ID_____ชื่อ _____

1. (10 คะแนน) มีโค้ดเมธอดของภาษาที่มี short-circuit Boolean evaluation ดังนี้

```
int f1(int n, int m, int p) {// assume 1<=n<=100 , 1<=m<=1000, m<p
        int sum =0;
        for(int i=1; i<=n || (i>=m && i <=p); i++)
                sum++;
        return sum;
}
```

ถ้าภาษานี้ไม่มี short-circuit แต่เราต้องการให้การเช็คและรันเหมือนกับภาษาที่ใช้ short-circuit จงเขียนเมธอดนี้ใหม่

```
public static int f2(int n, int m, int p) {
        int sum = 0;
        int i = 1;
        for (;; i++) {
                if (i <= n) {
                        sum++;
                } else if (i >= m) {
                        if (i <= p) {
                                sum++;
                        } else {
                                break;
                        }
                } else {
                        break;
                }
        }
        return sum;
}
```

Initialize variables (1)

Has loop that increments I (1)

If true, sum++ and go to next iteration (no other checks) (2)

If true, check another condition (1)

If true, sum++ and go to next iteration (no other checks) (2)

If false, exit loop (1)

Else , exit loop (1)

Don't forget to return the result (1)

2. มีโค้ดของภาษาที่เมธอด nest กันได้ ดังนี้:

ID_____ชื่อ _____

```
public class XY{
    public static void main(String[] args){
            int x =1;
            int y =1;
            public void method01(int a){
                    public void method2(int y){
                            public void method3(int m){
                                    m += x+y+a;
                                    System.out.println(x + “,” +y + “,” +m);  //line1
                            }
                            int x = y+1;
                            int m = x+y;
                            method3(m);
                            System.out.println(x + “,” +y + “,” +m);  //line2
                    }
                    x = x+ a - y;
                    method2(x);
            }
            method1(x+1);
            System.out.println(x + “,” +y);  //line3
    }
}
```

ให้ถือว่าพารามิเตอร์ของเมธอด เป็นการ declare ตัวแปร

(8 คะแนน) ถ้าใช้ static scope, line1 ถึง line 3 จะพิมพ์อะไรออกมา (ตอบตัวเลขละ 1 คะแนน)

Line1 : 3,2,12

Line2: 3,2,5

Line 3 :2,1

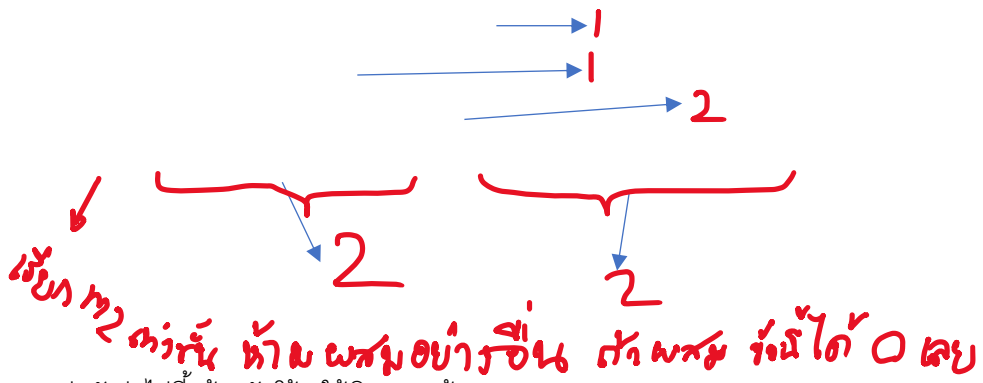
ให้คะแนนตัวเลขละ 1 คะแนน แต่ถ้าผิด ให้เอาค่าที่น้องได้จาก line 1 ไปคิดต่อ line2 และเอาค่าที่น้องได้จาก Line 2 ไปคิดต่อ line 3 ได้

3. (8 คะแนน) ภาษาที่จะใช้ต่อไปนี้ (syntax เหมือนจาว่า) สามารถ optimize การใช้งานเมธอดได้ถ้าเมธอดเขียนด้วย tail recursion จงเปลี่ยนโค้ดเมธอดที่ให้ต่อไปนี้ ให้ใช้ tail recursion

```
public static String m1(String s) { // i >=0
        if(s.isBlank()) return s;
        int n = s.length();
        if(n ==1) return s;
        return m1(s.substring(1)) + s.charAt(0);
}
```

```
public static String m2(String s, String acc) { // i >=0
    if(s.isBlank()) return acc;
    int n = s.length();
    if(n ==1) return s + acc;
    return m2(s.substring(1), s.charAt(0) +acc);
}
```

ID_____ชื่อ _____



4. มีโค้ดภาษาจาว่าดังต่อไปนี้ (ห้ามรันโค้ด ให้พิจารณาด้วยตา)

```
class Robot {..}
class RealRobot extends Robot {..}
class FantasyRobot extends Robot {..}
…
Robot a1, a2;
RealRobot r1, r2;
FantasyRobot f1, f2;
Robot[] rs = new Robot[3];

a1 = new Robot();
a2 = new FantasyRobot();
f2 = (FantasyRobot)a1;    //1

rs[0] = a1;
rs[1] = a2;
f1 = rs[1];     //2
```

- บรรทัดเบอร์ 1 คอมไพล์ไม่ได้หรือรันไม่ได้ ถ้าไม่ได้ เนื่องจากสาเหตุอะไร (2 คะแนน)

คอมไพล์ผ่าน แต่รันไม่ผ่าน a1 new มาจาก Robot จะcast เป็น FantasyaRobot ซึ่งไม่ได้เป็น superClass-subclass กันตอนรันไม่ได้ ถือว่าไทป์ ไม่ compatible ->ClassCastException

สมมุติว่าบรรทัด เบอร์ 1 โดนคอมเม้นทิ้งไป บรรทัดเบอร์ 2 คอมไพล์ไม่ได้หรือรันไม่ได้ ถ้าไม่ได้ เนื่องจากสาเหตุอะไร (3 คะแนน)

คอมไพล์ไม่ผ่าน เพราะว่า rs[1] ถือเป็น type Robot จะเอาไปเก็บในตัวแปร f1 ที่เป็น FantasyRobot (ซึ่งมีความละเอียดกว่า) ไม่ได้ เพราะถ้าทำได้ f1 จะมีเมธอดและตัวแปรให้เรียกไม่ครบ ดังนั้นถือว่า type ไม่ compatible แม้ว่า ตอนรันที่ rs[1] เก็บค่าจะเป็น FantasyRobot ก็ตาม แต่คอมไพล์เลอร์เช็คที่ type ของตัวแปร rs[1]