

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

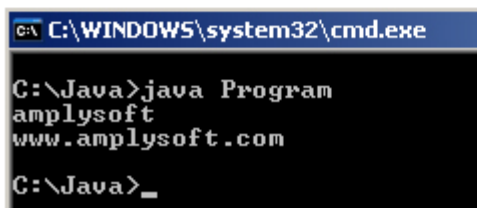
3.1. Abstract Class คืออะไร? มีลักษณะการทำงานอย่างไร? อธิบายพร้อมยกตัวอย่างประกอบ

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

ตัวอย่างโปรแกรม

```
1  abstract class Abstract {
2      public void sayHello(){
3          System.out.println( "amplysoft" );
4      }
5      abstract void sayWebsite();
6  }
7
8  class Sample extends Abstract {
9      public void sayWebsite(){
10         System.out.println("www.amplysoft.com");
11     }
12 }
13
14 class Program {
15     public static void main(String args[]){
16         Sample s = new Sample();
17         s.sayHello();
18         s.sayWebsite();
19     }
20 }
```

ผลลัพธ์



```
C:\WINDOWS\system32\cmd.exe

C:\Java>java Program
amplysoft
www.amplysoft.com
C:\Java>
```

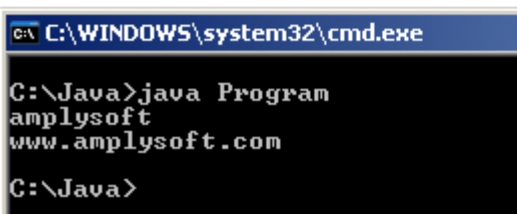
3.2. Interfaces คืออะไร? มีลักษณะการทำงานอย่างไร? อธิบายพร้อมยกตัวอย่างประกอบ

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

ตัวอย่างโปรแกรม

```
1  interface Interface {
2      void sayHello();
3      void sayWebsite();
4  }
5
6  class Sample implements Interface {
7      public void sayHello(){
8          System.out.println("amplysoft");
9      }
10     public void sayWebsite(){
11         System.out.println("www.amplysoft.com");
12     }
13 }
14
15 class Program {
16     public static void main(String args[]){
17         Sample s = new Sample();
18         s.sayHello();
19         s.sayWebsite();
20     }
21 }
```

ผลลัพธ์



```
C:\WINDOWS\system32\cmd.exe
C:\Java>java Program
amplysoft
www.amplysoft.com
C:\Java>
```

3.3. คำสั่ง extends และ implements มีการใช้งานที่ต่างกันอย่างไร?

- 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 ที่ต้องการระบุไว้เพียงแค่ชื่อ Method เท่านั้นจะต้องขึ้นต้นด้วย Keyword abstract

3.5. ภายใน Interface มี Constructor หรือไม่? เพราะเหตุใด?

- ในการประกาศ interface นั้นจะใช้คำสั่ง interface ตามด้วยชื่อของมัน InterfaceName และภายในบล็อกคำสั่งของ 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 เพื่อกำหนดค่าพลังการโจมตีพื้นฐาน

4.1.5. Method : SetHP() ; เพื่อทำการกำหนดค่าพลังเริ่มต้น

4.1.6. Method : GetHP() ; เพื่อตรวจสอบค่าพลัง ณ เวลาปัจจุบัน

4.1.7. Method : Attack(Tank Enemy) ; เพื่อทำการยิงปืนใหญ่โจมตีศัตรู 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 / Enemy_Vit) * Random(0.7, 0.9)

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

โค้ดโปรแกรมภายใน Abstract Class

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

โค้ดโปรแกรมภายใน NormalTank

```
class normalTankB extends ClassicTank{
    public void setHP0{
        HP = 250;
        Str = 8;
        Vit = 10;
        BaseDamage = 10;
    }
    public void getHP0{
        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 attank0{
        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

```
normalTankA tankA =new normalTankA0;
normalTankB tankB =new normalTankB0;

tankA.setHP0;
tankA.getHP0;
System.out.println("-----");
tankB.setHP0;
tankB.getHP0;
System.out.println("----- ROUND 1 -----");
tankA.attank0;
tankB.attank0;
tankA.getHP0;
tankB.getHP0;
System.out.println("----- ROUND 2 -----");
tankA.attank0;
tankB.attank0;
tankA.getHP0;
tankB.getHP0;
```

ผลลัพธ์การทำงานของโปรแกรม

```
eclipse-workspace - OOPProject/src/Project/LAB11.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help

Package Explorer
ENGCE174_OOP-main
  Exam1
  LAB1
  LAB10
  LAB11
    bin
    Project
    src
    Project
    Lab11.pdf
  LAB12
  LAB2
  LAB3
  LAB4
  LAB5
  LAB6
  Lab7
  LAB8
  Lab4
  Lab5
  Lab6
  Lab7
  OOP
  OOPProject
    JRE System Library [JavaSE-17]
    src
      Lab7
      OOPPackage
      Project
        LAB11.java
        module-info.java

LAB11.java
29      double number = (double)(Math.random()*(max-min+1)+min);
30      double DamagePoint = BaseDamage * ( 1.3 ) * number;
31      System.out.println(" DamagePoint_tankA : "+DamagePoint);
32      point_A = DamagePoint;
33
34  }
35
36  class normalTankB extends ClassicTank{
37      public void setHP() {
38          HP = 250;
39          Str = 8;
40      }
41  }

Console
<terminated> LAB11 [Java Application] C:\Users\parad\p2\poo\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.5.v20221102-0933\jre\bin\javaw.exe
|-- 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 : 17.46434416577503
DamagePoint_tankB : 6.258127212524414
|-- 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 : 11.055881866266862
DamagePoint_tankB : 5.769615173339844
|-- Tank A --|
HP : 200.0
Str : 12
Vit : 9
BaseDamage : 11
|-- Tank B --|
HP : 250.0
Str : 8
Vit : 10
BaseDamage : 10

Snip & Sketch
Snip saved to cli
Select here to ma
```

4.4. เปลี่ยน Abstract Class ให้กลายเป็น Interfaces และเปรียบเทียบผลลัพธ์การทำงานของโปรแกรม

หลังจากเปลี่ยน Abstract Class เป็น Interface แล้ว เกิดอะไรขึ้น อยู่ ง? อธิบายพร้อมยกตัวอย่างประกอบให้ชัดเจน

```
Package Lab11;
Interface ClassicTank{
    public abstract void getHP();
    public abstract void attank();
}
public class LAB11{
    public static void main(String[] args){
        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 attank0{
        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;
    }
}

normalTankA tankA=new normalTankA0;
normalTankB tankB=new normalTankB0;

tankA.setHP0;
tankA.getHP0;
System.out.println("-----");
tankB.setHP0;
tankB.getHP0;
System.out.println("----- ROUND 1 -----");
tankA.attank0;
tankB.attank0;
tankA.getHP0;
tankB.getHP0;
System.out.println("----- ROUND 2 -----");
tankA.attank0;
tankB.attank0;
tankA.getHP0;
tankB.getHP0;
}
}

```

5. สรุปผลการปฏิบัติการ

- จากการทดลองพบว่าเมื่อเราเปลี่ยนมาเป็นแบบใช้ interface จะยุ่งยากและสับสน

6. คำถามท้ายการทดลอง

6.1. เมื่อใดจึงควรเลือกใช้งาน Abstract Class

- เมื่อเวลาที่เราต้องการจะใช้ตัวแปรนั้นหลายๆรอบแต่ค่าจะไม่เหมือนกัน

6.2. เมื่อใดจึงควรเลือกใช้งาน Interface

- เมื่อเวลาที่เราต้องการจะใช้ตัวแปรนั้นหลายๆรอบแต่ค่าจะไม่เหมือนกัน และ เราต้องมาก หนดค่าอีก รอบหนึ่ง