์ของซูเปอร์คลาสหรืออินสแตนท์ของคลาสที่มีซูเปอร์คลาสเดียวกันได้ ถ้าจะลองนำไปชี้คลาสอื่นๆที่ไม่มีความเกี่ยวข้องกันเลยยิ่งไม่ได้
    7. การที่ตัวแปรของซูเปอร์คลาสสามารถชี้อินสแตนท์ของซับคลาสได้ เป็นการเพิ่มความยืดหยุ่นในการเขียนโปรแกรม ลักษณะนี้เรียกว่า โพลิมอร์ฟิซึม

Dynamic Binding

- กำหนดให้วัตถุ o เป็นอินสแตนซ์ของคลาส C_1 C_2 ... C_{n-1} C_n เมื่อ C_1 เป็นคลาสลูกของ C_2 และ C_2 เป็นคลาสลูกของ C3 ... และ C_{n-1} เป็นคลาสลูกของ C_n ดังปรากฏในรูปกล่าวคือ
    - C_n จะเป็นคลาสที่มีความเป็นทั่วไปมากที่สุด(most general class)
    - C_1 เป็นคลาสที่มีความเจาะจงมากที่สุด(most specific class) โดยคลาส C_n ในภาษาจาวาคือคลาส Object
- ถ้า o เป็นอินสแตนซ์ของคลาส C_1 โดยที่ o เรียกเมธอด p แล้ว JVM จะทำการค้นหาเมธอดที่ตรงกับที่ต้องการมากที่สุดโดยจะค้นหาเมธอด p ในคลาส C_1 C_2 ... C_{n-1} C_n ตามลำดับจนกว่าจะพบ

Static Binding

คือการผูกตัวแปรอ้างอิงกับชนิดข้อมูลช่วง Compile time

ชื่อ-นามสกุล นาย ธนัชชานนท์ พรหมธีรวิบ รหัสประจำตัวนักศึกษา 6604062636151 *ปีการศึกษา 2567*

1. ให้ศึกษาและทดลองพิมพ์ Code ตัวอย่างการสืบทอดคลาสจากตัวอย่างต่อไปนี้

```
class A{
   void f(){
      System.out.println("A");
   }
}
class B extends A{
   void f(){
      System.out.println("B");
   }
}
class Test38{
   static void t(A a){
      a.f();
   }
   public static void main(String[] args){
      t(new A());
      t(new B());
   }
}
```

ให้อธิบายผลลัพธ์ของโปรแกรมพร้อมอธิบายเหตุใดจึงได้คำตอบดังกล่าว

Output : A
B

function t เป็น static Binding ไม่ต้อง New class มีการรับ parameter เป็น Type A และมีการส่ง new class ที่เป็น A ไป สามารถเรียกใช้ได้ และ new class B ก็สามารถรับได้ เพราะเป็นแบบ polymorphism

```
class A{
   int x=1;
   void inc(){x++;}
   int get(){return x;}
}
class B extends A{
   int x=2;
   void inc(){x++;}
   int get(){return x;}
}
class Test{
  public static void main(String[] args){
      B b =new B();
      A a= b;
      a.inc();
      b.inc(); ①
      System.out.println(a.x+","+b.x); ②
      System.out.println(a.get()+","+b.get());
   }
}
```

ให้อธิบายถึงส่วนที่มีการทำงานเป็น Static binding และ Dynamic binding

① มีการทำ Dynamic binding
จะทำการดูตัวที่สร้าง new class มาว่าสร้างจาก class ใด และเรียกใช้ function class นั้น

② มีการทำ Static binding ดูจาก Type ของมัน

```
public class Test{
   public static void main(String[] args){
      Object fruit = new Fruit();
      Object apple= new (Apple)fruit;
   }        ①
}
class Apple extends Fruit{ }
class Fruit{}
```

จงอธิบายส่วนที่ผิดของโปรแกรม

① ผิด Casting เพราะ Type ต้องเป็น Type ก่อน casting

2. จงอธิบายผลลัพธ์ของ Output ต่อไปนี้

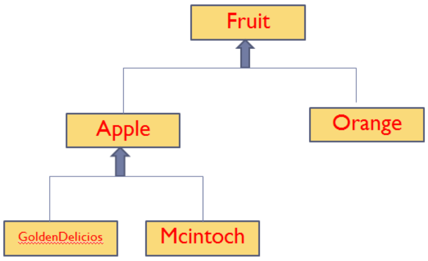
public class Test {
  public static void main(String[] args) {
  Object a1 = new A();
  Object a2 = new Object();
  System.out.println(a1);
  System.out.println(a2);
  }
}
class A {
  int x;
  public String toString() {
  return "A's x is " + x;
  }
}
	A's x is 0
	Object@1a2s

public class Test {
  public static void main(String[] args) {
  Object a1 = new A();
  Object a2 = new A();
  System.out.println(a1.equals(a2));
  }
}
class A {
  int x;
  public boolean equals(A a) {
  return this.x == a.x;
  }
}
	false

public class Test {
  public static void main(String[] args) {
  A a1 = new A();
  A a2 = new A();
  System.out.println(a1.equals(a2));
  }
}
class A {
  int x;
  public boolean equals(A a) {
  return this.x == a.x;
  }
}
	true

ชื่อ-นามสกุล นาย [illegible] รหัสประจำตัวนักศึกษา [illegible] *ปีการศึกษา 2567*

3. กำหนดให้ Fruit, Apple, Orange, Golden Delicious Apple, และ Macintosh Apple มีโครงสร้างการสืบทอดและการประกาศตัวแปรดังนี้ ให้เขียนโปรแกรมเพื่อจำลองการทำงานของโครงสร้างของคลาสดังกล่าวพร้อมตอบคำถามต่อไปนี้



กำหนดการทำงานภายในmain() มีการสร้าง Object ดังนี้

```
Fruit fruit = new GoldenDelicious();
Orange orange = new Orange() ;
```

1. Is fruit instanceof Orange? false
2. Is fruit instanceof Apple? true
3. Is fruit instanceof GoldenDelicious? true
4. Is fruit instanceof Macintosh? false
5. Is orange instanceof Orange? true
6. Is orange instanceof Fruit? true
7. Is orange instanceof Apple? false
8. Suppose the method makeApple() is defined in the Apple class. Can fruit invoke this method? false
   Can orange invoke this method? false
9. Suppose the method makeOrangeJuice() is defined in the Orange class. Can orange invoke this method? true Can fruit invoke this method? false

4. พิจารณาส่วนของโปรแกรมต่อไปนี้

```
public class Test{
   public static void main(String[] args){
     Animal x= new Tiger();
     System.out.println("1. x.news is "+ x.news);
     System.out.println("2. ((Tiger)x).news is "+ ((Tiger)x).news);
     System.out.println("3. x.smile() is "+ x.smile());
     System.out.println("4. ((Tiger)x).smile() is "+ ((Tiger)x).smile());
     System.out.println("5. x.getNews() is "+ x.getNews());
     System.out.println("6. x.getMessage() is "+ x.getMessage() );

   }

}
class Animal{
   public String news= "Animal's news";
   public String message="Animal's message";
   public static String smile(){
     return "smile from Animal";
   }
   public String getNews(){
     return news;
   }
   public String getMessage(){
     return message;
   }
```

```
26 }
27 class Tiger extends Animal{
28    public String news= "Tiger's news";
29    public String message="Tiger's message";
30    public static String smile(){
31      return "smile from Tiger";
32    }
33    //@overide
34    public String getNews(){
35      return news;
36    }
37 }
```

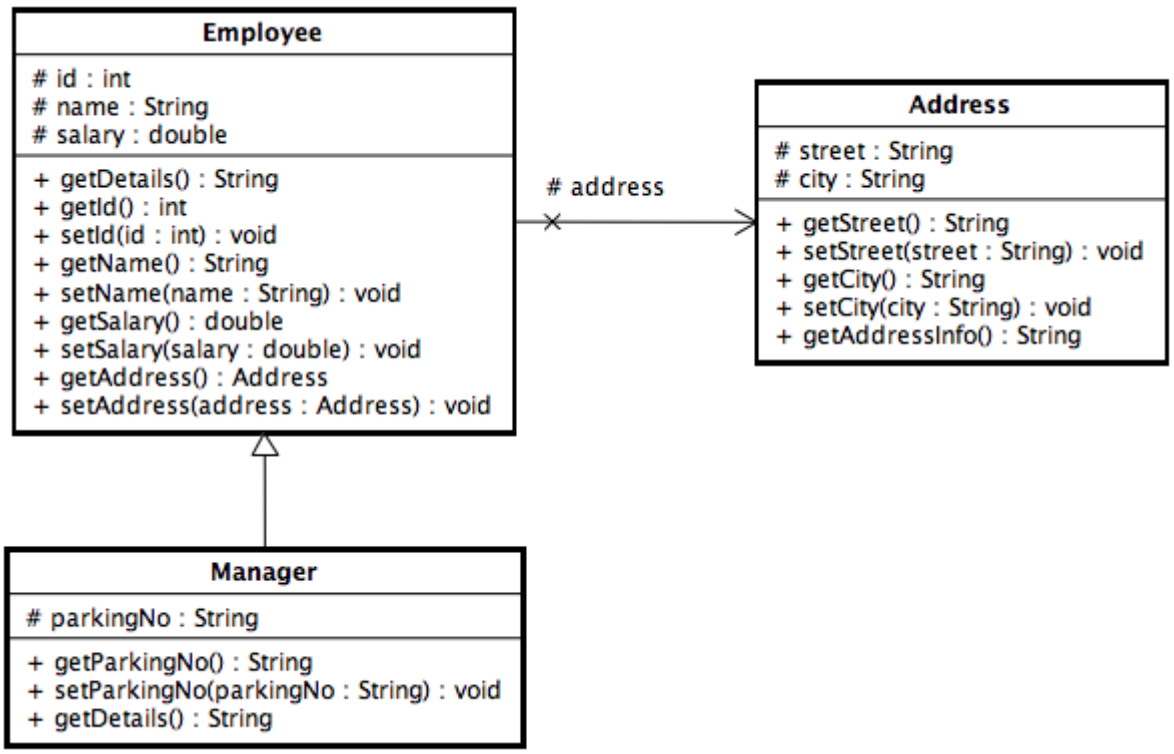
ผลลัพธ์ที่จะเกิดขึ้นจากการใช้โปรแกรมต่อไปนี้คืออะไร

1. x.news is Animal's news.
2. ((Tiger)x).news is Tiger's news
3. x.smile() is smile from Animal
4. ((Tiger)x).smile() is smile from Tiger
5. x.getNews() is Tiger's news
6. x.setMessage() is Animal's message

อธิบายเหตุผลประกอบคำตอบ

1. ใช้ news จาก class Animal
2. ((Tiger)x).news จาก class Tiger เพราะมีการ casting
3. ใช้ function smile จาก class Animal เพราะเป็น static
4. ใช้ function smile จาก class Tiger เพราะมีการ casting
5. ใช้ function setNews จาก class Tiger เพราะมี dynamic binding
6. ใช้ function จาก class Animal เพราะ Tiger ไม่มี function นี้

5.กำหนดความสัมพันธ์ของคลาส Employee และ Manager ในรูปของไดอะแกรม ดังนี้



ให้เขียนโปรแกรมเพื่อจำลองการทำงานของโครงสร้างของคลาสดังกล่าว และประยุกต์ใช้ Polymorphism สำหรับการสร้างวัตถุชนิด `Manager` โดยกำหนดให้สร้างวัตถุ ในคลาส Main ด้วยคำสั่ง `Employee emp = new Manager();` โดยให้กำหนดที่อยู่ เงินเดือนและรายละเอียดของพนักงานได้พร้อมทั้งสามารถพิมพ์รายละเอียดของวัตถุดังกล่าวได้

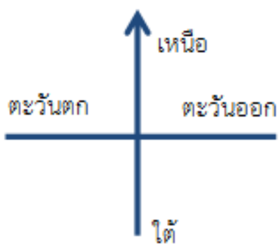
6. [Algorithms] **ฝ่าเขาวงกต (maze) (แบบฝึกหัดเพิ่มเติม ไม่ต้องใช้ concept polymorphism)**
นักล่าขุมทรัพย์นามว่า “อินเดียนา เจ” พลาดพลั้งตกลงไปในหลุมพรางที่ส่งเขาไปอยู่ในเขาวงกตซึ่งมีทางออกอยู่เพียงตำแหน่งเดียวเท่านั้น เคราะห์ดีที่นายอินเดียนามีแผนที่เขาวงกตติดตัวมาด้วย ทำให้เขาทราบตำแหน่งปัจจุบันของเขาและตำแหน่งของทางออก จากแผนที่ อินเดียนาพบว่าพื้นที่เขาวงกตถูกแบ่งออกเป็นช่องจำนวน M แถว N หลัก โดยแต่ละช่องในแผนที่จะมีเลขหนึ่งหรือเลขศูนย์อย่างใดอย่างหนึ่ง ซึ่งเลขศูนย์แทนกำแพงและเลขหนึ่งแทนทางเดิน นอกจากนี้เขาวงกตยังวางตัวในทิศเหนือ-ใต้ ตะวันออก-ตะวันตกพอดี
ดังแสดงในภาพตัวอย่างที่อยู่หน้าถัดไป

อย่าง ไรก็ตามปัญหาหนักใจมีอยู่ว่า บริเวณที่อินเดียนาตกลงมาไม่ได้เชื่อมต่อกับทางออก อินเดียนาจึงจำเป็นที่จะต้องระเบิดกำแพงเขาวงกตด้วยระเบิดที่มีติดตัวอยู่ เพียงลูกเดียวเท่านั้น นอกจากนี้อินเดียนาทราบว่าระเบิดนี้มีพลังทำลายกำแพงเขาวงกตได้เพียงหนึ่ง ช่องเท่านั้น

อินเดียนา จึงจำเป็นที่จะต้องวางแผนว่าเขาจะต้องเดินในเขาวงกตอย่างไร และใช้ระเบิดทำลายกำแพงตรงพื้นที่ช่องใด จึงจะสามารถเดินไปถึงทางออกได้ อินเดียนาทราบ ตำแหน่งเริ่มต้นของเขาและตำแหน่งทางออกเท่านั้น และเพื่อให้การวางแผนและประมาณระยะทางเดินเป็นไปโดยง่าย อินเดียนาจะเดินในทิศเหนือ ใต้ ตะวันออก หรือ ตะวันตก เท่านั้น อินเดียนาจะไม่เดินในทิศเฉียงเป็นอันขาด (เช่น ไม่เดินในทิศตะวันออกเฉียงเหนือ เป็นต้น)

ยกตัวอย่างจากแผนที่ในหน้าถัดไป เขาวงกตนี้ประกอบด้วยช่องจำนวนทั้งหมด 5 แถวและ 8 หลัก กำหนดให้อินเดียนาเริ่มต้นในช่องที่ถูกเน้นด้วยวงรี และทางออกอยู่ ณ ตำแหน่งที่เน้นด้วยสามเหลี่ยม หากอินเดียนาระเบิดกำแพงที่ช่องใดช่องหนึ่งที่ถูกเน้นด้วยลูกศรก็จะสามารถ เดินไปถึงทางออกได้ การระเบิดกำแพงที่ช่องอื่น ๆ นอกจากหนึ่งในสี่ช่องนี้จะไม่ทำให้อินเดียนาไปถึงทางออกได้

ยิ่งไปกว่านั้น อินเดียนายังสนใจด้วยว่าทางเดินจากจุดเริ่มต้นไปถึงทางออกที่ใกล้ที่สุดมี ระยะทางเท่าใด (ระยะทางนับจากจำนวนช่องที่เดินผ่าน) จากตัวอย่างเดิม ถ้าอินเดียนาระเบิดกำแพงที่ช่อง ณ ตำแหน่งแถวที่สอง หลักที่ห้า หรือตำแหน่งแถวที่สาม หลักที่หก จะทำให้ได้ทางเดินที่ใกล้ที่สุดด้วย คือได้ทางเดินที่ผ่านจำนวนช่องทั้งหมด 6 ช่อง (นับช่องที่จุดเริ่มต้นและสิ้นสุดและช่องที่เป็นกำแพงที่ถูกระเบิดด้วย)



0	0	1	1	0	0	0	0
1	0	1	1	0 ⬅	1	1	1 △
1	0	1	1	1	0 ⬅	0	1
1	1	0	0	(1)	0 ⬅	0	1
0	0	1	1	0 ⬅	1	1	1

จง เขียนโปรแกรมที่มีประสิทธิภาพในการหาจำนวนช่องของกำแพงที่อินเดียนาสามารถทำ การระเบิดเพื่อนำอินเดียนาไปสู่ทางออกได้ รวมทั้งหาระยะทางเดินที่สั้นที่สุดจากจุดเริ่มต้นไปจนถึงทางออก

ข้อมูลนำเข้า

1. บรรทัดแรกระบุค่า M และ N ซึ่งแทนจำนวนแถวและจำนวนหลักของเขาวงกตตามลำดับ โดยที่ $1 \leq M,N < 150$ โดย M และ N ถูกคั่นด้วยช่องว่าง

2. บรรทัดที่สองระบุแถว r_s และหลัก c_s ของช่องที่อินเดียนาเริ่มต้น โดยที่ $1 \leq r_s \leq M$ และ $1 \leq c_s \leq N$ โดย r_s และ c_s ถูกคั่นด้วยช่องว่าง

3. บรรทัดที่สามระบุแถว r_e และหลัก c_e ของช่องที่อินเดียนาเริ่มต้น โดยที่ $1 \leq r_e \leq M$ และ $1 \leq c_e \leq N$ โดย r_e และ c_e ถูกคั่นด้วยช่องว่าง

4. อีก M บรรทัดถัดมา ในแต่ละบรรทัดจะประกอบไปด้วยเลขจำนวน N ตัว แต่ละตัวคั่นด้วยช่องว่าง โดยเลขศูนย์แทนกำแพง และเลขหนึ่งแทนทางเดิน บรรทัดแรกใน M บรรทัด นี้บอกลักษณะช่องของแถวแรกในเขาวงกต (แถวแรกคือแถวที่อยู่ทางเหนือสุด) เรียงจากหลักทางทิศตะวันตกไปตะวันออก (หลักแรกคือหลักทางทิศตะวันตก) บรรทัดถัดมาบอกลักษณะของแถวที่สอง และเป็นเช่นนี้ไปเรื่อย ๆ จนครบ M บรรทัด

ข้อมูลส่งออก

1. บรรทัดแรกระบุจำนวนช่องกำแพงที่อินเดียนาสามารถวางระเบิดและพาอินเดียนาไปถึงทางออกได้

2. บรรทัด ที่สองระบุระยะทางที่น้อยที่สุดที่อินเดียนาสามารถเดินเพื่อไปถึงทางออก โดยระยะทางคือจำนวนช่องที่อินเดียนาเดินผ่านทั้งหมด ซึ่งนับรวมช่องที่เป็นจุดเริ่มต้นและจุดสิ้นสุด พร้อมทั้งนับรวมช่องกำแพงที่อินเดียนาระเบิดด้วย

ตัวอย่างข้อมูลนำเข้า	ตัวอย่างข้อมูลส่งออก
5 8 4 5 2 8 0 0 1 1 0 0 0 0 1 0 1 1 0 1 1 1 1 0 1 1 1 0 0 1 1 1 0 0 1 0 0 1 0 0 1 1 0 1 1 1	4 6
6 8 1 4 2 7 0 0 1 1 0 0 0 0 1 0 1 1 0 0 1 1 1 0 1 1 1 0 0 1 1 1 0 0 1 0 0 1 0 0 1 1 0 1 1 1 0 1 0 1 1 1 1 1	4 13