Programming Fundamentals II

Lap6: - Inheritance and Polymorphism
- Abstract Class and Method
- Basic Interface

Polymorphism และ Dynamic binding

เราจะนำ Class จาก Class Diagram 3 Class จากแลป5 มาพัฒนาต่อ ให้นิสิตสร้าง Folder แล้ว copy files ทั้ง 3 Class มาใส่ใน Folder ใหม่

Lab 6.1 ให้นิสิตสร้าง class ว่า Lab61GeometricTest และเขียนโปรแกรมดังนี้

```
public class Lab61GeometricTest
{
        public static void main(String[] args)
        {
                double area = 0.0;

                GeometircObject[] objs = {new Circle(5), new Rectangle(2,4)};

                for(GeometircObject obj : objs)
                {
                        area += obj.getArea();
                }
                System.out.println("Total area = "+area);
        }
}
```

ทำการคอมไพล์ทั้ง 4 คลาสใหม่ จะพบว่าเกิด compile error จงหาสาเหตุ compile error บันทึกสาเหตุของการเกิด compile error

```
Lab61GeometricTest.java:6: error: cannot find symbol
        area += obj.getArea();
                ^
  symbol:   method getArea()
  location: variable obj of type GeometricObject
1 error
```

error เพราะ ไม่พบ method getArea จาก class GeometricObject

แล้วทำการแก้ไข compile error เพื่อให้สามารถรันคลาส Lab61GeometricTest ได้เป็นผลลัพธ์ของพื้นที่รวมของทั้ง 2 object

บันทึกการแก้ไขของ compile error

นิสิตแก้ไขที่ Class ชื่อ GeometricObject

แก้ไข เพิ่ม/ลด ส่วนไหน ให้บันทึกใส่ในกล่องข้อความนี้

เพิ่ม method + getArea() : double

ให้นิสิตเพิ่ม Method getDiameter() ลงใน Class Circle ดังนี้

```
public double getDiameter()
{
	return 2*Math.PI*radius;
}
```

Lab 6.2 เราต้องการเล่น Method getDiameter ดังนั้น ให้นิสิตสร้าง class ว่า Lab62GeometricTest และเขียนโปรแกรมดังนี้

```
public class Lab62GeometricTest
{
       public static void main(String[] args)
       {
              double area = 0.0;

              GeometircObject[] objs = {new Circle(5), new Rectangle(2,4)};

              for(GeometircObject obj : objs)
              {
                 if(  obj.equals(objs[0])  )   // ตรวจสอบ Object circle inx 0 ?
                 {
                     Circle c = (Circle)obj     // Casting

                     double diameter =  c.getDiameter();

                     System.out.println("Diameter of circle = "+diameter);
                 }
              }
       }
}
```

ผลลัพท์ที่ได้จากการรันโปรแกรมข้างบน ให้บันทึกในกล่องข้อความด้านล่าง

```
Diameter of circle = 31.41592653589793
```

Abstract Class

ให้นิสิตสร้าง class ดังต่อไปนี้

1. Student 2. UnderGraduate 3. Graduate

Student.java เขียนโคดดังนี้

```
public abstract class Student
{
	private int score;

	public Student(int score)
	{
		this.setScore(score);
	}

	// Getter/Setter Score
	public int getScore()
	{
		return score;
	}
	public void setScore(int score)
	{
		this.score = score;
	}

	// Abstract Method
	public abstract String calculateGrade();

}
```

UnderGraduate.java เขียนโคดดังนี้

```
public class UnderGraduate extends Student
{
	public UnderGraduate(int score)
	{
		super(score);
	}

	@Override
	public String calculateGrade()
	{
		int score = getScore();
		if(score >= 50)
		{
			return "PASS";
		}
		else
		{
			return "FALL";
		}
	}
}
```

ให้นิสิตสร้างคลาส Graduate สืบทอดจากคลาส Student และ ให้ implement method calculateGrade โดยหลักเกณฑ์คะแนนที่ผ่านต้องได้คะแนน score >=70 จึงจะ return ค่าเป็น PASS นอกเหนือจากนั้นให้ return ค่าเป็น FAIL

```
public class Graduate extends Student
{
    public Graduate(int score){
        super(score);
    }

    @Override
    public String calculateGrade(){
        int score = getScore();
        if(score >= 70){
            return "PASS";
        }else{
            return "FAIL";
        }
    }
}
```

Lab 6.3 ให้นิสิตสร้าง Class ชื่อว่า Lab63StudentTest มีโค้ดดังต่อไปนี้

```
public class Lab63StudentTest
{
	public static void main(String[] args)
	{
		Student s1 = new Student(35);

		Student[] slist = {new UnderGraduate(50), new Graduate(50) };

		for(Student std : slist)
		{
			System.out.println(std.calculateGrade());
		}
	}
}
```

บันทึกสาเหตุเกิด Compile error

```
Lab63StudentTest.java:3: error: Student is abstract; cannot be instantiated
	Student s1 = new Student(35);
	             ^
1 error
```

error เพราะ Student ทำการ abstract ไม่สามารถ new ตัว Student ได้

ให้แก้ไขโปรแกรมให้สามารถ Run ได้ หมายเหตุ : ห้ามแก้ไขที่คลาส Student ให้แก้ที่คลาส TestStudent เท่านั้น หลังแก้ไข compile error แล้วให้ทำการรันคลาส TestStudent

บันทึกผลลัพธ์ที่ได้

```
PS D:\6530300295\week06> javac *.java
PS D:\6530300295\week06> java Lab63StudentTest
PASS
FAIL
```

Interface

ให้นิสิตสร้าง class ดังต่อไปนี้

1. Rectangle, 2. ComparableRectangle, 3. Circle

Rectangle.java เขียนโคดดังนี้

```
public class Rectangle
{
        private double width;
        private double height;
        public Rectangle(double width, double height){
                this.width = width;
                this.height = height;
        }
        public double getWidth(){
                return width;
        }
        public void setWidth(double width){
                this.width = width;
        }
        public double getHeight(){
                return height;
        }
        public void setHeight(double height){
                this.height = height;
        }
        public double getArea() {
                return width * height;
        }
}
```

ComparableRectangle.java เขียนโคดดังนี้

```
public class ComparableRectangle extends Rectangle implements Comparable
{
        public ComparableRectangle(double width, double height)
        {
                super(width, height);
        }

        public int compareTo(Object o)
        {
                if (getArea() > ((ComparableRectangle)o).getArea())
                        return 1;
                else if (getArea() < ((ComparableRectangle)o).getArea())
                        return -1;
                else
                        return 0;
        }
}
```

Circle.java เขียนโคดดังนี้

```
public class Circle
{
	private double radius;
	public Circle(double radius) {
		this.radius = radius;
	}
	public double getRadius() {
		return radius;
	}
	public void setRadius(double radius) {
		this.radius = radius;
	}
	public double getArea() {
		return radius * radius * Math.PI;
	}
}
```

Class ComparableCircle เป็น Class ที่ implement interface Comparable โดยมี Method compareTo ไว้เปรียบเทียบ object ของคลาส ComparableCircle ดังนั้น นิสิตจงพัฒนา Class ComparableCircle ต่อไปนี้ให้สมบูรณ์เพื่อนำไปใช้งานโดย Class Max และ Class Lab64MaxTest ข้างล่าง

เขียนโค้ด ComparableCircle.java ที่นิสิตพัฒนาลงกล่องข้อความด้านล่าง

```
public class ComparableCircle extends Circle implements Comparable
{
    public ComparableRectangle(double width, double height){
        super(width, height);
    }

    public int compareTo(Object o){
        if(getArea() > ((ComparableRectangle)o).getArea())
            return 1;
        else if(getArea() < ((ComparableRectangle)o).getArea())
            return -1;
        else
            return 0;
    }
}
```

Class Max.java มี Method max เพื่อไว้เปรียบเทียบระหว่าง Obj1 และ Obj2 ดัง source codes ต่อไปนี้

```
public class Max
{
	/** Return the maximum of two objects */
	public static Comparable max (Comparable o1, Comparable o2) {
		if (o1.compareTo(o2) > 0)
			return o1;
		else
			return o2;
	}
}
```

Lab 6.4 ให้นิสิตสร้าง Class ชื่อว่า Lab64MaxTest มีการสร้าง object จาก Class ComparableRectangle แล้วส่งต่อให้กับ method max ของ Class Max และนิสิตจงเพิ่ม source codes ส่วนเรียกใช้งาน Class ComparableCircle ที่พัฒนาขึ้น ลงใน Class Lab58MaxTest ต่อไปนี้

```
public class Lab64MaxTest
{
        public static void main(String[] args)
        {
                ComparableRectangle rectangle1 = new ComparableRectangle(4, 5);     //20
                ComparableRectangle rectangle2 = new ComparableRectangle(3, 6);     //18
                Rectangle r = (Rectangle)Max.max(rectangle1, rectangle2);
                System.out.println(r.getArea());

                /* เพิ่ม Code เพื่อทดสอบ Class ComparableCircle ด้วยตนเอง */

                ComparableCircle circle1 = new ComparableCircle(5);
                ComparableCircle circle2 = new ComparableCircle(3);
                Circle c = (Circle)Max.max(circle1, circle2);
                System.out.println(c.getArea());

        }
}
```

หลังจากพัฒนา Class Lab64MaxTest แล้วจงทำการรันและศึกษาการทำงานต่างๆ ของ Class ก่อนหน้าทั้งหมด

Homework#6

ข้อ1 ในงานชิ้นนี้เราจะสร้าง Invoice ขึ้น โดยให้ตัว Invoice มีรายการสินค้าได้หลายรายการ รายการสินค้าจะแยกเป็นคลาส LineItem และ Product ไปต่างหาก นอกจากนี้ในแต่ละ Invoice ยังมีชื่อของลูกค้ากำกับด้วย ดังนั้นในงานนี้เราจะสร้างคลาสทั้งหมด 5 คลาสดังนี้

1. Invoice (ใบแจ้งรายการสินค้า)

ประกอบด้วยคุณสมบัติดังต่อไปนี้

1. id เป็น String แทนรหัสของใบแจ้งรายการสินค้า
2. customer เป็น Customer แทนลูกค้าที่เป็นผู้ซื้อของ Invoice นี้
3. items เป็นอาเรย์ของ LineItem แทนรายการซื้อแต่ละรายการ

และมีเมทอดดังต่อไปนี้

4. Invoice(String id, Customer customer)
5. void addItem(Product product, int quantity)
   สร้าง LineItem ใหม่จาก product และ quantity และใส่เข้าไปในรายการซื้อ
6. String getId()
7. Customer getCustomer()
8. LineItem getLineItem(int i) ส่งค่ากลับเป็นรายการซื้อลำดับที่ i โดยนับรายการแรกเป็นรายการที่ 0
9. double getTotalPrice() ส่งค่ากลับเป็นราคารวมของทุกรายการซื้อ
10. void print() แสดงข้อมูลของ Invoice ในรูปแบบดังตัวอย่างด้านล่าง

*** กำหนดให้ใช้ ArrayList ในการเก็บ LineItem แต่ละรายการ

2. Customer (ลูกค้า)

ประกอบด้วยคุณสมบัติดังต่อไปนี้

1. id เป็น String แทนรหัสลูกค้า
2. firstName เป็น String แทนชื่อ
3. lastName เป็น String แทนนามสกุล

และมีเมทอดดังต่อไปนี้

4. Customer(String id, String firstName, String lastName)
5. String getId()
6. String getFirstName()
7. String getLastName()

3. LineItem (รายการซื้อ)

ประกอบด้วยคุณสมบัติดังต่อไปนี้

1. item เป็น Product แทนสินค้ารายการนั้น

2. quantity เป็น int แทนจำนวนสินค้าในรายการ

และมีเมทอดดังต่อไปนี้

3. LineItem(Product product, int quantity) ถ้า quantity เป็นลบ ให้กำหนดให้เป็น 0
4. Product getProduct()
5. int getQuantity()
6. double getTotalPrice() ส่งค่ากลับเป็นราคารวมของสินค้ารายการนี้

4. Product (สินค้า)

ประกอบด้วยคุณสมบัติดังต่อไปนี้

1. id เป็น String แทนรหัสสินค้า
2. name เป็น String แทนชื่อสินค้า
3. price เป็น double แทนราคาสินค้า

และมีเมทอดดังต่อไปนี้

4. Product(String id, String name, double price) ถ้า price เป็นลบ ให้กำหนดให้เป็น 0.0
5. String getId()
6. String getName()
7. void setPrice(double price) ถ้า price เป็นลบ ให้กำหนดให้เป็น 0.0
8. double getPrice()

5. InvoiceTest (ตัวทดสอบโปรแกรม)

ตัวทดสอบจะต้องสร้าง Invoice ขึ้นมาอย่างน้อย 3 Invoice ลูกค้า (Customer) อย่างน้อย 2 คน สินค้า (Product) อย่างน้อย 5 ชนิด Invoice มากกว่า 1 อันอาจเป็นของลูกค้าคนเดียวกันได้ และแต่ละ Invoice จะต้องมีรายการซื้ออย่างน้อย 3 รายการ และแสดงผลลัพธ์โดยการเรียกเมทอด print ของ Invoice แต่ละอันออกมา รูปแบบการแสดงผลของ Invoice แต่ละอันจะอยู่ในแบบดังตัวอย่างนี้ (อย่าลืมว่ามี Invoice 3 อัน เพราะฉะนั้นจะต้องมีการแสดงผลในลักษณะตามตัวอย่างทั้งหมด 3 ครั้งที่แตกต่างกัน)

ตัวอย่างผลลัพท์ เมื่อทำการรันโปรแกรม

```
INVOICE: #<invoice id>
CUSTOMER: <customer name>

ITEMS:
1. <item name> x <item quantity> = <price>
2. ...
...

TOTAL: <total price>
```

หมายเหตุ1 คุณสมบัติทั้งหมดต้องเป็น private และให้เมทอดทั้งหมดเป็น public

หมายเหตุ2 สำหรับแบบฝึกหัด ให้ทำการ comment ในโปรแกรม ในส่วนของ statement หลักๆของโค้ดที่นิสิตเขียน ถ้าไม่มีการ comment จะถือว่าโค้ดไม่ครบสมบูรณ์

HomeWork #6

```java
import java.util.ArrayList;

public class Invoice{
    //instance variables
    private String id;
    private Customer customer;
    private ArrayList<LineItem> items = new ArrayList<LineItem>();

    //Invoice constructor
    public Invoice(String id, Customer customer){
        this.id = id;
        this.customer = customer;
    }

    //method for adding an Items
    public void addItem(Product product, int quantity){
        this.items.add(new LineItem(product, quantity));
    }

    //method for get id
    public String getId(){
        return this.id;
    }

    //method for get customer
    public Customer getCustomer(){
        return this.customer;
    }

    //method for get lineitem
    public LineItem getLineItem(int i){
        return this.items.get(i);
    }

    //method for get total price
    public double getTotalPrice(){
        double sum = 0;
        for(int i = 0; i < this.items.size(); i++){
            sum += this.items.get(i).getTotalPrice();
        }
        return sum;
    }

    //print method
    public void print(){
        System.out.println("INVOICE: #" + getId());
        System.out.println("CUSTOMER: " + getCustomer().getFirstName() + " " + getCustomer().getLastName());
        System.out.println("ITEMS:");
        for(int i = 0; i < this.items.size(); i++){
            System.out.println((i+1) + ". " + this.items.get(i).getProduct().getName() + " x " + this.items.get(i).getQuantity() + " = " + this.items.get(i).getTotalPrice());
        }
        System.out.println("TOTAL: " + getTotalPrice() + "\n");
    }
}
```

```java
public class Customer {
    //instance variables
    private String id;
    private String firstName;
    private String lastName;

    //Customer constructor
    public Customer(String id, String firstName, String lastName){
        this.id = id;
        this.firstName = firstName;
        this.lastName = lastName;
    }

    //method for get id
    public String getId(){
        return this.id;
    }

    //method for get first name
    public String getFirstName(){
        return this.firstName;
    }

    //method for get last name
    public String getLastName(){
        return this.lastName;
    }
}
```

```java
public class LineItem {
    //instance variables
    private Product item;
    private int quantity;

    //LineItem constructor
    public LineItem(Product product, int quantity){
        this.item = product;
        this.quantity = quantity;
        if(this.quantity < 0){
            this.quantity = 0;
        }
    }

    //method for get product
    public Product getProduct(){
        return this.item;
    }

    //method for get quantity
    public int getQuantity(){
        return this.quantity;
    }

    //method for get total price
    public double getTotalPrice(){
        return this.quantity * getProduct().getPrice();
    }
}
```

```java
public class Product {
    //instance variables
    private String id;
    private String name;
    private double price;

    //Product constructor
    public Product(String id, String name, double price){
        this.id = id;
        this.name = name;
        this.price = price;
        if(this.price < 0.0){
            this.price = 0.0;
        }
    }

    //method for get id
    public String getId(){
        return this.id;
    }

    //method for get name
    public String getName(){
        return this.name;
    }

    //method for set the price
    public void setPrice(double price){
        this.price = price;
        if(this.price < 0.0){
            this.price = 0.0;
        }
    }

    //method for get price
    public double getPrice(){
        return this.price;
    }
}
```

```java
/*
 * Written by Nititorn Kijprasopchok
 * ID: 6530300295
 */

public class InvoiceTest {
    public static void main(String[] args){
        //creating customer object
        Customer customer1 = new Customer("1", "John", "Kater");
        Customer customer2 = new Customer("2", "Tom", "Lolan");
        Customer customer3 = new Customer("3", "Jacky", "Chan");

        //creating invoice object
        Invoice invoice1 = new Invoice("01", customer1);
        Invoice invoice2 = new Invoice("02", customer2);
        Invoice invoice3 = new Invoice("03", customer3);

        //creating product object
        Product product1 = new Product("001", "Table", 1000.0);
        Product product2 = new Product("002", "Chair", 250.0);
        Product product3 = new Product("003", "Sofa", 2500.0);
        Product product4 = new Product("004", "Bed", 3000.0);
        Product product5 = new Product("005", "TV", 5000.0);

        //adding items for first invoice
        invoice1.addItem(product1, 1);
        invoice1.addItem(product2, 4);
        invoice1.addItem(product5, 1);

        //adding items for second invoice
        invoice2.addItem(product3, 1);
        invoice2.addItem(product4, 1);
        invoice2.addItem(product5, 1);

        //adding items for thrid invoice
        invoice3.addItem(product1, 2);
        invoice3.addItem(product2, 8);
        invoice3.addItem(product3, 2);
        invoice3.addItem(product4, 1);
        invoice3.addItem(product5, 1);

        //print invoice from print method
        invoice1.print();
        invoice2.print();
        invoice3.print();
    }
}

```

```
PS D:\Coding_lab\java\6530300295\week06\Invoice> javac InvoiceTest.java
PS D:\Coding_lab\java\6530300295\week06\Invoice> java InvoiceTest
INVOICE: #01
CUSTOMER: John Kater
ITEMS:
1. Table x 1 = 1000.0
2. Chair x 4 = 1000.0
3. TV x 1 = 5000.0
TOTAL: 7000.0

INVOICE: #02
CUSTOMER: Tom Lolan
ITEMS:
1. Sofa x 1 = 2500.0
2. Bed x 1 = 3000.0
3. TV x 1 = 5000.0
TOTAL: 10500.0

INVOICE: #03
CUSTOMER: Jacky Chan
ITEMS:
1. Table x 2 = 2000.0
2. Chair x 8 = 2000.0
3. Sofa x 2 = 5000.0
4. Bed x 1 = 3000.0
5. TV x 1 = 5000.0
TOTAL: 17000.0
```