ใบงานการทดลองที่ 11

เรื่อง การใช้งาน Abstract และ Interface

1. จุดประสงค์ทั่วไป

1.1. รู้และเข้าใจการกำหนดวัตถุ การใช้วัตถุ การซ่อนวัตถุ และการสืบทอดประเภทของวัตถุ

1.2. รู้และเข้าใจโครงสร้างของโปรแกรมเชิงวัตถุ

2. เครื่องมือและอุปกรณ์

เครื่องคอมพิวเตอร์1 เครื่อง ที่ติดตั้งโปรแกรม Eclipse

3. ทฤษฎีการทดลอง

3.1. Abstract Class คืออะไร? มีลักษณะการทำงานอย่างไร? อธิบายพร้อมยกตัวอย่างประกอบ

- คือ Class ชนิดหนึ่งที่สามารถกำหนด Method ทั้งในรูปแบบที่ระบุขั้นตอนการทำงาน (Method Description Process) และแบบระบุเพียงแค่ชื่อ Method และให้ Class อื่น ๆ ที่เรียกใช้งานนำไปเขียนขั้นตอนการดำเนินงานเองเหมือนกับ Interface Class

3.2. Interfaces คืออะไร? มีลักษณะการทำงานอย่างไร? อธิบายพร้อมยกตัวอย่างประกอบ

- คือ Class ชนิดหนึ่งที่คุณลักษณะพิเศษคือ เราสามารถที่จะกำหนดหน้าที่การทำงาน (Method) ได้โดยไม่จำเป็นต้องระบุรูปแบบการทำงาน หรือขั้นตอนการทำงานภายใน Method (Method Description Process) เพื่อให้ Class ที่นำ Interface Class นี้ไปใช้งาน ไประบุรูปแบบการทำงานเองตามที่ต้องการ

3.3. คำสั่ง extends และ implements มีการใช้งานที่แตกต่างกันอย่างไร?

- extends จะ extends ได้แค่ class เดียว ส่วน implements จะกี่ class ก็ได้ แต่เมื่อ implements แล้วเราจะต้อง Overriding ทุก method ของ class ที่เรา implements มา

3.4. ภายใน Abstract Class มี Constructor หรือไม่? เพราะเหตุใด?

- การเรียกใช้ Abstract Class จะใช้คำสั่ง extends [ชื่อ Abstract Class]
Method ที่ต้องการระบุเ?พียงแค่ชื่อ Method เท่านั้นจะต้องขึ้นต้นด้วย Keyword abstract

3.5. ภายใน Interface มี Constructor หรือไม่? เพราะเหตุใด?

- ในการประกาศ interface นั้นจะใช้คำสั่ง interface ตามด้วยชื่อของมัน InterfaceName และภายในบล็อคคำสั่งของ interface จะประกอบไปด้วยค่าคงที่ และส่วนหัวของเมธอดเท่านั้น

4. ลำดับขั้นการปฏิบัติการ

4.1. ให้ผู้เรียนสร้าง Abstract Class ของรถถัง(ClassicTank) โดยจะต้องมีรายละเอียดดังต่อไปนี้

4.1.1. Properties : HP เพื่อกำหนดค่าพลังให้กับรถถัง

4.1.2. Properties : Str เพื่อกำหนดค่าความแรงในการยิงของรถถัง

4.1.3. Properties : Vit เพื่อกำหนดค่าพลังป้องกันของรถถัง

4.1.4. Properties : BaseDamage เพื่อการกำหนดค่าพลังการโจมตีพื้นฐาน

4.1.5. Method : SetHP( ) ; เพื่อทำการกำหนดค่าพลังเริ่มต้น

4.1.6. Method : GetHP( ) ; เพื่อตรวจสอบค่าพลัง ณ เวลาปัจจุบัน

4.1.7. Method : Attack(Tank Enemy) ; เพื่อทำการยิงปืนใหญ่โจมตศัตรู โดยการโจมตีจะเป็นการลดค่าพลังของรถถังฝั่ง ตรงกันข้าม (Enemy คือรถถังของศัตรู, Points คือค่าพลังโจมตีของเรา)

4.2. ให้ผู้เรียนสร้างคลาส NormalTank เพื่อสืบทอด ClassicTank เพื่อเขียนรายละเอียดของ Method ทั้งหมด อันได้แก่ SetHP() , GetHP() , Attack(Tank Enemy)

4.3. ในคลาสหลัก ให้สร้าง Instance จาก NormalTank อยู่จำนวน 2 คัน เพื่อทำการต่อสู้กัน โดยควรต้องมีบทบาทดังนี้ 4.3.1. สร้างรถถัง A และ B ให้มีค่าพลังเบื้องต้นดังต่อไปนี้

ค่าสถานะ	รถถัง A	รถถัง B
HP	200	250
Str	12	8
Vit	9	10
BaseDamage	11	10

4.3.2. รถถังทั้ง A และ B ผลัดกันโจมตีซึ่งกันและกัน เพื่อมุ่งหวังให้ค่าพลังของฝั่งตรงกันข้ามลดลงจนค่า HP = 0

4.3.3. รายละเอียดของพลังการโจมตีสามารถคำนวณได้ตามสมการดังต่อไปนี้ DamagePoint = MyTank_BaseDamage * Floor(MyTank_Str / Enermy_Vit) * Random(0.7, 0.9)

4.3.4. แสดงผลการทำงานผ่าน Console เพื่อให้เห็นรายละเอียดค่าพลังปัจจุบันของรถถังแต่ละคัน พลังการโจมต่อ ณ ขณะนั้น จนกว่าจะมีรถถังคันใดคันหนึ่งมีค่า HP = 0

โค้ดโปรแกรมภายใน Abstract Class

```
abstract class ClassicTank{
    int  Str, Vit, BaseDamage;
    double HP , point_A , point_B  ;
    public abstract void setHP();
    public abstract void getHP();
    public abstract void attank();
}
```

โค้ดโปรแกรมภายใน NormalTank

```
   public static void main(String[] atgs) {
       class normalTankA extends ClassicTank{
           public void setHP() {
               HP = 200;
               Str = 12;
               Vit = 9;
               BaseDamage = 11;
           }
           public void getHP() {
               System.out.println("|-- Tank A --|");
               System.out.println(" HP : " + HP);
               System.out.println(" Str : " + Str);
               System.out.println(" Vit : " + Vit);
               System.out.println(" BaseDamage : "+ BaseDamage);
           }
           public void attank() {
               double min = 0.7 ;
               double max = 0.9 ;
               double number = (double)(Math.random()*(max-min+1)+min);
               double DamagePoint = BaseDamage *( 1.3 ) * number;
               System.out.println(" DamagePoint_tankA : "+DamagePoint);
               point_A = DamagePoint;

           }
       }
class normalTankB extends ClassicTank{
    public void setHP() {
        HP = 250;
        Str = 8;
        Vit = 10;
        BaseDamage = 10;
    }
    public void getHP() {
        System.out.println("|-- Tank B --|");
        System.out.println(" HP : "+HP);
        System.out.println(" Str : "+Str);
        System.out.println(" Vit : "+Vit);
        System.out.println(" BaseDamage : "+ BaseDamage);
    }
    public void attank() {
        float min = (float) 0.7 ;
        float max = (float) 0.9 ;
        float number = (float)(Math.random()*(max-min+0.1)+min);
        double DamagePoint = BaseDamage *( 0.8 ) * number;
        System.out.println(" DamagePoint_tankB : "+ DamagePoint);
        point_B = DamagePoint;

    }
}
```

โค้ดโปรแกรมภายในฟังก์ชันการทำงานหลัก

```
public class Lab11 {
    public static void main(String[] atgs) {
        class normalTankA extends ClassicTank{
            public void setHP() {
                HP = 200;
                Str = 12;
                Vit = 9;
                BaseDamage = 11;
            }
```

ผลลัพธ์การทำงานของโปรแกรม

```
|-- Tank A --|
 HP : 200.0
 Str : 12
 Vit : 9
 BaseDamage : 11
------------------------------------------------------------
|-- Tank B --|
 HP : 250.0
 Str : 8
 Vit : 10
 BaseDamage : 10
-------------- ROUND 1 ---------------
 DamagePoint_tankA : 14.1944188318329
 DamagePoint_tankB : 6.475021839141846
|-- Tank A --|
 HP : 200.0
 Str : 12
 Vit : 9
 BaseDamage : 11
|-- Tank B --|
 HP : 250.0
 Str : 8
 Vit : 10
 BaseDamage : 10
-------------- ROUND 2 ---------------
 DamagePoint_tankA : 12.12476313647427
 DamagePoint_tankB : 6.919761657714844
|-- Tank A --|
 HP : 200.0
 Str : 12
 Vit : 9
 BaseDamage : 11
|-- Tank B --|
 HP : 250.0
 Str : 8
 Vit : 10
 BaseDamage : 10
```

4.4. เปลี่ยน Abstract Class ให้กลายเป็น Interfaces และเปรียบเทียบผลลัพธ์การทำงานของโปรแกรม

หลังจากเปลี่ยน Abstract Class เป็น Interface แลว้ เกิดอะไรขึ้น อย่ ง? อธิบายพร้อมยกตัวอย่างประกอบให้ชัดเจน

5. สรุปผลการปฏิบัติการ

- จากการทดลองพบว่าเมื่อเราเปลี่ยนมาเป็นแบบใช้ interface จะยุ่งยากและสบสัน

6. คำถามท้ายการทดลอง

6.1. เมื่อใดจึงควรเลือกใช้งาน Abstract Class

- เมื่อเวลาที่เราต้องการจะใช้ตัวแปรนั้นหลายๆรอบแต่ค่าจะไม่เหมือนกัน

6.2. เมื่อใดจึงควรเลือกใช้งาน Interface

- เมื่อเวลาที่เราต้องการจะใช้ตัวแปรนั้นหลายๆรอบแต่ค่าจะไม่เหมือนกัน และ เราต้องมาก าหนดค่าอีกรอบนึง