ข้อสอบเป็นแบบ Close Book ไม่อนุญาตให้นำเอกสารใด ๆ เข้าห้องสอบ
ให้นักศึกษาเลือกทำปัญหาต่อไปนี้ซึ่งคะแนนสูงสุดที่ได้จะเป็น 10 คะแนน
นักศึกษาสามารถทำได้หลายข้อเพื่อให้ครบ 10 คะแนน
เมื่อทำเสร็จแล้วสามารถเรียกกรรมการในห้องตรวจให้คะแนนได้ตลอด
** กรรมการในห้องจะไม่มีการตรวจให้คะแนนในข้อย่อย
** กรรมการในห้องจะตรวจทั้งโปรแกรม หากทำงานได้อย่างสมบูรณ์จึงจะได้คะแนนเต็มในข้อนั้น

ในกรณีที่ส่งในห้องแล้วได้คะแนนยังไม่ครบ 10 คะแนน สามารถเลือกข้อที่ยังไม่ได้คะแนนจำนวน 1 ข้อเพื่อตรวจ Code ได้
** การตรวจ Code จะตรวจให้คะแนนในข้อย่อยด้วย
** การตรวจ Code จะรู้คะแนนในภายหลังและจะนำคะแนนที่ได้มารวมเป็นคะแนน lab

1. [3 คะแนน] เขียนและทดลองใช้ `class Rectangle` ที่เก็บค่าความกว้างกับความสูงของสี่เหลี่ยมมุมฉาก และมีเมธอดคำนวณและคืนค่าเส้นรอบรูป (perimeter) โดยคิดจาก 2x(ความกว้าง+ความสูง) และพื้นที่ (area) โดยคิดจาก ความกว้าง+ความสูง
    a. [1คะแนน] `public Rectangle(double width, double height)` เป็น constructor เพื่อรับค่าความกว้างกับความสูงมาเก็บไว้
    b. [1 คะแนน] `public double getArea()` คำนวณและคืนค่าพื้นที่ (area) โดยคิดจาก ความกว้าง+ความสูง
    c. [1 คะแนน] `public double getPerimeter()` คำนวณและคืนค่าเส้นรอบรูป (perimeter) โดยคิดจาก 2x(ความกว้าง+ความสูง)

ตัวอย่างข้อมูลทดสอบ

ความกว้าง	ความสูง	พื้นที่	เส้นรอบรูป
210	3,645	765,450	7,710
104	140	14,560	488
94	150	14,100	488

2. [3 คะแนน] จงเขียนเมธอด `static boolean isPrime(int num)` ซึ่งเขียนใน driver class เดียวกับที่มีเมธอด `main` ได้เลย เมธอด `isPrime` รับจำนวนเต็มแล้วตรวจสอบว่าตัวเลขนั้นเป็นจำนวนเฉพาะหรือไม่ โดยที่จำนวนเฉพาะคือจำนวนที่ไม่มีเลขไหนหารลงตัวเลยยกเว้นเลข **1** กับตัวมันเอง เมธอด `isPrime` คืนค่าเป็น `true` หากจำนวนตัวเลขนั้นเป็นจำนวนเฉพาะ และคืนค่าเป็น `false` หากไม่ใช่จำนวนเฉพาะ

ตัวอย่างข้อมูลทดสอบที่**เป็น**จำนวนเฉพาะ
1, 2, 3, 5, 7
211, 307, 311, 277, 149
ตัวอย่างข้อมูลทดสอบที่**ไม่เป็น**จำนวนเฉพาะ
4, 6, 8, 9, 10
340, 203, 314, 214, 399

ตัวอย่างส่วนของโปรแกรมในเมธอด main

```
int num=11;
System.out.println(num + " is " + (isPrime(num) ? "": "not ") + "a prime
number ");
```

ตัวอย่าง output

```
11 is a prime number
```

3. [4 คะแนน] ให้นักศึกษาสร้างเกมหยิบไม้ขีดที่มีผู้เล่น 2 คนผลัดกันหยิบไม้ขีดออกจากกองโดยผู้เล่นแต่ละคนหยิบไม้ขีดได้ครั้งละ 1-3 ก้านเท่านั้น และผู้เล่นที่หยิบไม้ขีดครั้งสุดท้ายจะเป็นผู้ชนะ

1.1 ให้นักศึกษาสร้างคลาสชื่อ Matchstick ดังแสดงใน UML Class Diagram

Matchstick
- matchsticks: int
+ Matchstick(initMatches: int) + takeMatches(numMatches: int): void + isValidMove(numMatches: int): boolean + isGameOver(): boolean

ใน UML Class Diagram มีรายละเอียดของ Attribute และ Methods ดังนี้

Attribute:

- `private int matchsticks` ใช้สำหรับเก็บจำนวนก้านไม้ขีดที่เหลือ

Methods:

- [1 คะแนน] `public Matchstick (int initMatches)` เป็น constructor สำหรับกำหนดจำนวนก้านไม้ขีดเริ่มต้น
- [1 คะแนน] `public void takeMatches (int numMatches)` เป็น method ที่ใช้สำหรับหยิบก้านไม้ขีดออกตามจำนวนที่ระบุ
- [1คะแนน] `isValidMove(int numMatches)` เป็น method ที่ใช้สำหรับตรวจสอบว่าจำนวนของก้านไม้ขีดที่หยิบออกถูกต้องหรือไม่ คือหยิบ 1-3 ก้านเท่านั้นและต้องไม่เกินจำนวนก้านไม้ขีดที่เหลืออยู่
- [1 คะแนน] `public boolean isGameOver()` เป็น method ที่ใช้สำหรับตรวจสอบว่าเกมสิ้นสุดแล้วหรือยัง ซึ่งเกมส์จะจบเมื่อไม่เหลือก้านไม้ขีดอยู่เลย MatchGam

1.1 จงสร้างคลาสชื่อ MatchGame เพื่อใช้สำหรับเล่นเกม ดังนี้

```
import java.util.Scanner;
public class MatchGame {
    public static void main (String[] args) {
        int numMatches=0;
        int player = 0;
        Matchstick game = new Matchstick(20);
        Scanner input = new Scanner(System.in);
        do {
            System.out.print("Player " + player + " >> ");
            numMatches = input.nextInt();
            if (game.isValidMove(numMatches)){
                game.takeMatches(numMatches);
                player = (player + 1) % 2;
            }
        } while (!game.isGameOver());
        System.out.println("Player " + ((player + 1) % 2) + " wins");
    }
}
```

ตัวอย่าง output

```
Player 0 >> 0
Player 0 >> 1
Player 1 >> 1
Player 0 >> 2
Player 1 >> 2
Player 0 >> 3
Player 1 >> 3
Player 0 >> 4
Player 0 >> 3
Player 1 >> 0
Player 1 >> 4
Player 1 >> 3
Player 0 >> 2
Player 0 wins
```

4. [8 คะแนน] การแสดงข้อมูลที่ขึ้นกับเวลาอย่างเช่น ความเร็วรถ อุณหภูมิของเครื่องจักร ปริมาณฝุ่น ณ เวลาใด ๆ สามารถที่จะคำนวณและเก็บค่าทางสถิติบางอย่างจากข้อมูลที่เกิดขึ้นทั้งหมดโดยไม่จำเป็นต้องเก็บข้อมูลทั้งหมดไว้ได้ จงเขียน `class StreamingStat` เพื่อใช้ในการเก็บและประมวลผลค่าทางสถิติ ค่าสูงสุด ค่าต่ำสุด และค่าเฉลี่ย โดยมีความสามารถดังต่อไปนี้

   a. [1 คะแนน] `public StreamingStat(String name)` เป็น **constructor** มีหน้าที่กำหนดค่าต่าง ๆดังนี้
      - กำหนด `name` ตามที่ได้รับมา
      - กำหนดค่าเริ่มต้นของ ค่าสูงสุด(`max`) ค่าเฉลี่ย(`mean`) ข้อมูลสุดท้ายที่ได้รับ `(lastValue)` และจำนวนข้อมูล `(n)` เป็น 0
      - กำหนดค่าเริ่มต้นของค่าต่ำสุด(min) เท่ากับ `Double.POSITIVE_INFINITY`

   b. [2 คะแนน] `public void setValue(doble num)` จะบันทึกค่าปัจจุบัน เพิ่มตัวนับจำนวนข้อมูล และ **update** ค่าทางสถิติต่าง ๆ โดยเรียกใช้เมธอดอื่น ๆ ในข้อ c-e

   c. [1.5 คะแนน] `private void updateMax()` จะเปรียบเทียบค่าปัจจุบันกับค่า **max** ที่เก็บไว้ หากค่าปัจจุบันมากกว่าค่า **max** ที่เก็บไว้ก็จะนำค่าปัจจุบันไปเก็บไว้ใน **max** แทน

   d. [1.5 คะแนน] `private void updateMin()` จะเปรียบเทียบค่าปัจจุบันกับค่า **min** ที่เก็บไว้ หากค่าปัจจุบันน้อยกว่าค่า **min** ที่เก็บไว้ก็จะนำค่าปัจจุบันไปเก็บไว้ใน **min** แทน

   e. [1.5 คะแนน] `private void updateAverage()` จะคำนวณและเก็บค่าเฉลี่ยใหม่จากสมการ

$$\text{ค่าเฉลี่ยใหม่} = \frac{\left(\text{ค่าเฉลี่ยเดิม} \times (\text{จำนวนข้อมูลปัจจุบัน} - 1)\right) + \text{ข้อมูลที่เข้ามาใหม่}}{(\text{จำนวนข้อมูลปัจจุบัน})}$$

   f. [0.5 คะแนน] `public String toString()` สามารถคลิกขวา **insert code** และใช้ **toString** ตามที่ **NetBeans** สร้างให้ได้เลย

ตัวอย่างการ **code** ในเมธอด **main**

```
StreamingStat carSpeed = new StreamingStat("Speed");
carSpeed.setValue(20.0);
carSpeed.setValue(28.0);
carSpeed.setValue(40.0);
carSpeed.setValue(30.0);
System.out.println(carSpeed.toString());

StreamingStat machineTemp = new StreamingStat("Temp");
machineTemp.setValue(80.0);
machineTemp.setValue(85.0);
machineTemp.setValue(75.0);
System.out.println(machineTemp.toString());
```

ตัวอย่าง **output** (แสดงผลต่างจากนี้ได้แต่ขอให้ได้ข้อมูลครบถ้วน)

```
StreamingStat{name=Speed, mean=29.5, max=40.0, min=20.0, lastValue=30.0, n=4}
StreamingStat{name=Temp, mean=80.0, max=85.0, min=75.0, lastValue=75.0, n=3}
```

5. [10 คะแนน] บริษัทส่งสินค้าด่วนต้องการคำนวณค่าส่งสินค้าที่ขึ้นอยู่กับน้ำหนักและขนาดของกล่องจงเขียนโปรแกรมเพื่อรับน้ำหนักและขนาดของกล่องทั้ง ความกว้าง ความยาว ความสูง จากนั้นคำนวณราคาขนส่งตามเงื่อนไขดังต่อไปนี้ (จะสร้างเป็น Class แยกหรือไม่สร้างก็ได้แต่อย่างน้อยควรทำเป็น เมธอดแยกออกมาจากเมธอด main)
    a. [2 คะแนน] ราคาเมื่อเทียบกับน้ำหนักรับส่งน้ำหนักสูงสุด 20 กิโลกรัม หากเกิน จะคืนค่า -1

$$c(w) = \begin{cases} 85 & 0 < w \leq 5 \\ 100 & 5 < w \leq 10 \\ 140 & 10 < w \leq 15 \\ 185 & 15 < w \leq 20 \end{cases}$$

    b. [2 คะแนน] ราคาเมื่อเทียบกับขนาดของกล่องขนาดของพัสดุ โดยนำ $l =$ ความกว้าง + ความยาว + ความสูง รับความยาวรวมสูงสุด 150 เซนติเมตร หากเกิน จะคืนค่า -1

$$c(l) = \begin{cases} 85 & 0 < l \leq 60 \\ 100 & 60 < l \leq 90 \\ 140 & 90 < l \leq 120 \\ 185 & 120 < l \leq 150 \end{cases}$$

    c. [3 คะแนน] ราคาขนส่งจะคิดจาก ราคาที่มากที่สุดเมื่อพิจารณาจาก $c(w)$ และ $c(l)$ แล้ว แต่ถ้ามี $c(w)$หรือ $c(l)$ อันใดอันหนึ่ง เป็น -1 ค่าขนส่งจะคืนค่าเป็น -1 ด้วย
    d. [3 คะแนน] โปรแกรมเพื่อแสดงเมนูการคิดราคาเมื่อกด 0 ให้ออกจากโปรแกรมนอกนั้นให้รอรับค่าน้ำหนักและขนาดของกล่องทางแป้นพิมพ์แล้วแสดงข้อมูลต่าง ๆ ดังตัวอย่าง

ตัวอย่างการแสดงผล (แสดงผลต่างจากนี้ได้แต่ขอให้ได้ข้อมูลครบถ้วน)

```
weight: 8.0
width 1: 22.5
width 2: 19.0
height: 9.0
cost by weight : 100.0   cost by size: 85.0
cost : 100.0
```

ตัวอย่างข้อมูลทดสอบ

น้ำหนัก	ความกว้างด้านที่ 1	ความกว้างด้านที่ 2	ความสูง	ค่าขนส่งตามน้ำหนัก	ค่าขนส่งตามขนาด	ค่าขนส่ง
21.5	54	55	30	-1	185	**-1**
3	14.5	4.5	15	85	85	**85**
16	3.5	11.5	48	185	100	**185**
7	14.5	27.5	27	100	100	**100**
11.5	6.5	12.5	26.5	140	85	**140**
14	61.5	40.5	70.5	140	-1	**-1**