1. Given code in Java:

```
public class Demo {
   public static void main(String[] args) {
         System.out.print(m1(3));
   }

   public static int m1(int n) {
         if (n ==1) return 1;
         else return n * m1(n-1)
   }
}
```

(5 marks) Show how you can rewrite this program to minimize the use of activation record.

แก้ที่ตัวโปรแกรม

frame (ชุดข้อมูลสำหรับการประมวลผลของโปรแกรมย่อย)

```
public class Demo{
    public static void main(String[] args){
        System.out.print(m1(3));
    }
    public static int m1(int n){
        int ans = 1;
        for(int i=1; i<=n; i++){
            ans *= i;
        }
        return ans;
    }
}
```

หัก 2 คะแนน (ยังมีการเรียก m1)

เฉลย :

```
public class Demo {
   public static void main(String[] args) {
         int s =1;
         for(int i=1;i<=3;i++){
               s = s*i;
         }
         System.out.print(s);
}
```

//หลักการคือ เอาที่เป็นการเรียกเมธอดทิ้งให้หมด แม้แต่การเรียก m1 ก็ด้วย (แต่การปริ้นยังจำเป็นต้องเอาไว้) ถ้ายังมีส่วนอื่นที่มีการเรียกเมธอด หักที่ละ 2 คะแนน ส่วนถ้าโค้ดผิด หักที่ละ 0.5 คะแนน

2. A modified version of Java allows nested method definitions.

```
public static void main(String[] args){
        int x =5;
        int y =3;
        int z = 2;
        public void method01(int x){
                public void method2(int y){
                        x = z+1;
                        int m = x - y;
                        method3(m);
                        System.out.println(x + “,” +y +”,”+ z);  //line1
                }

                public void method3(int m){
                        int z = y +x+m;
                        y = x + z
                        System.out.println(x + “,” +y +”,”+ z);  //line2
                }

                int m = x + y + z;
                method2(m);
        }
        method1(x+1);
        System.out.println(x + “,” +y +”,”+ z);  //line3
    }
}
```

You must:

- Count method parameter declaration as a variable declaration.
- For dynamic scope, a variable declaration no longer exists if its method or scope has finished its execution.

a. (4.5 marks) If this code uses static scope, what will be printed at line1, line 2, and line 3?

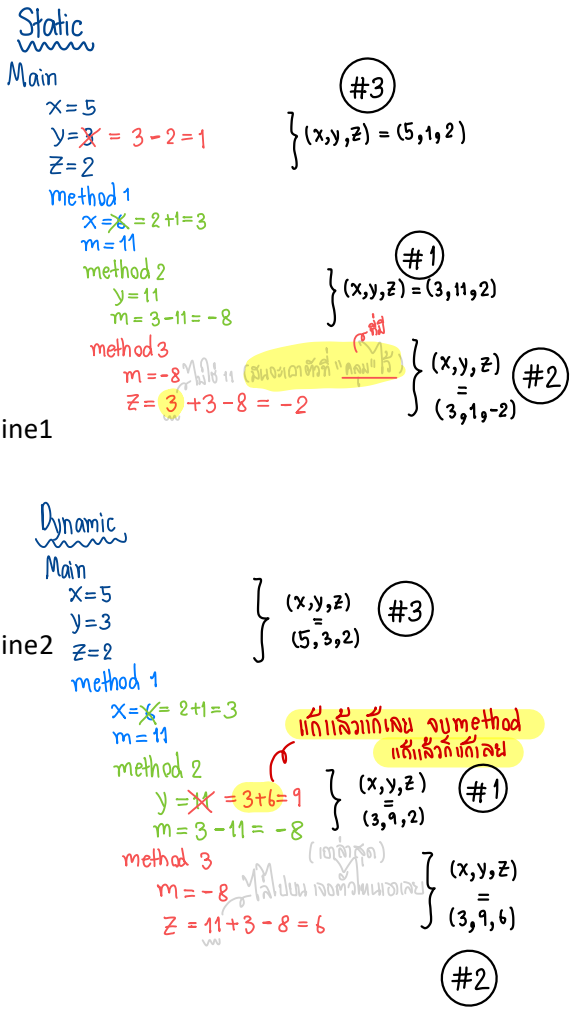
b. (4.5 marks) If this code uses dynamic scope, what will be printed at line1, line 2, and line 3?

2. A modified version of Java allows nested method definitions.

```
public static void main(String[] args){
        int x =5;
        int y =3;
        int z = 2;
        public void method01(int x){
                public void method2(int y){
                        x = z+1;
                        int m = x - y;
                        method3(m);
                        System.out.println(x + “,” +y +”,”+ z);  //line1
                }

                public void method3(int m){
                        int z = y +x+m;
                        y = x + z
                        System.out.println(x + “,” +y +”,”+ z);  //line2
                }

                int m = x + y + z;
                method2(m);
        }
        method1(x+1);
        System.out.println(x + “,” +y +”,”+ z);  //line3
    }
}
```



You must:

- Count method parameter declaration as a variable declaration.
- For dynamic scope, a variable declaration no longer exists if its method or scope has finished its execution.

a. (4.5 marks) If this code uses static scope, what will be printed at line1, line 2, and line 3?

ใน line 1 ค่าที่ print ออกมา คือ 3, 11, 2 ✓
ใน line 2 ค่าที่ print ออกมา คือ 3, 1, -2 ✓
ใน line 3 ค่าที่ print ออกมา คือ 5, 1, 2 ✓

b. (4.5 marks) If this code uses dynamic scope, what will be printed at line1, line 2, and line 3?

ใน line 1 ค่าที่ print ออกมา คือ 3, 9, 2 ** ✓ ไม่ใช่ 6 นะ (มองบน! เทศัวล่าสุด)
ใน line 2 ค่าที่ print ออกมา คือ 3, 9, 6 ✓
ใน line 3 ค่าที่ print ออกมา คือ 5, 3, 2 ✓

ห้ามมองต่ำ
ถ้าไม่มี ให้มองบนเพื่อนา

3. Given code

```
class Box { //Box containing value
public:
    Box(double l, double w, double h, double v) {
        length = l;
        width = w;
        height = h;
        value = v;
    }
    virtual double volume() { return length*width*height; }  //virtual tells compiler that the subclass can override this method
    virtual double containedValue() { return value; }  // it allows the subclass method to be called from pointer of the superclass type
protected:
    double length;
    double width;
    double height;
    double value;
};

class Cube : public Box {        //subclass of Box
public:
    Cube(double s, double v) : Box(s, s, s, v) { }    //use parent's constructor to initialize cube of equal
                                                      //sides and its value
    double containedValue() {        return value*value*value; }
    void cloneCube() {
        Cube* clone = new Cube(length, value);                    //line k
    }
};
...
void doSomething () {
  Box* boxPtr1;                                                   //line a
  Box* boxPtr2;                                                   //line b
  Box box1(1.0, 2.0, 3.0, 4.0);                                   //line c
  Cube cube1(2.0, 3.0);                                           //line d
  double result;                                                  //line e
  boxPtr1 = new Cube(3.0, 4.0);                                   //line f
  result = boxPtr1->containedValue();                             //line g
  boxPtr2 = &box1;                                                //line h
  boxPtr2 = &cube1;                                               //line i
  cube1.cloneCube();                                              //line j
}
```

สร้างตัวแปร box1 ขึ้นมา

ไม่ leak เพราะอยู่ใน stack จบฟังก์ชันก็ทำลาย

หลังจบบรรทัดนี้

อันนี้ยังไม่ถูก clear ก็จริง แต่ยังอยู่ใน Scope ที่ใช้ได้ ***
∴ ไม่เป็น memory leak

a. (3 marks) At line h and i, is there a memory leak? Explain (Thai language allowed)

ไม่มี memory leak เพราะถึงแม้ตัว boxPtr2 จะเปลี่ยนไปชี้ที่อื่น (ใน line i)
แต่ &box1 นั้น มี box1 เป็นตัวแปรอ้างถึงอยู่ จึงยังสามารถจัดการกับ box1 ได้
หรือก็คือ สามารถเข้าถึงผ่านตัวแปร box1 ได้นั่นเอง

ไม่ได้มีการใช้ "new" ∴ เก็บไว้ใน stack ปกติ

Box box1(1.0, 2.0, 3.0, 4.0);

No. Because box1 was not dynamically allocated (new was not used). So box1 is kept in stack and will be deallocated automatically when no longer used.

//อธิบายประมาณนี้ก็ได้เต็มเลย ถ้ามีอะไรแปลกๆก็มาดูเป็นเคส ๆ แต่ข้อนี้น่าจะไม่ 0 ก็ 3 เลย

b. (3 marks) After cloneCube() exits, are there memory spaces not cleared? Is there any memory leak? Explain (Thai language allowed).

มี Memory leak และ memory spaces not cleared เพราะ จาก line k
จะใช้หน่วยความจำ 2 ส่วน คือ ส่วน Stack ที่จะมีการ push local variable ชื่อ clone ซึ่งเป็น
ชนิด pointer ซึ่งจะเก็บที่อยู่ของข้อมูลจริงๆที่จะอยู่ใน heap (ข้อมูลจริงอยู่ใน heap เพราะมีการใช้คำสั่ง new)
เมื่อหมด Scope ของฟังก์ชัน cloneCube() ตัว pointer ชื่อ clone จะถูกลบไป แต่ข้อมูล
ส่วน heap ยังคงอยู่ ซึ่งเราไม่สามารถจัดการกับข้อมูลตรงนี้ได้ เพราะไม่มีตัวแปรอ้างถึง
ทำให้เกิด Memory Leak นั่นเอง

1.5

After cloneCube just exits, there are 2 places in heap that are not cleared by "delete".

1: Cube pointed to by boxPtr1. But I wouldn't call this a memory leak yet since it is still in a scope of use. //1.5 ถ้าได้ใจความตรงนี้

2: a Cube created by cloneCube(). This one is a memory leak since the variable is gone but the space allocated by "new" is still not freed. // 1.5 ถ้าได้ใจความตรงนี้