ชื่อ-นามสกุล....นายณัฐเชษฐ ทรัพย์ชัย....รหัสประจำตัวนักศึกษา....6609062636151.... *ปีการศึกษา 2567*

แบบฝึกหัดปฏิบัติการคาบที่ 6: Inheritance

คำสั่ง

1. ให้ศึกษาการสืบทอดข้อมูล Inheritance จากตัวอย่างต่อไปนี้

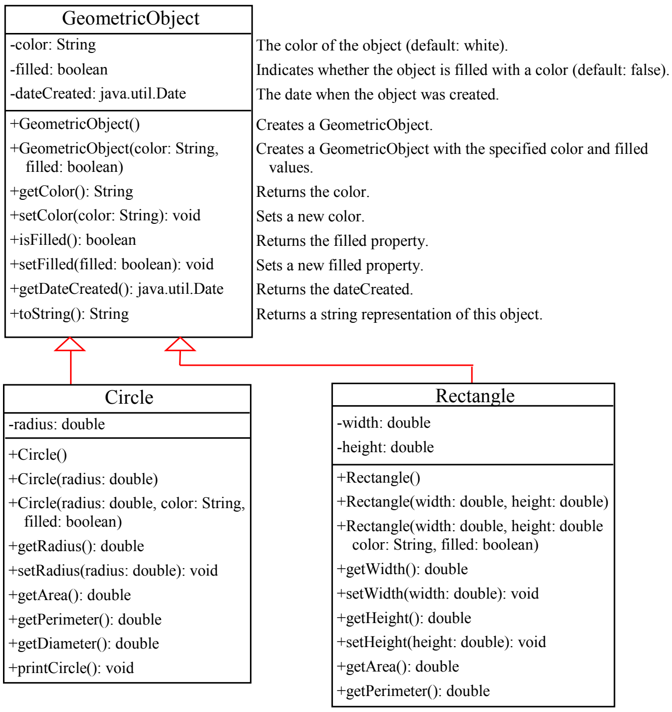
การเขียนโปรแกรมเชิงวัตถุ จะอนุญาตให้สามารถสร้างคลาสใหม่จากคลาสเดิมที่เคยมีอยู่แล้วซึ่งเรียกเทคนิคนี้ว่าการสืบทอด (Inheritance) การสืบทอด (Inheritance) เป็นเทคนิคที่สำคัญมากสำหรับการนำซอฟต์แวร์กลับมาใช้ใหม่ (Reuse software)

Super class และ Sub class

เทคนิคการสืบทอดข้อมูลอนุญาตให้กำหนดคลาสทั่วไปที่เรียกว่า General class หรือ Super class และภายหลังสามารถขยาย (extends) คลาสดังกล่าวไปเป็นคลาสที่มีความจำเพาะเจาะจงได้ specialized class หรือ sub class เราสามารถใช้คลาสเพื่อจำลองวัตถุที่มีชนิดข้อมูลเดียวกันได้ คลาสต่าง ๆ กันอาจจะมีคุณสมบัติและพฤติกรรมที่เหมือนกัน ที่สามารถทำให้อยู่ในรูปทั่วไปได้และสามารถใช้ร่วมกับคลาสอื่น ๆ ได้ เรายังสามารถกำหนดคลาสที่มีคุณสมบัติจำเพาะ ที่สืบทอดหรือขยายมาจากคลาสทั่วไปหรือ Generalized class ได้โดยที่คลาสที่มีความจำเพาะเหล่านั้นสืบทอดคุณสมบัติและเมธอดมาจากคลาสทั่วไป

ตัวอย่าง

สมมติต้องการคลาสวงกลม สี่เหลี่ยมและสามเหลี่ยม คลาสต่าง ๆ เหล่านี้จะมีคุณลักษณะบางอย่างที่คล้ายกัน การออกแบบคลาสเพื่อไม่ให้ใช้ตัวแปรซ้ำ และทำให้ง่ายต่อการแก้ไข วิธีที่ดีที่สุดคือการใช้หลักการของการสืบทอดข้อมูล โดยการสร้างคลาสที่คลาสทุกคลาสมีคุณสมบัติเดียวกันเรียกว่า Superclass ซึ่งจากตัวอย่างคือ GeometricObject หลังจากนั้นเมื่อต้องการสร้างวงกลม Circle ก็ให้สืบทอดคุณสมบัติจากคลาส Geometric และเพิ่มคุณสมบัติเฉพาะตัวของวงกลมเข้าไป



ชื่อ-นามสกุล......นายณโรจน์เวช......ทรัพย์มีชัย..........รหัสประจำตัวนักศึกษา....6604062636151...... ปีการศึกษา 2567

1. ให้ศึกษาและทดลองพิมพ์ Code ตัวอย่างการสืบทอดคลาสจากตัวอย่างต่อไปนี้

```
public abstract class GeometricObject {
      private String color = "white";
      private boolean filled;
      /** Default constructor */
      protected GeometricObject() {
      }
      /** Convenience constructor */
      protected GeometricObject(String color, boolean filled) {
            this.color = color;
            this.filled = filled;
      }
      public String getColor() {
            return color;
      }
      public void setColor(String color) {
            this.color = color;
      }
      public boolean isFilled() {
            return filled;
      }
      public void setFilled(boolean filled) {
            this.filled = filled;
      }
      public abstract double findArea();
      public abstract double findPerimeter();
}
public class Circle extends GeometricObject {
      private double radius;

      /** Default constructor */
      public Circle() {
            this(1.0);
      }

      /** Radius convenience constructor */
      public Circle(double radius) {
            this(radius, "white", false);
      }

      /** Convenience constructor for all properties */
      public Circle(double radius, String color, boolean filled) {
            super(color, filled);
            this.radius = radius;
      }

      /**
       * Return the radius
       * @return radius Current radius of Circle
       */
      public double getRadius() {
            return radius;
      }

      /**
       * Set the radius of the circle
       */
```

```java
	public void setRadius(double radius) {
		this.radius = radius;
	}

	/**
	 * Returns the area of the current circle
	 * Implemention of abstract method in GeometricObject
	 * @return area of the circle
	 */
	public double findArea() {
		return radius * radius * Math.PI;
	}

	/**
	 * Returns the perimeter of the current circle
	 * Implemention of abstract method in GeometricObject
	 * @return perimeter of the circle
	 */
	public double findPerimeter() {
		return 2 * radius * Math.PI;
	}

	/**
	 * Provide a string representation of the object
	 */
	public String toString() {
		return "Circle: radius = " + radius;
	}
}
```

1.1 **ให้อธิบายลำดับการเรียกใช้** constructor **เมื่อสร้างวัตถุจากคลาส** Circle

1. GeometricObject

2. Circle

1.2 **ให้สร้างคลาส** Rectangle **ตาม** UML Class diagram **ด้านบน**

```
class Rectangle extends GeometricObject{
    private double width;
    private double height;

    public Rectangle(){
        this.width =0;
        this.height=0;
        super();
    }
    public Rectangle(double width, double height){
        super();
        this.width = width;
        this.height = height;
    }
```

```
public Rectangle(double width, double height, String color, boolean filled)
{
    super(color, filled);
    this.width = width;
    this.height = height;
}

public double getWidth() {
    return this.width;
}
public void setWidth(double width) {
    this.width = width;
}
public double getHeight() {
    return this.height;
}
public void setHeight(double height) {
    this.height = height;
}
public double setArea() {
    return this.width * this.height;
}
public double setPerimeter() {
    return 2*(this.width + this.height);
}
}
```

ชื่อ-นามสกุล [illegible] ทรัพย์มั่นชัย รหัสประจำตัวนักศึกษา 6604062636151 *ปีการศึกษา 2567*

2. **จงหาข้อผิดพลาดของส่วนของโปรแกรมต่อไปนี้**

```
public class Circle{
    private double radius;

    public Circle(double radius){
        radius=radius;
    }
    public double getRadius(){
        return radius;
    }
    public double findArea(){
        return radius*radius*Math.PI;
    }
}
class Cylinder extends Circle{
    private double length;

    Cylinder(double radius, double length){
        Circle(radius);
        length=length;
    }
}
```

ต้องเป็น super

3. **จงหาผลลัพธ์การรันโปรแกรมต่อไปนี้**

3.1

`class A{`
`   public A(){`
`      System.out.println("The no-arg constructor of A is invoked");`
`   }`
`}`
`class B extends A{`
`   public B(){`
 	
`   }`
`}`
`public class C{`
`   public static void main(String[] args){`
`     B b =new B();`
`   }`
`}`
	ผลลัพธ์การรันโปรแกรม
	The no-arg construction of A is invoked

ชื่อ-นามสกุล [illegible] รหัสประจำตัวนักศึกษา 6609062636151 *ปีการศึกษา 2567*

3.2

Animal.java:
```java
public class Animal {
        public Animal() {
                System.out.println("A new animal has been created!");
        }

        public void sleep() {
                System.out.println("An animal sleeps...");
        }

        public void eat() {
                System.out.println("An animal eats...");
        }
}
```

Bird.java:
```java
public class Bird extends Animal {
        public Bird() {
                super();
                System.out.println("A new bird has been created!");
        }

        @Override
        public void sleep() {
                System.out.println("A bird sleeps...");
        }

        @Override
        public void eat() {
                System.out.println("A bird eats...");
        }
}
```

Dog.java:
```java
public class Dog extends Animal {
        public Dog() {
                super();
                System.out.println("A new dog has been created!");
        }

        @Override
        public void sleep() {
                System.out.println("A dog sleeps...");
        }

        @Override
        public void eat() {
                System.out.println("A dog eats...");
        }
}
```

MainClass.java:
```java
public class MainClass {
        public static void main(String[] args) {
                Animal animal = new Animal();
                Bird bird = new Bird();
                Dog dog = new Dog();

                System.out.println();

                animal.sleep();
                animal.eat();

                bird.sleep();
                bird.eat();

                dog.sleep();
                dog.eat();
        }
}
```

ผลลัพธ์การรันโปรแกรม

A new animal has been created!
A new animal has been created!
A new brid has been created!
A new animal has been created!
A new dog has been created!

An animal sleeps...
An animal eats...
An brid sleeps...
An brid eats...
A dog sleeps...
A dog eats...

ชื่อ-นามสกุล.....นาย[illegible]........ทรัพย์[illegible].........รหัสประจำตัวนักศึกษา.....6609062636151.....   *ปีการศึกษา 2567*

4. จากโปรแกรมด้านล่างต่อไปนี้ ข้อใดคือ **Overriding** ที่สามารถนำมาเติมลงในบรรทัดที่ 9 ของโปรแกรมได้โดยไม่ทำให้เกิดข้อผิดพลาดขึ้นกับการทำงานของโปรแกรม อธิบายเหตุผลประกอบ

```
 1 class SuperClass{
 2    private int num=1;
 3    protected int getNumber(){
 4       return num;
 5    }
 6 }
 7 class Subclass extends SuperClass{
 8    private int num=10;
 9    //overiding method
10    public static void main(String[] args){
11      Subclass s= new Subclass();
12      System.out.println(s.getNumber());
13    }
14 }
```

โปรแกรม	สามารถนำมาเติมลงในบรรทัดที่ 9 ของโปรแกรมได้โดยไม่ทำให้เกิดข้อผิดพลาด**หรือไม่**	อธิบายเหตุผลประกอบ
`protected int getNumbers(){ return num+5; }`	ไม่ได้	เพราะการ overriding superclass นั้น ต้องมีชื่อ Method เหมือนเดียวกัน
`protected long getNumber(){ return num+5; }`	ไม่ได้	return type ไม่เหมือน superclass ของ Method นั้น
`protected int getNumber(){ return num+5; }`	ได้	ทุกอย่างตรงตามกฎ
`public int getNumber(){ return num+5; }`	ได้	สามารถทำให้เข้าถึงในวงกว้างขึ้นได้แต่ลดลงไม่ได้
`protected int getNumber(int num){ return num+5; }`	ไม่ได้	เพราะ superclass method นั้น ไม่ได้รับ parameter
`int getNumber(){ return num+5; }`	ไม่ได้	เพราะไม่มี method header modifier
`private int getNumber(){ return num+5; }`	ไม่ได้	ไม่สามารถลดการเข้าถึงของ superclass ได้

ชื่อ-นามสกุล....นายณัฐชุเลณ ทรัพย์มีชัย....รหัสประจำตัวนักศึกษา....6604062636151.... *ปีการศึกษา 2567*

5. [**Application**] จงเขียนโปรแกรมเพื่อแสดงรายละเอียดของคลาส **Account** ที่ประกอบด้วยสมาชิกต่อไปนี้

-ตัวแปร **private** ชนิดข้อมูล **int** ชื่อ **id** สำหรับเก็บหมายเลขบัญชี

-ตัวแปร **private** ชนิดข้อมูล **double** ชื่อ **balance** สำหรับเก็บยอดเงินคงเหลือ

-ตัวแปร **private** ชนิดข้อมูล **double** ชื่อ **annualInterestRate** สำหรับเก็บอัตราดอกเบี้ย

-ตัวแปร **private** ชนิดข้อมูล **Date** ชื่อ **dateCreated** สำหรับเก็บวันที่ที่บัญชีถูกสร้าง

- **constructor** ที่ไม่มี **argument** สำหรับการสร้างบัญชีแบบ **default**

- **constructor** ที่มี **argument** สำหรับการสร้างบัญชีแบบระบุเลขที่บัญชี และยอดเงินเริ่มต้น

- **accessor method(get)** และ **mutator method(set)** สำหรับตัวแปร **id, balance, annualInterestRate, dateCreated**

-เมธอด **getMonthlyInterestRate()** ที่คืนอัตราดอกเบี้ยรายเดือน

-เมธอด **getMonthlyInterest()** ที่คืนดอกเบี้ยรายเดือน

-เมธอด **withdraw()** ที่ถอนเงินตามจำนวนที่ระบุ

-เมธอด **deposit()** ที่ฝากเงินตามจำนวนที่ระบุ

วาดคลาสไดอะแกรมและเขียนส่วนของ **Client** สำหรับเรียกใช้คลาส **Account** โดยสร้างอ๊อบเจ็คของบัญชีเลขที่ (**ID**) **1122** ยอดเงินเปิดบัญชีคือ **20000** และอัตราดอกเบี้ยคือ **4.5 %** หลังจากนั้นให้ใช้ **withdraw method** สำหรับถอนเงิน **2500** บาท และใช้ **deposit method** สำหรับฝากเงิน **3000** บาท และโปรแกรมสามารถแสดงยอดเงินคงเหลือและอัตราดอกเบี้ยรายเดือนได้

6. [Application] จากคลาส **Account** ให้เพิ่มสมาชิกของคลาส **Account** เพิ่มเติมดังต่อไปนี้

-ตัวแปร **private** ชนิดข้อมูล **Date** ชื่อ **dateCreated** สำหรับเก็บวันที่ที่บัญชีถูกสร้าง โดยชนิดข้อมูล **Date** ให้สร้างเองโดยประกอบด้วยสมาชิกภายในดังนี้

- **Constructor** ที่ไม่มี **argument** สำหรับการสร้างวันที่แบบ **default**
- **Constructor** ที่มี **argument** สำหรับการสร้างวันที่ที่ระบุวัน เดือน ปี
- ตัวแปร **private** ชนิดข้อมูล **int** ชื่อ **day**
- ตัวแปร **private** ชนิดข้อมูล **String** ชื่อ **month**
- ตัวแปร **private** ชนิดข้อมูล **int** ชื่อ **year**

-ตัวแปร **private** ชนิดข้อมูล **Person** ชื่อ **objPerson** สำหรับเก็บประวัติของลูกค้า โดยชนิดข้อมูล **Person** ให้สร้างเองโดยกำหนดให้ประกอบด้วยสมาชิกภายในดังนี้

- **Constructor** ที่ไม่มี **argument** สำหรับการสร้างลูกค้าแบบ **default**
- **Constructor** ที่มี **argument** สำหรับการสร้างลูกค้าที่ระบุชื่อ นามสกุล
- ตัวแปร **private** ชนิดข้อมูล **String** ชื่อ **name**
- ตัวแปร **private** ชนิดข้อมูล **String** ชื่อ **surname**
- ตัวแปร **private** ชนิดข้อมูล **int** ชื่อ **age**
- ตัวแปร **private** ชนิดข้อมูล **Date** ชื่อ **bDate**

-สร้าง **accessor method(get)** และ **mutator method(set)** สำหรับตัวแปรให้เหมาะสมในแต่ละคลาส

-เมธอด **transferMoney(Account acc1, amount)** ที่ทำหน้าที่ในการโอนเงินจากบัญชีปัจจุบันไปบัญชี **acc1**

-แก้ไขเมธอด **getMonthlyInterest()** ที่คืนดอกเบี้ยรายเดือนโดยคิดจากวันเปิดบัญชี

-เมธอด **toString()** สำหรับแสดงรายละเอียดภายในของคลาสแต่ละคลาส

จากนั้นให้สร้างคลาสใหม่เพิ่มอีก 2 คลาส ชื่อ SavingAccount และ FixAccount โดยสืบทอดมาจากคลาส Account โดยให้แก้ไขเมธอด transferMoney() ในคลาส SavingAccount โดยให้เพิ่มค่าธรรมเนียมที่จะต้องจ่ายให้กับทางธนาคาร 20 บาทต่อ 1 รายการ

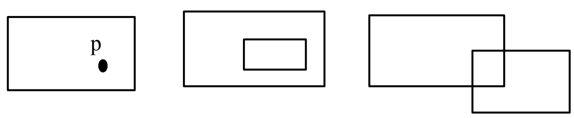
แก้ไขเมธอด transferMoney() ในคลาส FixAccount โดยบัญชีประเภทดังกล่าวจะไม่สามารถโอนเงินได้ โดยระบบต้องแจ้งต่อผู้ใช้

6.1 วาดคลาสไดอะแกรมและเขียนส่วนของ Client สำหรับเรียกใช้คลาส SavingAccount โดยสร้างออบเจ็คของบัญชีเลขที่ (ID) 1123 ยอดเงินเปิดบัญชีคือ 20000 และอัตราดอกเบี้ยต่อปีคือ 4.5 % หลังจากนั้นให้ป้อนข้อมูลชื่อ นามสกุล อายุ วันเดือนปีเกิด ของผู้เปิดบัญชี หลังจากนั้นให้ใช้ withdraw() สำหรับถอนเงิน 2500 บาท และใช้ deposit() สำหรับฝากเงิน 3000 บาท จากนั้นให้ทำการโอนเงินโดยใช้ () สำหรับการโอนเงินจากบัญชี ID 1123 ไปยังบัญชี ID 1100 และโปรแกรมสามารถแสดงยอดเงินคงเหลือและดอกเบี้ยรายเดือนได้

6.2 วาดคลาสไดอะแกรมและเขียนส่วนของ Client สำหรับเรียกใช้คลาส FixAccount โดยสร้างออบเจ็คของบัญชีเลขที่ (ID) 1124 ยอดเงินเปิดบัญชีคือ 20000 และอัตราดอกเบี้ยต่อปีคือ 7 % หลังจากนั้นให้ป้อนข้อมูลชื่อ นามสกุล อายุ วันเดือนปีเกิด ของผู้เปิดบัญชี หลังจากนั้นให้ใช้ withdraw() สำหรับถอนเงิน 2500 บาท และใช้ deposit() สำหรับฝากเงิน 3000 บาท จากนั้นให้ทำการโอนเงินโดยใช้ transferMoney() สำหรับการโอนเงินจากบัญชี ID 1124 ไปยังบัญชี ID 1100 และโปรแกรมสามารถแสดงยอดเงินคงเหลือและดอกเบี้ยรายเดือนได้ ในการถอนเงินโปรแกรมจะต้องเช็คว่าปีที่ถอนต้องมากกว่าปีที่ฝาก 1 ปีจึงจะทำการถอนได้ ส่วนเมื่อเรียกใช้ transferMoney() โปรแกรมจะต้องแจ้งผู้ใช้ว่าไม่สามารถโอนเงินได้

7. [Algorithms] จากคลาส Rectangle ในตัวอย่าง ให้สร้างคลาส MyRectangle2D ที่ประกอบไปด้วยรายละเอียดต่อไปนี้:

- •ตัวแปรชนิด double ชื่อ x และ y ที่กำหนดจุดศูนย์กลางของสี่เหลี่ยม โดยมีเมธอด get และ set สำหรับกำหนดค่าและดึงค่า
- •ตัวแปรชนิด double ชื่อ width and height โดยมีเมธอด get และ set สำหรับกำหนดค่าและดึงค่า
- •no-arg constructor สำหรับการสร้างสี่เหลี่ยมแบบ default rectangle ที่มีการกำหนด (0, 0) สำหรับตัวแปร (x, y) และ 1 สำหรับ width and height.
- •constructor สำหรับการสร้างสี่เหลี่ยมแบบกำหนดค่า x, y, width, and height.
- •เมธอด getArea() ที่คืนพื้นที่ของสี่เหลี่ยม.
- •เมธอด getPerimeter() ที่คืนเส้นรอบวงของสี่เหลี่ยม.
- •เมธอด contains(double x, double y) ที่คืนค่า true ถ้ามีจุด point (x, y) อยู่ภายในสี่เหลี่ยมดังรูปที่ 1(a).
- •เมธอด contains(MyRectangle2D r) ที่คืนค่า true ถ้ามีสี่เหลี่ยม rectangle อยู่ภายในสี่เหลี่ยม ดังรูปที่1(b).
- •เมธอด overlaps(MyRectangle2D r) ที่คืนค่า true ถ้ามีสี่เหลี่ยม rectangle) ที่มีบางส่วนซ้อนทับกัน ดังรูปที่1(c).



(a) (b) (c)

รูปที่ 1

(a) จุดอยู่ในสี่เหลี่ยม. *(b)* สี่เหลี่ยมอยู่ภายในสี่เหลี่ยมตัวอื่น. *(c)* สี่เหลี่ยมที่มีบางส่วนซ้อนทับกัน

- วาดคลาสไดอะแกรมของคลาส MyRectangle2D.
- สร้างเมธอด getArea(), getPerimeter(), contains(double x, double y), contains(MyRectangle2D r), และ overlaps(MyRectangle2D r).

8. ให้นักศึกษา นิยาม **Class** ชื่อ **Rectangle** และ **Line** โดยแต่ละ **class** ประกอบไปด้วย

Rectangle	Line
Attributes width, height, และ คู่ลำดับ (x, y) แสดงมุมซ้ายมือด้านบนของ Rectangle	Attributes คู่ลำดับ (x1,y1) และ (x2,y2) แสดงจุดเริ่มต้นและจุดสุดท้ายของ Line
No-arg constructor	No-arg constructor
Constructor ที่กำหนดค่าให้แก่ width, height, (x, y)	Constructor ที่กำหนดค่าให้แก่ (x1,y1) และ (x2,y2)
Method getArea(Rectangle a) // return พื้นที่ของ Rectangle	Method getLong(Line a) // return ความยาวของ Line

class Rectangle
{
 	
}
	class Line
	{
	
	}

ให้นักศึกษาเขียนโปรแกรม method ต่อไปนี้

-Method contains(Line a, Rectangle b) ที่ return 1 ถ้าค่า object a อยู่ภายใน object b และ return 0 ถ้าค่า object a ไม่อยู่ภายใน object b

-Method cross(Line a, Line b) ที่ return 1 ถ้า object a overlap กับ object b และ return 0 ถ้า object a ไม่ overlab กับ object b

-Method overlaps(Rectangle a, Rectangle b) ที่ return 1 ถ้า object a overlap กับ object b และ return 0 ถ้า object a ไม่ overlab กับ object b

- Method distance**(**Line a, Rectangle b**) ที่** return **ระยะห่างระหว่างจุดกึ่งกลาง** Rectangle **กับ จุดกึ่งกลาง** Line

Method contains**(**Line a, Rectangle b**)**