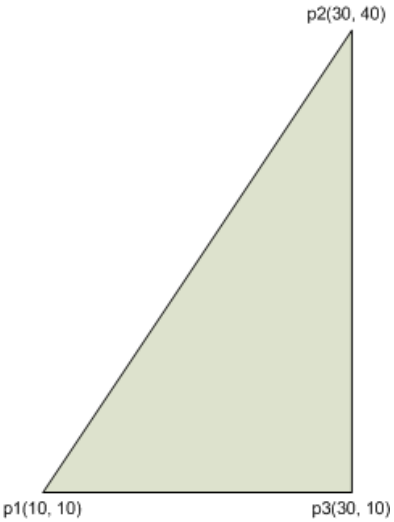
ปฏิบัติการที่ 12 Final

1. คลาส Point สำหรับใช้เก็บข้อมูลตำแหน่ง (x, y)



- วิธีการคำนวณระยะทางระหว่างจุดสองจุด

```
distance = Math.sqrt( (x2-x1)*(x2-x1) + (y2-y1)*(y2-y1) )
```

Variable

- x p3, x = 30 (int)
- y p3, y = 10 (int)

Constructor

- Point() x = 0, y = 0 ใช้ this()
- Point(int x, int y) x = x, y = y

Method

- getX() อ่าน x
- setX(int x) เปลี่ยน x
- getY() อ่าน y
- setY(int y) เปลี่ยน y
- distance(int x, int y) ระยะทางจากตัวเอง ไปยัง จุด x, y ที่ส่งมาให้
- distance(Point other) ระยะทางจากตัวเอง ไปยัง จุด x, y ของ other ที่ส่งมาให้
- toString() สถานะของข้อมูล

Program

- โปรแกรม Point.java

```
public class Point {

	// Your code
}
```

- โปรแกรม PointTest.java ทดสอบการใช้งาน Constructor และ Method ทั้งหมด

```
public class PointTest {
	public static void main(String[] args) {
		// Point
		System.out.println("Point");
		System.out.println("--------------------");
		Point p;

		// Your code
	}
}
```

- โปรแกรม PointApp.java คำนวณหาความยาวของเส้นรอบรูปสามเหลี่ยมด้านบน

```
public class PointApp {
	public static void main(String[] args) {

		// Your code
	}
}
```

```
Result: 86.06
```

2. ออกแบบคลาส RainFall สำหรับใช้เก็บข้อมูลปริมาณน้ำฝนแต่ละเดือน

RainFall

ข้อมูล

- rains อาร์เรย์สำหรับเก็บข้อมูลปริมาณน้ำฝน (int) จำนวน 12 เดือน (**ห้ามใช้ ArrayList**)

การทำงาน

- บันทึกข้อมูลในแต่ละเดือน เช่น บันทึก(1, 882) เดือน 1 ปริมาณน้ำฝน 882
- เรียกดูข้อมูลในแต่ละเดือน เช่น เรียกดู(1) เดือน 1 ปริมาณน้ำฝน 882
- ปริมาณน้ำฝนรวมทั้งปี
- ปริมาณน้ำฝนเฉลี่ยแต่ละเดือน
- **เดือน**ที่มีฝนตกมากที่สุด
- **เดือน**ที่มีฝนตกน้อยที่สุด

โปรแกรม

- โปรแกรม RainFall.java

```
public class RainFall {
	int rains[];

	// Your Code

	}
}
```

- ไม่ต้องเขียนโปรแกรมสำหรับทดสอบ RainFall

- โปรแกรม RainFallApp.java
  - โปรแกรมซึ่งสามารถแสดงผลลัพธ์ตามที่กำหนดให้
  - แนะนำให้แสดงผลลัพธ์ด้วย **\t**

```
public class RainFallApp {
	public static void main(String[] args) {
		RainFall rf = new RainFall();
		rf.setRain(1, 882);
		rf.setRain(2, 275);
		rf.setRain(3, 164);
		rf.setRain(4, 420);
		rf.setRain(5, 666);
		rf.setRain(6, 400);
		rf.setRain(7, 285);
		rf.setRain(8, 282);
		rf.setRain(9, 908);
		rf.setRain(10, 375);
		rf.setRain(11, 684);
		rf.setRain(12, 354);

		// Your Code
	}

	public static String monthThai(int month) {
		String month_thai[] = { "", "มกราคม", "กุมภาพันธ์", "มีนาคม", "เมษายน", "พฤษภาคม",
"มิถุนายน", "กรกฎาคม", "สิงหาคม", "กันยายน", "ตุลาคม", "พฤศจิกายน", "ธันวาคม" };
		return month_thai[month];
	}
}
```

```
รายงานปริมาณน้ำฝน
----------------
มกราคม		882
กุมภาพันธ์		275
มีนาคม		164
เมษายน		420
พฤษภาคม		666
มิถุนายน		400
กรกฎาคม		285
สิงหาคม		282
กันยายน		908
ตุลาคม		375
พฤศจิกายน		684
ธันวาคม		354
----------------
ปริมาณน้ำฝนรวมทั้งปี		5695
ปริมาณน้ำฝนเฉลี่ยแต่ละเดือน	474
เดือนที่มีฝนตกมากที่สุด		กันยายน, 908
เดือนที่มีฝนตกน้อยที่สุด		มีนาคม, 164
```

3. ออกแบบคลาส BankAccount สำหรับใช้เก็บข้อมูลบัญชีธนาคาร

ข้อมูล

- ชื่อเจ้าของบัญชี
- ยอดเงินคงเหลือ
- จำนวนครั้งที่ฝากเงินสำเร็จ
- จำนวนครั้งที่ถอนเงินสำเร็จ

การทำงาน

- ฝากเงิน deposit(amount)
  - ตรวจสอบ amount > 100
  - เพิ่มยอดเงินคงเหลือ
  - เพิ่มจำนวนครั้งที่ฝากเงินสำเร็จ
- ถอนเงิน withdraw(amount)
  - ตรวจสอบ amount > 100 และ amount < ยอดเงินคงเหลือ
  - ลดยอดเงินคงเหลือ
  - เพิ่มจำนวนครั้งที่ถอนเงินสำเร็จ

โปรแกรม

- โปรแกรม BankAccount.java

```
public class BankAccount {

	// Your code
}
```

- โปรแกรม BankAccountTest.java (**ห้ามแก้ไข**)

```
public class BankAccountTest {
	public static void main(String[] args) {
		// BankAccount
		System.out.println("BankAccount");
		System.out.println("------------------------------");
		BankAccount ba = new BankAccount("สมชาย", 3000);
		report(ba);

		System.out.println("Deposit");
		System.out.println("--------------------");
		if (ba.deposit(2000)) {
			System.out.println("ฝากเงิน 2000 " + "ยอดเงินคงเหลือ " + ba.getBalance());
		}
		if (ba.deposit(1000)) {
			System.out.println("ฝากเงิน 1000 " + "ยอดเงินคงเหลือ " + ba.getBalance());
		}
		if (!ba.deposit(20)) {
			System.out.println("ฝากเงิน 20 ฝากเงินไม่ได้-ฝากเงินน้อยกว่า 100");
		}
```

```
		System.out.println("Withdraw");
		System.out.println("--------------------");
		if (ba.withdraw(2000)) {
			System.out.println("ถอนเงิน 2000 " + "ยอดเงินคงเหลือ " + ba.getBalance());
		}
		if (!ba.withdraw(5000)) {
			System.out.println("ถอนเงิน 5000 " + "ถอนเงินไม่ได้-ยอดเงินคงเหลือไม่พอ");
		}
		if (!ba.withdraw(20)) {
			System.out.println("ถอนเงิน 20 ถอนเงินไม่ได้-ถอนเงินน้อยกว่า 100");
		}
		report(ba);
	}

	public static void report(BankAccount acc) {
		System.out.println("บัญชี: " + acc.getName() + " ยอดเงินคงเหลือ: " +
acc.getBalance()
				+ " จำนวนครั้งที่ฝากเงิน: " + acc.getDeposit() + " จำนวนครั้งที่ถอนเงิน: " +
acc.getWithdraw());
	}
}
```

```
BankAccount
---------------------------
บัญชี: สมชาย ยอดเงินคงเหลือ: 3000 จำนวนครั้งที่ฝากเงิน: 0 จำนวนครั้งที่ถอนเงิน: 0
Deposit
--------------------
ฝากเงิน 2000 ยอดเงินคงเหลือ 5000
ฝากเงิน 1000 ยอดเงินคงเหลือ 6000
ฝากเงิน 20 ฝากเงินไม่ได้-ฝากเงินน้อยกว่า 100
Withdraw
--------------------
ถอนเงิน 2000 ยอดเงินคงเหลือ 4000
ถอนเงิน 5000 ถอนเงินไม่ได้-ยอดเงินคงเหลือไม่พอ
ถอนเงิน 20 ถอนเงินไม่ได้-ถอนเงินน้อยกว่า 100
บัญชี: สมชาย ยอดเงินคงเหลือ: 4000 จำนวนครั้งที่ฝากเงิน: 2 จำนวนครั้งที่ถอนเงิน: 1
```

4. ออกแบบคลาส SavingsAccount สำหรับใช้เก็บข้อมูลบัญชีเงินฝากออมทรัพย์โดยสืบทอดมาจากคลาส BankAccount

ข้อมูล

- สืบทอดมาจากคลาส BankAccount

การทำงาน

- ฝากเงิน deposit(amount)
  - **ตรวจสอบ amount > 1000**
  - เพิ่มยอดเงินคงเหลือ
  - เพิ่มจำนวนครั้งที่ฝากเงินสำเร็จ
- ถอนเงิน withdraw(amount)
  - **ตรวจสอบ amount > 1000 และ หลังจากถอนเงินแล้วยอดเงินคงเหลือต้อง > 1000**
  - ลดยอดเงินคงเหลือ
  - เพิ่มจำนวนครั้งที่ถอนเงินสำเร็จ

โปรแกรม

- โปรแกรม SavingsAccount.java

```
public class SavingsAccount {

	// Your code
}
```

- โปรแกรม SavingsAccountTest.java (**ห้ามแก้ไข**)

```
public class SavingsAccountTest {
	public static void main(String[] args) {
		// SavingsAccount
		System.out.println("SavingsAccount");
		System.out.println("------------------------------");
		SavingsAccount sa = new SavingsAccount("รักดี", 3000);
		report(sa);

		System.out.println("Deposit");
		System.out.println("--------------------");
		if (sa.deposit(2000)) {
			System.out.println("ฝากเงิน 2000 " + "ยอดเงินคงเหลือ " + sa.getBalance());
		}
		if (!sa.deposit(200)) {
			System.out.println("ฝากเงิน 200 ฝากเงินไม่ได้-ฝากเงินน้อยกว่า 1000");
		}

		System.out.println("Withdraw");
		System.out.println("--------------------");
		if (sa.withdraw(2000)) {
			System.out.println("ถอนเงิน 2000 " + "ยอดเงินคงเหลือ " + sa.getBalance());
		}
		if (sa.withdraw(1500)) {
```

```
                System.out.println("ถอนเงิน 1500 " + "ยอดเงินคงเหลือ " + sa.getBalance());
        }
        if (!sa.withdraw(2000)) {
                System.out.println("ถอนเงิน 2000 " + "ถอนเงินไม่ได้-หลังจากถอนเงินทำให้ยอดเงินคงเหลือไม่ถึง
1000");
        }
        if (!sa.withdraw(200)) {
                System.out.println("ถอนเงิน 200 ถอนเงินไม่ได้-ถอนเงินน้อยกว่า 1000");
        }
        report(sa);
    }

    public static void report(BankAccount acc) {
        System.out.println("บัญชี: " + acc.getName() + " ยอดเงินคงเหลือ: " +
acc.getBalance()
                        + " จำนวนครั้งที่ฝากเงิน: " + acc.getDeposit() + " จำนวนครั้งที่ถอนเงิน: " +
acc.getWithdraw());
    }
}
```

```
SavingsAccount
---------------------------
บัญชี: รักดี ยอดเงินคงเหลือ: 3000 จำนวนครั้งที่ฝากเงิน: 0 จำนวนครั้งที่ถอนเงิน: 0
Deposit
-------------------
ฝากเงิน 2000 ยอดเงินคงเหลือ 5000
ฝากเงิน 200 ฝากเงินไม่ได้-ฝากเงินน้อยกว่า 1000
Withdraw
-------------------
ถอนเงิน 2000 ยอดเงินคงเหลือ 3000
ถอนเงิน 1500 ยอดเงินคงเหลือ 1500
ถอนเงิน 2000 ถอนเงินไม่ได้-หลังจากถอนเงินทำให้ยอดเงินคงเหลือไม่ถึง 1000
ถอนเงิน 200 ถอนเงินไม่ได้-ถอนเงินน้อยกว่า 1000
บัญชี: รักดี ยอดเงินคงเหลือ: 1500 จำนวนครั้งที่ฝากเงิน: 1 จำนวนครั้งที่ถอนเงิน: 2
```

วิธีการส่งงาน

- สร้างเอกสารด้วย Microsoft Word โดยตั้งชื่อ S1234567890-Lab12.doc

รหัส: 1234567890 **ชื่อ-สกุล: นาย สมชาย รักดี**

- ให้แยกโปรแกรมละ 1 หน้า

`PointApp.java`

`PointTest.java`

`Point.java`

`RainFallApp.java`

`RainFall.java`

`BankAccount.java`

`SavingsAccount.java`

- บันทึกเอกสารเป็น PDF (ไม่ตรวจงานในรูปแบบอื่น)
- Upload ไปที่ LMS2

ส่งงานเฉพาะที่ทำเสร็จ

คะแนน 0

สำหรับงานต้นฉบับและงานที่คัดลอก

ปฏิบัติการเป็นส่วนหนึ่งของ"การสอบปลายภาค"

คะแนนปฏิบัติการ

- นักศึกษาจะต้องเขียนโปรแกรมด้วยตนเอง
- ส่งงานเฉพาะที่ทำเสร็จภายในเวลาที่กำหนด
- ให้คะแนนตามความสมบูรณ์ถูกต้องของงาน

ตรวจปฏิบัติการ

- ไม่ตรวจงานที่ส่งจาก IP หมายเลขเดียวกัน
- ไม่ตรวจงานที่คัดลอกและงานต้นฉบับ
- ไม่ตรวจงานที่ส่งผิดปฏิบัติการ
- ไม่ขยายเวลาส่งงาน
- ไม่รับงานย้อนหลัง

การวัดและประเมินผล

กลางภาค (60 %)

คะแนนเก็บ	20 %	(เข้า Lab, ส่ง Lab, เข้าเรียน)
คะแนนสอบกลางภาค	40 %	

ปลายภาค (40 %)

ปฏิบัติการครั้งที่ 1	17 มีนาคม 2563 (1000-1200, 1300-1500)
ปฏิบัติการครั้งที่ 2	24 มีนาคม 2563 (1000-1200, 1300-1500)
ปฏิบัติการครั้งที่ 3	31 มีนาคม 2563 (1300-1600)
แบบฝึกหัดที่ 1	03 เมษายน 2563