

ใบงานการทดลองที่ 11

เรื่อง การใช้งาน Abstract และ Interface

1. จุดประสงค์ทั่วไป

- 1.1. รู้และเข้าใจการกำหนดวัตถุ การใช้วัตถุ การซ่อนวัตถุ และการสืบทอดประเภทของวัตถุ
- 1.2. รู้และเข้าใจโครงสร้างของโปรแกรมเชิงวัตถุ

2. เครื่องมือและอุปกรณ์

เครื่องคอมพิวเตอร์ 1 เครื่อง ที่ติดตั้งโปรแกรม Eclipse

3. ทฤษฎีการทดลอง

- 3.1. Abstract Class คืออะไร? มีลักษณะการทำงานอย่างไร? อธิบายพร้อมยกตัวอย่างประกอบ

คือ Class ที่เอาไว้ซ่อนรายละเอียดจุดเด่น คือจะมี Abstract Method ซึ่ง Abstract Method จะไม่มีรายละเอียดของ Method อยู่ข้างใน ถ้าอยากรู้ใช้งานต้องสืบทอดไปอีกทีหนึ่ง

- 3.2. Interfaces คืออะไร? มีลักษณะการทำงานอย่างไร? อธิบายพร้อมยกตัวอย่างประกอบ

คือ Abstract Class ที่สมบูรณ์แบบ สมบูรณ์แบบในที่นี้คือใน Method ของ Interfaces จะไม่มีรายละเอียดอยู่เลย ๐ ถ้าอยากรู้ใช้งานต้อง implements ไป ถึงจะใช้งานได้ interfaces ไม่สามารถที่จะสร้าง instance ตรงๆได้จะต้องสืบทอดไป แล้วไปสร้าง instance ใน class ลูกอีกทีหนึ่งถึงจะทำได้

- 3.3. คำสั่ง extends และ implements มีการใช้งานที่แตกต่างกันอย่างไร?

Extends ใช้กับ Class และ Abstract Class Implements ใช้กับ Interfaces

- 3.4. ภายใน Abstract Class มี Constructor หรือไม่? เพราะเหตุใด?

มีได้เพราะ มีการประกาศ Properties ใน class

- 3.5. ภายใน Interface มี Constructor หรือไม่? เพราะเหตุใด?

ไม่มีเพราะใน Interface ไม่มี Properties

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 เพื่อกำหนดค่าพลังการโจมตีพื้นฐาน
- 4.1.5. Method : SetHP() ; เพื่อทำการกำหนดค่าพลังเริ่มต้น
- 4.1.6. Method : GetHP() ; เพื่อตรวจสอบค่าพลัง ณ เวลาปัจจุบัน
- 4.1.7. Method : Attack(Tank Enemy) ; เพื่อทำการยิงปืนใหญ่โจมตีศัตรู โดยการโจมตี จะเป็นการลดค่าพลังของรถถังฝั่งตรงกันข้าม (Enemy คือรถถังของศัตรู, Points คือค่าพลังโจมตีของเรา)
- 4.2. ให้ผู้เรียนสร้างคลาส NormalTank เพื่อสืบทอด ClassicTank เพื่อเขียนรายละเอียดของ Method ทั้งหมดอันได้แก่ 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

$$\text{DamagePoint} = \text{MyTank_BaseDamage} * \text{Floor}(\text{MyTank_Str} / \text{Enemy_Vit}) * \text{Random}(0.7, 0.9)$$
- 4.3.4. แสดงผลการทำงานผ่าน Console เพื่อให้เห็นรายละเอียดค่าพลังปัจจุบันของรถถังแต่ละคัน พลังการโจมตี ณ ขณะนั้น จนกว่าจะมีรถถังคันใดคันหนึ่งมีค่า HP = 0

โค้ดโปรแกรมภายใน Abstract Class

```

1 package lab11;
2
3 abstract class ClassicTank {
4
5     public int HP;
6     public int Str;
7     public int Vit;
8     public int BaseDamage;
9
10    public abstract void SetHP(int x);
11    public abstract void GetHP();
12    public abstract void Attack(int x);
13
14 } //end abstract class

```

โค้ดโปรแกรมภายใน NormalTank

```

1 package lab11;
2
3 public class NmTank1 extends ClassicTank{
4
5     @Override
6     public void SetHP(int x) {
7         HP = x;
8     }
9
10    @Override
11    public void GetHP() {
12        System.out.println("Tank 1 Have HP " + HP);
13    }
14
15    @Override
16    public void Attack(int x) {
17        System.out.println("Tank 1 Take DMG " + x);
18        HP = HP - x;
19    }
20
21 } //end class

```

```

1 package lab11;
2
3 public class NmTank2 extends ClassicTank{
4
5     @Override
6     public void SetHP(int x) {
7         HP = x;
8     }
9
10    @Override
11    public void GetHP() {
12        System.out.println("Tank 2 Have HP " + HP);
13    }
14
15    @Override
16    public void Attack(int x) {
17        System.out.println("Tank 2 Take DMG " + x);
18        HP = HP - x;
19    }
20
21 } //end class

```

โค้ดโปรแกรมภายในฟังก์ชันการทำงานหลัก

```

1 package lab11;
2
3 import java.lang.Math;
4
5 public class main {
6     public static float random(double d, double e) {
7         return (float) ( (float)(Math.random() * (e - d)) + d);
8     }
9     public static void main(String[] args) {
10
11         int turn = 1;
12         int DamagePoint = 0;
13
14         NmTank1 NT1 = new NmTank1();
15         NmTank2 NT2 = new NmTank2();
16
17         //Tank A
18         NT1.SetHP(200);
19         NT1.Str = 12;
20         NT1.Vit = 9;
21         NT1.BaseDamage = 11;
22
23         //Tank B
24         NT2.SetHP(250);
25         NT2.Str = 8;
26         NT2.Vit = 10;
27         NT2.BaseDamage = 10;
28
29         System.out.println("////////////////////////////////");
30         NT1.GetHP();
31         NT2.GetHP();
32
33         System.out.println("////////////////////////////////");
34         System.out.println("Start");
35
36         do {
37             System.out.println("////////////////////////////////");
38             System.out.println("Round = " + turn);
39             if(turn%2 == 0) {
40                 //Tank1 ATTACK
41                 DamagePoint = (int) (NT1.BaseDamage * Math.floorDiv( NT1.Str , NT2.Vit ) * random(0.7, 0.9) );
42                 NT2.Attack(DamagePoint);
43                 NT2.GetHP();
44             } else {
45                 //Tank2 ATTACK
46                 //Floor(8/9) = 0 Tank2 DamagePoint == 0 ?
47                 DamagePoint = (int) (NT2.BaseDamage *
48                     Math.floorDiv( NT2.Str , NT1.Vit ) * random(0.7, 0.9) );
49                 NT1.Attack(DamagePoint);
50                 NT1.GetHP();
51             }
52             turn++;
53             if( NT1.HP <= 0 || NT2.HP <= 0 ) {
54                 break;
55             }
56             System.out.println("////////////////////////////////");
57         } while( turn != 0 );
58         System.out.println("////////////////////////////////");
59         if(NT1.HP <= 0) {
60             System.out.println("Tank 2 WIN!!!!!!");
61         } else if(NT2.HP <= 0) {
62             System.out.println("Tank 1 WIN!!!!!!");
63         }
64         System.out.println("////////////////////////////////");
65     }
66 } //end class

```

ผลลัพธ์การทำงานของโปรแกรม

```
Round = 16
Tank 2 Take DMG 7
Tank 2 Have HP 21
////////////////////////////////////
Round = 17
Tank 1 Take DMG 0
Tank 1 Have HP 50
////////////////////////////////////
Round = 18
Tank 2 Take DMG 7
Tank 2 Have HP 14
////////////////////////////////////
Round = 19
Tank 1 Take DMG 0
Tank 1 Have HP 50
////////////////////////////////////
Round = 20
Tank 2 Take DMG 8
Tank 2 Have HP 6
////////////////////////////////////
Round = 21
Tank 1 Take DMG 0
Tank 1 Have HP 50
////////////////////////////////////
Round = 22
Tank 2 Take DMG 7
Tank 2 Have HP -1
Tank 1 WIN!!!!!!
////////////////////////////////////
```

4.4. เปลี่ยน Abstract Class ให้กลายเป็น Interfaces และเปรียบเทียบผลลัพธ์การทำงานของโปรแกรม

หลังจากเปลี่ยน Abstract Class เป็น Interface แล้ว เกิดอะไรขึ้นอย่าง? อธิบายพร้อมยกตัวอย่างประกอบให้ชัดเจน

```
abstract class ClassicTank {

    public int HP;
    public int Str;
    public int Vit;
    public int BaseDamage;

    public abstract void SetHP(int x);
    public abstract void GetHP();
    public abstract void Attack(int x); ;
    public class NmTank1 extends ClassicTank{
```

5. สรุปผลการปฏิบัติการ

การใช้ Abstract Class กับ Interface มีการใช้งานที่คล้ายๆกัน จะมีส่วนที่ต่างกันตรงที่ ็ Properties โดยใน Interface จะไม่สามารถประกาศ Properties ได้แต่ใน Abstract Class ทำได้จากการทดลองที่ให้ทำการสร้างถึง 2 คำนมาสลับกันยังแบบใช้ ็ Abstract กับ Interface

หากแก้ไขตรงตามเงื่อนไขแล้วพบว่า ็ ว่า ผลลัพธ์ของที่

ง 2 แบบเหมือนกัน และผลลัพธ์ของ DamagePoint ที่คำนวณได้จากสูตร

$$Nt2.BaseDamage * Math.floorDiv(Nt2.Str, Nt1.Vit) * random(0.7, 0.9)$$
 พบว่าได้ 0 ั ตลอดเพราะ $Math.floorDiv(Nt2.Str, Nt1.Vit)$

หากแทนค่าจะพบว่า $Math.floorDiv(8, 9)$ จะได้ 0 แล้วคูณในสมการจะได้ ็ 0 ($10 * 0 * random(0.7, 0.9)$)

6. คำถามท้ายการทดลอง

6.1. เมื่อใดจึงควรเลือกใช้งาน Abstract Class

เมื่อต้องเขียนโปรแกรมที่มี Properties ซ้ำกันเยอะๆ หรือมี Properties ที่เหมือนกันเยอะ เช่น HP STRDEF AGI เป็นต้น

6.2. เมื่อใดจึงควรเลือกใช้งาน Interface

เมื่อต้องเขียนโปรแกรมที่มี Properties ไม่ซ้ำกัน หรือ มี Properties เฉพาะเยอะ