Faculty of Information Technology
King Mongkut's Institute of Technology Ladkrabang

ชื่อ นายพีรวิชญ์ นามสกุล พิชญชาดาพงศ์

รหัสนักศึกษา 65070162 กลุ่มการเรียน Sec 2

แบบฝึกปฏิบัติ ครั้งที่ 5

เรื่อง การเขียนโปรแกรมเชิงวัตถุ

วัตถุประสงค์
1. เพื่อฝึกฝนการใช้คุณสมบัติการห่อหุ้ม (Encapsulation)
2. เพื่อฝึกฝนการใช้คุณสมบัติการสืบทอด (Inheritance)

---

1. ให้นักศึกษาสร้างคลาส `Player` ตามคลาสไดอะแกรมต่อไปนี้

Player	
- name	: String
- team	: String
+ setName (String n)	: void
+ getName ()	: String
+ setTeam (String t)	: void
+ getTeam ()	: String
+ isSameTeam(Player p)	: boolean

โดยกำหนดให้แต่ละเมธอดมีรายละเอียดดังต่อไปนี้

- เมธอด `setName()` จะนำค่า `n` ไปกำหนดให้แอททริบิวท์ `name` ขณะที่ เมธอด `setTeam()` จะนำค่า `t` ไปกำหนดให้แอททริบิวท์ `team`
- เมธอด `getName()` จะคืนค่าเป็นแอททริบิวท์ `name` และเมธอด `getTeam()` จะคืนค่าเป็นแอททริบิวท์ `team`
- เมธอด `isSameTeam()` จะคืนค่าเป็น `true` ก็ต่อเมื่อ วัตถุของคลาส `Player` ที่เรียกใช้มีค่าของแอททริบิวท์ `team` เหมือนกับของวัตถุ `p` ถ้าไม่เหมือนกันจะคืนค่าเป็น `false`

กำหนดโค้ดสำหรับทดสอบความถูกต้องของคลาส `Player` ที่นักศึกษาได้พัฒนาขึ้น

กรณีที่ 1

```
public class Main {
  public static void main(String[] args) {
    Player p1 = new Player();
    p1.setName("Bank");
    p1.setTeam("Gate OR");

    Player p2 = new Player();
    p2.setName("Khim");
    p2.setTeam("Gate OR");

    if(p1.isSameTeam(p2))
      System.out.println(p1.getName() +" is a same team with "+p2.getName());
    else
      System.out.println(p1.getName() +" is not a same team with "+p2.getName());
  }
}
```

ตัวอย่างผลลัพธ์

```
Bank is a same team with Khim
```

กรณีที่ 2

```
public class Main {
  public static void main(String[] args) {
    Player p1 = new Player();
    p1.setName("Bank");
    p1.setTeam("Gate OR");

    Player p2 = new Player();
    p2.setName("Khim");
    p2.setTeam("Gate AND");

    if(p1.isSameTeam(p2))
      System.out.println(p1.getName() +" is a same team with "+p2.getName());
    else
      System.out.println(p1.getName() +" is not a same team with "+p2.getName());
  }
}
```

ตัวอย่างผลลัพธ์

```
Bank is not a same team with Khim
```

กรณีที่ 3

```
public class Main {
  public static void main(String[] args) {
    Player p1 = new Player();
    p1.setName("Bank");
    p1.setTeam("Gate OR");
    System.out.println(p1.name);
  }
}
```

ตัวอย่างผลลัพธ์

```
Main.java:6: error: name has private access in Player
```

เนื่องจากสาเหตุใดทำไมจึงเกิดข้อความ Error ดังกล่าว

เพราะว่า attribute name ของ class Player ถูกตั้งค่า access modifier ให้เป็น private ดังนั้น object จากต่างคลาสกันจึงไม่สามารถเรียกใช้ attribute name ได้

2. ให้นักศึกษาสร้างคลาส `FootballPlayer` ซึ่งสืบทอดมาจากคลาส `Player` ดังไดอะแกรมต่อไปนี้

FootballPlayer	
- playerNumber	: int
- position	: String
+ setPlayerNumber (int n)	: void
+ getPlayerNumber ()	: int
+ setPosition (String p)	: void
+ getPosition ()	: String
+ isSamePosition(FootballPlayer p)	: boolean

โดยกำหนดให้แต่ละเมธอดมีรายละเอียดดังต่อไปนี้

- เมธอด `setPlayerNumber()` จะนำค่า `n` ไปกำหนดให้แอททริบิวท์ `playerNumber` ขณะที่ เมธอด `setPosition()` จะนำค่า `p` ไปกำหนดให้แอททริบิวท์ `position`
- เมธอด `getPlayerNumber ()` จะคืนค่าเป็นแอททริบิวท์ `playerNumber`และเมธอด `getPosition()` จะคืนค่าเป็นแอททริบิวท์ `position`
- เมธอด `isSamePosition()` จะคืนค่าเป็น `true` ก็ต่อเมื่อ วัตถุของคลาส `FootballPlayer` ที่เรียกใช้มีค่าของแอททริวบิวท์ `team` และ `position` เหมือนกับของวัตถุ `p` ที่รับเข้ามา ถ้าไม่เหมือนกันจะคืนค่าเป็น `false`

2.1. ให้นักศึกษาลองสร้างเมธอด `isSamePosition()` โดยอาศัยโค้ดดังต่อไปนี้

```
public boolean isSamePosition (FootballPlayer  p) {
      if((p.getPosition().equals(this.getPosition()))&
      (p.getTeam().equals(this.getTeam()))){
            return true;
      }else{
            return false;
      }
}
```

โปรแกรมสามารถประมวลผลได้ตามปกติหรือไม่ ถ้าไม่เพราะอะไร

ประมวลผลได้ตามปกติ

2.2. ให้นักศึกษาลองสร้างเมธอด `isSamePosition()` โดยอาศัยโค้ดดังต่อไปนี้

```
public boolean isSamePosition (FootballPlayer  p) {
      if((p.getPosition().equals(this.position))&
      (p.getTeam().equals(this.team))){
            return true;
      }else{
            return false;
      }
}
```

โปรแกรมสามารถประมวลผลได้ตามปกติหรือไม่ ถ้าไม่เพราะอะไร

ประมวลผลไม่ได้ เพราะแม้ class FootballPlayer จะสืบทอด attribute มาจาก class Player จึงทำให้มี attribute team เหมือนกันแต่ว่า ไม่สามารถเรียกใช้งานได้เพราะไม่ได้ set access modifier ให้ team เป็น protected

2.3. กำหนดโค้ดสำหรับทดสอบความถูกต้องของคลาส `FootballPlayer` ที่นักศึกษาได้พัฒนาขึ้น

กรณีที่ 1

```
public class Main {
  public static void main(String[] args) {
    FootballPlayer p1 = new FootballPlayer();
    p1.setName("Harry");
    p1.setTeam("Gryffindor");
    p1.setPlayerNumber(1);
    p1.setPosition("keeper");
    FootballPlayer p2 = new FootballPlayer();
    p2.setName("Jame");
    p2.setTeam("Gryffindor");
    p2.setPlayerNumber(1);
    p2.setPosition("keeper");

    System.out.println("We are same position : " + p1.isSamePosition(p2));
    System.out.println("We are same team : "+ p1.isSameTeam(p2));
  }
}
```

ตัวอย่างผลลัพธ์

```
We are same position : true
We are same team : true
```

กรณีที่ 2

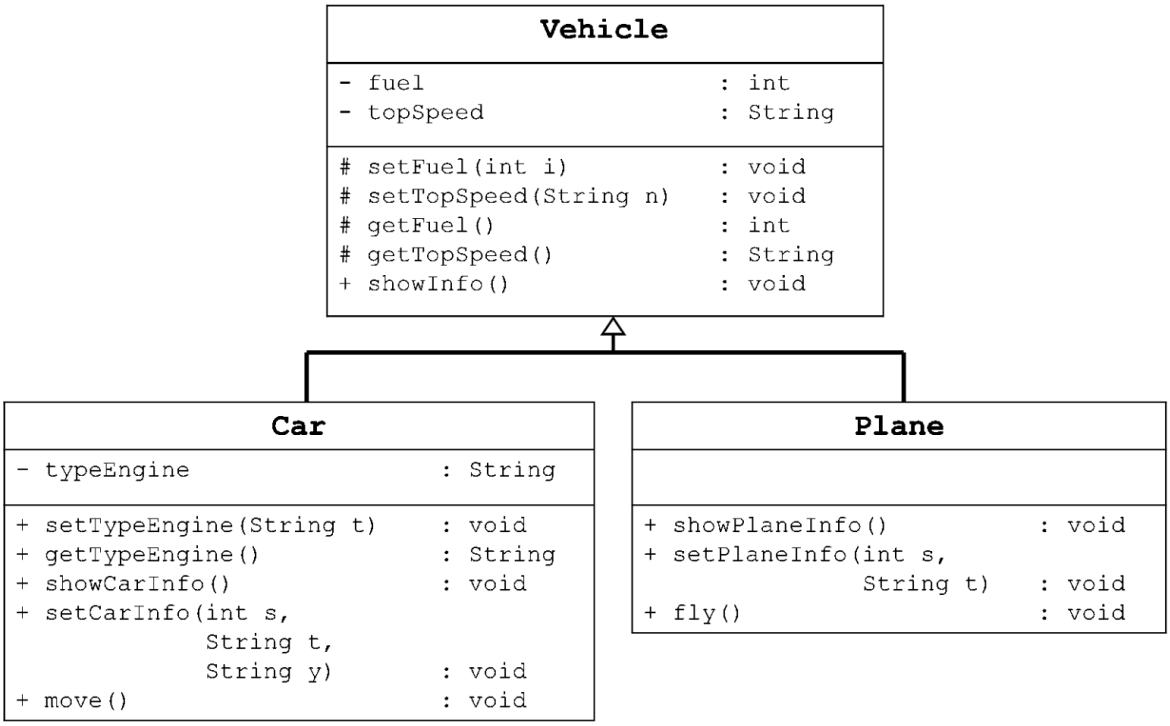
```
public class Main {
  public static void main(String[] args) {
    FootballPlayer p1 = new FootballPlayer();
    p1.setName("Harry");
    p1.setTeam("Gryffindor");
    p1.setPlayerNumber(1);
    p1.setPosition("keeper");
    FootballPlayer p2 = new FootballPlayer();
    p2.setName("Jame");
    p2.setTeam("Gryffindor");
    p2.setPlayerNumber(1);
    p2.setPosition("fullback");

    System.out.println("We are same position : " + p1.isSamePosition(p2));
    System.out.println("We are same team : "+ p1.isSameTeam(p2));
  }
}
```

ตัวอย่างผลลัพธ์

```
We are same position : false
We are same team : true
```

3. ให้นักศึกษาสร้างคลาส `Vehicle, Car` และ `Plane` ตามคลาสไดอะแกรมต่อไปนี้ โดยอาศัยหลักการห่อหุ้มและการสืบทอดตามโครงสร้างภาษาจาวา



หมายเหตุ เครื่องหมาย # ในคลาสไดอะแกรมบ่งบอกถึง Access Modifier ประเภท **protected**

โดยกำหนดให้แต่ละเมธอดมีรายละเอียดดังต่อไปนี้

คลาส **Vehicle**

- เมธอด `setFuel(int i)` จะนำค่า `i` ไปกำหนดให้แอททริบิวท์ `fuel` ขณะที่ เมธอด `setTopSpeed (String n)` จะนำค่า `n` ไปกำหนดให้แอททริบิวท์ `topSpeed`
- เมธอด `getFuel()` จะคืนค่าเป็นแอททริบิวท์ `fuel` ขณะที่ เมธอด `getTopSpeed ()` จะคืนค่าเป็นแอททริบิวท์ `topSpeed`
- เมธอด `showInfo()` จะแสดงค่าของแต่ละแอททริบิวดังตัวอย่าง

Fuel is [ค่าจากแอททริวบิวท์ **fuel] litre and Top Speed is [**ค่าจากแอททริวบิวท์ **topSpeed] m/s.**

คลาส **Car**

- เมธอด `setTypeEngine(String t)` จะนำค่า `t` ไปกำหนดให้แอททริบิวท์ `typeEngine` ขณะที่ เมธอด `getTypeEngine ()` จะคืนค่าเป็นแอททริบิวท์ `typeEngine`
- `setCarInfo(int s, String t, String y)` จะนำค่า `s` ไปกำหนดให้แอททริบิวท์ `fuel`, ค่า `t` ไปกำหนดให้แอททริบิวท์ `topSpeed` และค่า `y` ไปกำหนดให้แอททริบิวท์ `typeEngine`
- เมธอด `move()` จะดำเนินการพิมพ์ข้อความต่อไปนี้ทางจอภาพ จากนั้นค่าแอททริบิวท์ `fuel` จะลดลง 50

Move.

กรณีค่าแอททริบิวท์ `fuel` ไม่เพียงพอให้หักจะแสดงข้อความต่อไปนี้แทน

```
Please add fuel.
```

- เมธอด `showCarInfo()` จะแสดงค่าของแต่ละแอททริบิวดังตัวอย่าง

```
Car engine is [ค่าจากแอททริวบิวท์ typeEngine].
Fuel is [ค่าจากแอททริวบิวท์ fuel] litre and Top Speed is [ค่าจากแอททริวบิวท์ topSpeed] m/s.
```

คลาส **Plane**

- `setPlaneInfo(int s, String t)` จะนำค่า `s` ไปกำหนดให้แอททริบิวท์ `fuel` และค่า `t` ไปกำหนดให้แอททริบิวท์ `topSpeed`
- เมธอด `fly()` จะดำเนินการพิมพ์ข้อความต่อไปนี้ทางจอภาพ จากนั้นค่าแอททริบิวท์ `fuel` จะลดลง 200

```
Fly.
```

กรณีค่าแอททริบิวท์ `fuel` ไม่เพียงพอให้หักจะแสดงข้อความต่อไปนี้แทน

```
Please add fuel.
```

- เมธอด `showPlaneInfo()` จะแสดงค่าของแต่ละแอททริบิวดังตัวอย่าง

```
Plane detail is, Fuel is [ค่าจากแอททริวบิวท์ fuel] litre and Top Speed is [ค่าจากแอททริวบิวท์ topSpeed] m/s.
```

กำหนดโค้ดสำหรับทดสอบความถูกต้องของคลาสข้างต้นที่นักศึกษาได้พัฒนาขึ้น

กรณีที่ 1

```
class Main {
  public static void main(String[] args) {
    Plane p1 = new Plane();
    p1.setPlaneInfo(500, "High");
    p1.showPlaneInfo();
    Car c1 = new Car();
    c1.setCarInfo(500, "High", "Diesel");
    c1.showCarInfo();
   }
}
```

ตัวอย่างผลลัพธ์

```
Plane detail is, Fuel is 500 litre and Top Speed is High m/s.
Car engine is Diesel.
Fuel is 500 litre and Top Speed is High m/s.
```

กรณีที่ 2

```
public class Main {
  public static void main(String[] args) {
    Plane p1 = new Plane();
    p1.setPlaneInfo(300, "High");
    p1.showPlaneInfo();
    p1.fly();
    p1.showPlaneInfo();
    p1.fly();
    p1.showPlaneInfo();
  }
}
```

ตัวอย่างผลลัพธ์

```
Plane detail is, Fuel is 300 litre and Top Speed is High m/s.
Fly.
Plane detail is, Fuel is 100 litre and Top Speed is High m/s.
Please add fuel.
Plane detail is, Fuel is 100 litre and Top Speed is High m/s.
```

กรณีที่ 3

```
public class Main {
  public static void main(String[] args) {
    Car c1 = new Car();
    c1.setCarInfo(60, "High", "Diesel");
    c1.showCarInfo();
    c1.move();
    c1.showCarInfo();
    c1.move();
    c1.showCarInfo();
  }
}
```

ตัวอย่างผลลัพธ์

```
Car engine is Diesel.
Fuel is 60 litre and Top Speed is High m/s.
Move.
Car engine is Diesel.
Fuel is 10 litre and Top Speed is High m/s.
Please add fuel.
Car engine is Diesel.
Fuel is 10 litre and Top Speed is High m/s.
```