Programming Fundamentals II

Lap5: - Inheritance and Polymorphism
- Abstract Class and Method
- Basic Interface

1.1 การใช้คำสั่ง super() และคำสั่ง super.

Lab 5.1 ให้นิสิตสร้าง class ดังต่อไปนี้ 1. Employee.java 2. Faculty.java

Employee.java เขียนโคดดังนี้

```
public class Employee
{
	public Employee(String s)
	{
		System.out.println(s);
	}
}
```

Faculty.java เขียนโคดดังนี้

```
public class Faculty extends Employee
{
	public Faculty()
	{
		System.out.println("Faculty's no-arg constructor is invoked");
	}
}
```

Lab 5.1 ให้นิสิตสร้าง class ชื่อ Lab51Super และให้เขียนโคดดังนี้

```
public class Lab51Super
{
	public static void main(String[] args)
	{
		new Faculty();
	}
}
```

จงหาสาเหตุของการเกิด error ของไฟล์ Faculty.java เกิดจากสาเหตุอะไร หลังจากนั้นให้ทำการแก้ไขไฟล์ Faculty.java ให้แสดงข้อความ Invoke Employee’s constructor เป็นข้อความที่ส่งให้กับ public Employee(String s)

ผลการรัน ที่ต้องการ

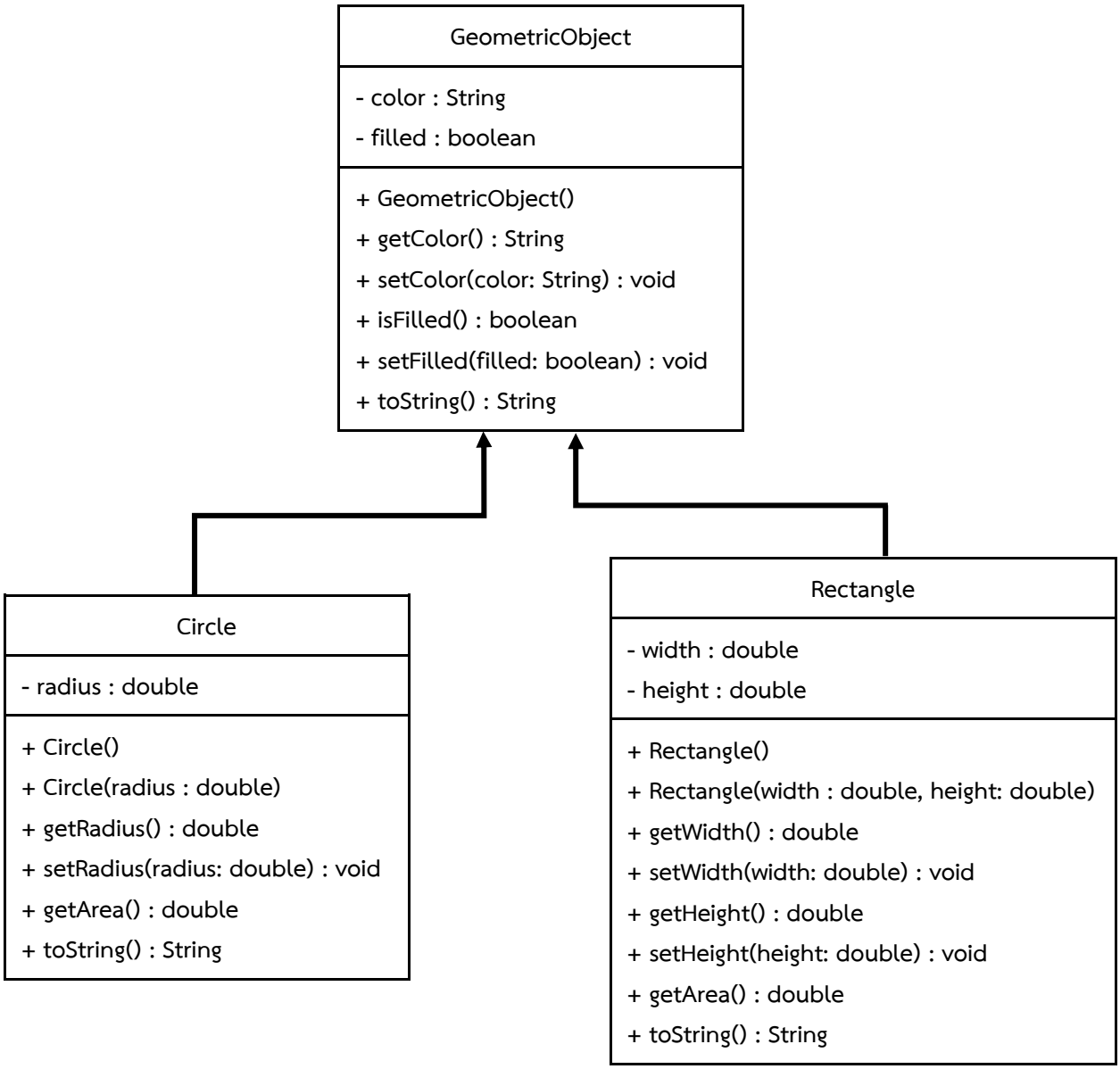
```
Invoke Employee’s constructor
Faculty's no-arg constructor is invoked
```

เขียนโคดที่แก้ไขที่ Faculty.java ให้ได้ผลลัพท์ตามที่ต้องการ

```
public class Faculty extends Employee
{
	public Faculty()
	{
		super("Invoke Employee's constructor");

		System.out.println("Faculty's no-arg constructor is invoked");
	}
}
```

Lab 5.2 ให้นิสิตสร้าง Class ดังนี้ 1. GeometricObject 2. Circle และ 3. Rectangle โดยแต่ละ Class มีส่วนประกอบตาม Class Diagram ข้างล่าง โดยมีรายละเอียดดังต่อไปนี้



GeometricObject มีรายละเอียดดังนี้

- มี Attribute คือ สี (color) , ระบาย (filled)
- Default Attribute color=white, filled=false
- Methods คือ getter และ setter
- toString() มีการทำงานดังนี้

```
public String toString() {
       return "Color: "+color + ",isFilled: "+ filled;
}
```

Circle สืบทอดมาจาก GeometricObject มีรายละเอียดดังนี้

- มี Attribute คือ รัศมี (radius)
- Default Attribute radius =0.0
- Methods คือ getter และ setter
- toString() มีการทำงานดังนี้

```
public String toString() {
	return "Color: "+getColor() + ",isFilled: "+ isFilled() +
",radius:"+radius;
}
```

- getArea() มีการทำงานดังนี้

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

Rectangle สืบทอดมาจาก GeometricObject มีรายละเอียดดังนี้

- มี Attribute คือ ความกว้าง (width), ความสูง (height)
- Default Attribute width =0.0 , height = 0.0
- Methods คือ getter และ setter
- toString() มีการทำงานดังนี้

```
public String toString() {
	return "Color: "+getColor() + ",isFilled: "+ isFilled()
	+ ",width: "+ width + ",height: " + height;
}
```

- getArea() มีการทำงานดังนี้

```
public double getArea(){
	return width*height;
}
```

Lab 5.2 ให้นิสิตสร้าง Class ชื่อ Lab52GeometricTest และให้เขียนโปรแกรมให้มีผลลัพท์ออกมาดังนี้

ตัวอย่างผลลัพท์

```
Color: white,isFilled: false,radius:5.0
Color: white,isFilled: false,width: 2.0,height: 4.0
```

เขียนโคด Lab52GeometricTest.java ให้ได้ผลลัพท์ตามที่ต้องการ

```
public class Lab52GeometricTest
{
	public static void main(String[] args)
	{
		_____________________________________ // สร้าง Object

		_____________________________________ // toString Method

		_____________________________________ // สร้าง Object

		_____________________________________ // toString Method
	}
}
```

ให้นิสิตสังเกตุ Method toString() ของทั้ง 3 Class จะเห็นได้ว่า มี String บางส่วนซ้ำกัน ดังนั้นควรเขียนโคดเพิ่มเฉพาะส่วนที่แตกต่างกันเท่านั้น ให้นิสิตทำการแก้ไข Method toString() ของ Circle และ Rectangle ต้องใช้คำสั่ง “super.” เข้าช่วยเท่านั้น โดยที่ผลการรัน Class Lab52GeometricTest.java ยังคงเหมือนเดิม

เขียนโคดที่แก้ไขที่ Method toString() ของ Class Circle

```
public String toString()
{



}
```

เขียนโคดที่แก้ไขที่ Method toString() ของ Class Rectangle

```
public String toString()
{



}
```

1.2 Try Catch and Finally

Lab 5.3 ให้นิสิตสร้าง Class ชื่อ Lab53TryCatchFinally.java นำไปใส่ใน Package ตามที่ได้สร้างไว้

Lab53TryCatchFinally.java

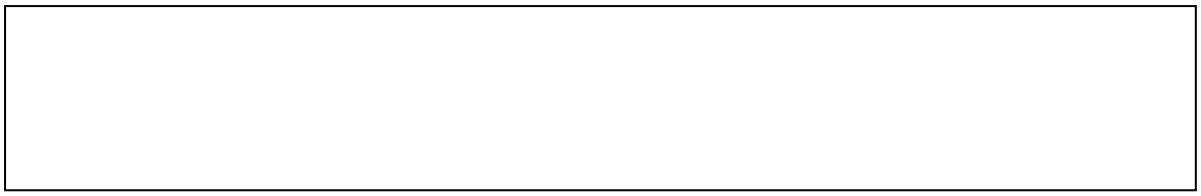
```
public class Lab53TryCatchFinally
{
	public static void main(String[] args)
	{
		int ans = MathDevider2.devider(10, 0);
		System.out.println(ans);
	}
}

class MathDevide2
{
	public static int devider(int num1, int num2)
	{
		int result = 0;
		try
		{
			result = (int)(num1/num2);
			System.out.println("Print from try");
		}
		catch(ArithmeticException e)
		{
			System.out.println(e);
			System.out.println("Print from catch");
		}
		finally
		{
			System.out.println("Print from finally");
		}
		return result;
	}
}
```

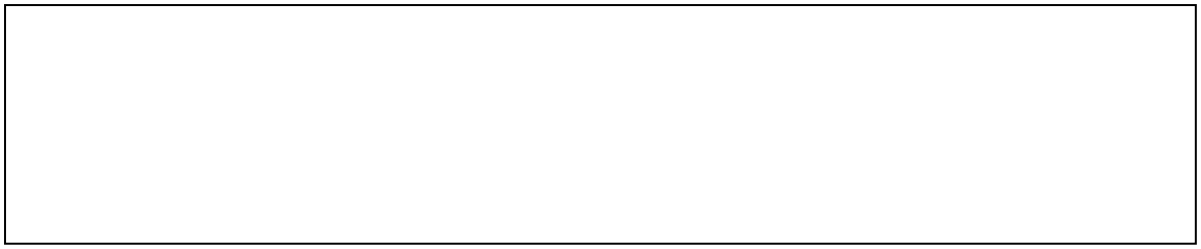
ให้นิสิตเขียนผลลัพท์ที่ได้ ลงด้านล่าง

ให้นิสิตทดลองเอา Block finally ออก รันได้ไหม ผลลัพท์เป็นเช่นไร

ให้นิสิตทดลองเอา Block catch ออก รันได้ไหม ผลลัพท์เป็นเช่นไร



ให้นิสิตเขียนให้กลับเหมือนเดิม และทดลองเปลี่ยนบรรทัดนี้เป็น int ans = MathDevider.devider(10, 5); จงเขียนผลลัพท์



1.3 วิธีการ overriding และ final

ให้นิสิตสร้าง package โดยให้ตั้งชื่อ package ว่า “p54” และให้นิสิตสร้าง class ดังต่อไปนี้ 1. A.java 2. B.java

A.java เขียนโคดดังนี้

```
package p54;

public class A
{
	public static void m1()
	{
		System.out.println("m1 in A");
	}

	public void m2()
	{
		System.out.println("m2 in A");
	}

	public final void m3()
	{
		System.out.println("m3 in A");
	}
}
```

B.java เขียนโคดดังนี้

```
package p54;

public class B extends A
{
	-----------(A)-----------
	{
		-----------(B)-----------
	}
}
```

ให้ใส่ code ต่อไปนี้ลง Class B ทีละ Method แล้วบันทึกผลการทดลองที่ได้ หลังจากนั้นให้ comment Method ที่ทดลองก่อนใส่ Method ถัดไป

Method	A	B
m1	public void m1()	System.out.println("m1 in B");
m2	void m2()	System.out.println("m2 in B");
m3	public void m3()	System.out.println("m3 in B");

บันทึกผลการทดลองที่ได้ลงในตารางข้างล่างนี้

กรณีใส่ Method	ผลการ compile และให้ระบุเหตุผลของการ error
m1	
m2	
m3	

วิธีการ Hiding fields, static method

เราจะนำ Class จาก package p54 ข้างบนมาแก้ไข ต่อดังนี้

A.java เขียนโคดดังนี้

```
package p54;

public class A
{
	public int i = 1;
	public static int j = 11;

	public static String m1()
	{
		return "A's static m1";
	}

	public String m2()
	{
		return "A's instance m2";
	}

	public String m3()
	{
		return "A's instance m3";
	}
}
```

B.java เขียนโคดดังนี้

```
package p54;

public class B extends A
{
	public int i = 2;
	public static int j = 12;

	public static String m1()
	{
		return "B's static m1";
	}

	public String m2()
	{
		return "B's instance m2";
	}
}
```

Lab 5.4 ให้นิสิตสร้าง Class ชื่อว่า Lab54HidingFields มีโค้ดดังต่อไปนี้

```
package p54;

public class Lab54HidingFields
{
	public static void main(String[] args)
	{
		A x = new B();

		// Access instance data field i
		System.out.println("(1)    x.i is " + x.i);          // (A)
		System.out.println("(2) (B)x.i is " + ((B)x).i);     // (B)

		// Access static data field j
		System.out.println("(3)    x.j is " + x.j);          // (C)
		System.out.println("(4) (B)x.j is " + ((B)x).j);     // (D)

		// Invoke static method m1
		System.out.println("(5)    x.m1() is " + x.m1());    // (E)
		System.out.println("(6) (B)x.m1() is " + ((B)x).m1());      //(F)

		// Invoke instance method m2
		System.out.println("(7)    x.m2() is " + x.m2());    // (G)
		System.out.println("(8)    x.m3() is " + x.m3());    // (H)
	}
}
```

ก่อนจะทำการรัน นิสิตลองตรวจสอบดูก่อนว่าถ้ารันคลาส Lab53HidingFields แล้วจะได้ผลลัพธ์เป็นอะไร ทำการรัน และบันทึกผลการทดลอง

บรรทัด	ผลที่คิดว่าจะเกิดขึ้นก่อนทำการ Run	ผลลัพท์ที่ได้จากการ Run
A		
B		
C		
D		
E		
F		
G		
H		

1.4 Static block , Initialization block

ให้นิสิตสร้าง package โดยให้ตั้งชื่อ package ว่า “p54” และให้นิสิตสร้าง class ดังต่อไปนี้ 1. N.java 2. M.java

N.java เขียนโคดดังนี้

```
package p55;

class N
{
	N()
	{
		System.out.println("N's constrctor body");
	}

	{
		System.out.println("N's instance initialization block");
	}

	static
	{
		System.out.println("N's static initialization block");
	}
}
```

M.java เขียนโคดดังนี้

```
package p55;

public class M extends N
{
	M()
	{
		System.out.println("M's constrctor body");
	}

	{
		System.out.println("M's instance initialization block");
	}

	static
	{
		System.out.println("M's static initialization block1");
	}

	static
	{
		System.out.println("M's static initialization block2");
	}
}
```

Lab 5.5 ให้นิสิตสร้าง Class ชื่อว่า Lab54Block มีโค้ดดังต่อไปนี้

```
package p55;

public class Lab55Block
{
	public static void main(String[] args)
	{
		M obj = new M();
	}

	static
	{
		System.out.println("Lab56Block's static block");
	}
}
```

ก่อนจะทำการรัน นิสิตลองตรวจสอบดูก่อนว่าถ้ารันคลาส Lab54Block แล้วจะได้ผลลัพธ์เป็นอะไร

ผลลัพท์ที่คิดก่อนจะทำการ Run

ผลลัพท์เมื่อทำการรัน

นิสิตทดลองหาเหตุผล ที่ทำให้เกิดผลลัพท์เช่นนี้

Homework#5

ในการบ้านนี้จะสร้างคลาส BankAccount (ใช้สำหรับเก็บข้อมูลบัญชีธนาคาร) โดยให้ตัว BankAccount จะมีไปปรับปรุงสร้างเป็นคลาสลูกได้หลายประเภท ดังนั้นในงานนี้จะให้ทดลองสร้างคลาส CheckingAccount ที่สืบทอดจากคลาสแม่ BankAccount ดังนั้นจะสร้างคลาสทั้งหมด 3 คลาสดังนี้

1. BankAccount (ใช้สำหรับเก็บข้อมูลบัญชีธนาคาร)

ประกอบด้วยคุณสมบัติดังต่อไปนี้

1. customerName เป็น String แทนชื่อของผู้ถือบัญชี
2. accountNumber เป็น String แทนหมายเลขบัญชี
3. balance เป็น double แทนยอดเงินที่เหลือในบัญชี

และมีเมทอดดังต่อไปนี้

4. BankAccount (String customerName, String accountNumber, double balance)
5. public String getCustomerName()
6. public void setCustomerName(String name)
7. public String getAccountNumber()
8. public void setAccountNumber(String number)
9. public double getBalance()
10. public void setBalance(double b)
11. public void withdraw(double amount)
    Method นี้จะต้องจรวจสอบยอดเงินที่จะถอนออกมา โดยห้ามให้ถอนเกินกว่าที่บัญชีที่ยอดเงินเหลืออยู่
12. public void deposit(double amount)
    Method นี้ยอดเงินที่ฝากเข้ามาจต้องมีค่ามากกว่า 0 ไม่สามารถฝากยอดติดลบได้

2. CheckingAccount (บัญชีกระแสรายวัน) ให้สืบทอดจากคลาส 1. BankAccount

ประกอบด้วยคุณสมบัติดังต่อไปนี้

1. overdraftFee เป็น double แทนค่าธรรมเนียมการโอน

และมีเมทอดดังต่อไปนี้

2. public CheckingAccount(String customerName, String accountNumber, double balance, double overdraftFee)
3. public double getOverdraftFee()
4. public void setOverdraftFee(double fee)
5. public void withdraw(double amount)
   Method นี้จะต้องนำค่าธรรมเนียมมาประกอบการคำนวณ ให้ยอดโอน + ค่าธรรมเนียม ต้องมีเพียงพอที่สามารถถอนเงินออกจากบัญชีได้ ถ้ามีไม่เพียงพอจะต้องมีการแจ้งเตือน

3. AccountTest (ตัวทดสอบโปรแกรม)

ตัวทดสอบจะต้องสร้าง BankAccount ขึ้นมาอย่างน้อย 2 BankAccount และทดลองสร้าง CheckingAccount ขึ้นมาอย่างน้อย 2 CheckingAccount โดยสามารถกำหนดชื่อบัญชี หมายเลขบัญชี และยอดเงินในบัญชี โปรแกรมสามารถทดสอบการฝากเงิน และถอนเงินได้ โดยข้อกำหนดในการฝากเงิน และถอนเงินให้ขึ้นกับแต่ละบัญชี (กรณี CheckingAccount ต้องมีการกำหนดค่าธรรมเนียมการถอนเงินได้ในโปรแกรม)

ตัวอย่างผลลัพท์ เมื่อทำการรันโปรแกรม

```
#1 BankAccount1
NAMR: Bob
ADDR: 55879
BALANCE: 101.5

#2 BankAccount2
NAMR: Mary
ADDR: 48537
BALANCE: 100824.5


#3 CheckingAccount1
NAMR: Jane
ADDR: 25837
BALANCE: 1253
FEE: 15


#4 CheckingAccount2
NAMR: Max
ADDR: 52428
BALANCE: 12.251523
FEE: 0.0007


Choose Account: 52428
Choose Action (1. Withdraw 2. Deposit): 1
Value: 1.123
Withdraw Complete!!
52428 Max Withdraw 1.123 Fee 0.0007 Balance 11.127823
########

Choose Account: 55879
Choose Action (1. Withdraw 2. Deposit): 1
Value: 1100
Withdraw Fail!!
########

Choose Account: 52428
Choose Action (1. Withdraw 2. Deposit): 2
Value: 0.01
Deposit Complete!!
52428 Max Deposit 0.01 Balance 11.137823
########
```