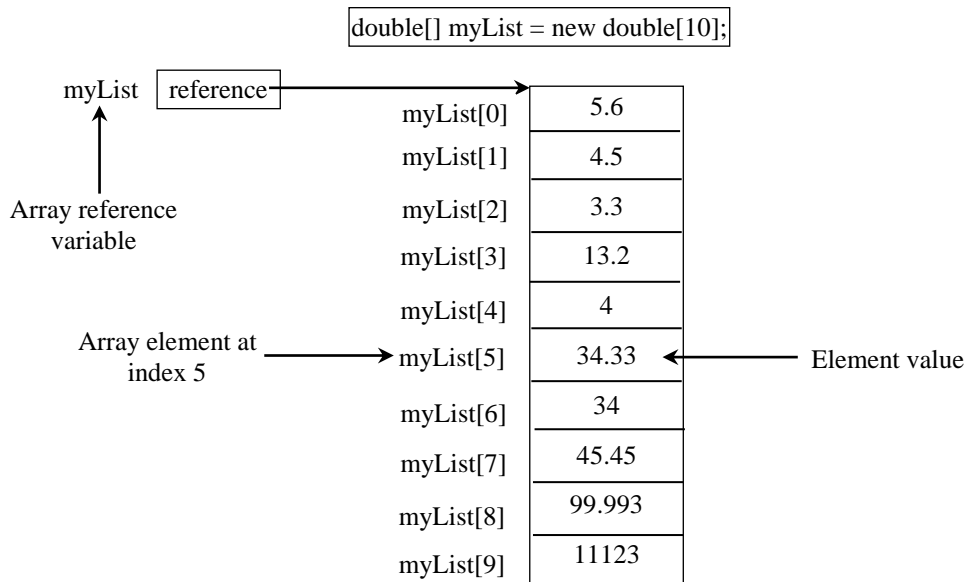


## แบบฝึกหัดปฏิบัติการคาบที่ 4: Array

### คำสั่ง

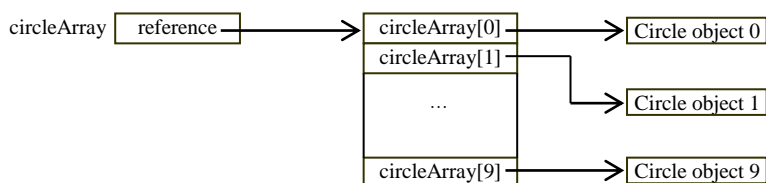
1. ให้ศึกษาการสร้างอาร์เรย์จากตัวอย่างต่อไปนี้

อาร์เรย์คือโครงสร้างข้อมูลที่มีชนิดข้อมูลประเภทเดียวกัน  
การประกาศอาร์เรย์ของชนิดข้อมูลพื้นฐาน มีรูปแบบดังนี้



### อาร์เรย์ของวัตถุ (Array of Objects)

- ▶ ตัวอย่างของการประกาศอาร์เรย์ของวัตถุ  
`Circle[] circleArray = new Circle[10];`
- ▶ อาร์เรย์ของวัตถุหรือเรียกอีกอย่างว่าอาร์เรย์ของตัวแปรอ้างอิงวัตถุ(*array of reference variables*)
- ▶ การเรียกใช้ `circleArray[1].findArea()` จะทำงานใน 2 ขั้นตอนคือ
  - ▶ การประกาศ `circleArray` จะเป็นการอ้างอิงไปยังอาร์เรย์ทั้งหมด
  - ▶ ส่วน `circleArray[1]` จะอ้างอิงไปยังวัตถุของคลาส `Circle`
- ▶ ตัวอย่างเมื่อประกาศ `Circle[] circleArray = new Circle[10];` จะมีรูปแบบของการอ้างอิงดังนี้



## 1. ให้ศึกษาและทดลองพิมพ์ Code ตัวอย่างการสร้างอาร์เรย์จากตัวอย่างต่อไปนี้

```
public class MinMaxDemo{
    public static void main(String[] args){
        int a[] ={-128, 65, -235, 99, 0, 26};
        int minIndex= findMinIdx(a);
        //int maxIndex= findMaxIdx(a);
        System.out.println("min value is a["+minIndex+"]="+a[minIndex]);
        //System.out.println("max value is a["+maxIndex+"]="+a[maxIndex]);

    }
    public static int findMinIdx(int[] a){
        int k, minIdx=0;
        for(k=1;k<a.length;k++){
            if(a[k]<a[minIdx])
            {
                minIdx=k;
            }
        }
        return minIdx;
    }
}
```

## 1.1 ผลลัพธ์ของโปรแกรมคือ

## 1.2 ให้อธิบายการทำงานของ public static int findMinIdx(int[] a)

## 1.3 ให้เพิ่มการทำงานของเมธอด public static int findMaxIdx(int[] a) สำหรับหาดำแหน่ง index ของอาร์เรย์ที่มีค่ามากที่สุด

## 2. จงอธิบายว่าเหตุใดโปรแกรมด้านล่างจึง compiles ไม่ผ่าน

โปรแกรม	ผลลัพธ์ของโปรแกรม
<pre>public class Test {     public static void main(String[] args) {         double[100] r;          for (int i = 0; i &lt; r.length(); i++);             r(i) = Math.random * 100;     } }</pre>	
<pre>public class Test {     public static void main(String[] args) {         int list[] = {1, 2, 3, 4, 5, 6};         for (int i = 1; i &lt; list.length; i++)             list[i] = list[i - 1];          for (int i = 0; i &lt; list.length; i++)             System.out.print(list[i] + " ");     } }</pre>	
<pre>public class Test {     public static void main(String[] args) {         int number = 0;         int[] numbers = new int[1];         m(number, numbers);         System.out.println("number is " + number             + " and numbers[0] is " + numbers[0]);     }     public static void m(int x, int[] y) {         x = 3;         y[0] = 3;     } }</pre>	
<pre>public class Test {     public static void main(String[] args) {         int[] list = {1, 2, 3, 4, 5};         reverse(list);         for (int i = 0; i &lt; list.length; i++)             System.out.print(list[i] + " ");     }     public static void reverse(int[] list) {         int[] newList = new int[list.length];         for (int i = 0; i &lt; list.length; i++)             newList[i] = list[list.length - 1 - i];         list = newList;     } }</pre>	
<pre>public class Test {     public static void main(String[] args) {         int[][] array = {{1, 2}, {3, 4}, {5, 6}};         for (int i = array.length - 1; i &gt;= 0; i--) {             for (int j = array[i].length - 1; j &gt;= 0; j--)                 System.out.print(array[i][j] + " ");             System.out.println();         }     } }</pre>	
<pre>public class Test {     public static void main(String[] args) {         int[][] array = {{1, 2, 3, 4}, {5, 6, 7, 8}};         System.out.println(m1(array)[0]);     } }</pre>	

```

        System.out.println(m1(array)[1]);
    }
    public static int[] m1(int[][] m) {
        int[] result = new int[2];
        result[0] = m.length;
        result[1] = m[0].length;
        return result;
    }
}

```

### 3.จากส่วนของ code ต่อไปนี้

```

1  class Test2{
2      public static void main(String[] args){
3          int[] list= {1,9,3,7,2};
4          list=dosomething(list);
5      }
6      public static int[] dosomething(int[] input){
7          int temp;
8          for (int i = 1; i < input.length; i++) {
9              for(int j = i ; j > 0 ; j--){
10                 if(input[j] < input[j-1]){
11                     temp = input[j];
12                     input[j] = input[j-1];
13                     input[j-1] = temp;
14                 }
15             }
16         }
17         for(int i=0;i<input.length;i++){
18             System.out.print(input[i]+" ");
19         }
20         return input;
21     }
22 }

```

จากโปรแกรมต่อไปนี้ให้แสดงผลลัพธ์ของโปรแกรมในแต่ละรอบของการทำงานของ Loop i

พร้อมแสดงผลลัพธ์สุดท้าย

ค่าในอาร์เรย์ list เริ่มต้น

ภายในเมธอด public static int[] dosomething(int[]input)

i=

ค่าในอาร์เรย์ input

--	--	--	--	--

i=

ค่าในอาร์เรย์ input

--	--	--	--	--

i=

ค่าในอาร์เรย์ input

--	--	--	--	--

i=

ค่าในอาร์เรย์ input

--	--	--	--	--

i=

ค่าในอาร์เรย์ input

--	--	--	--	--

ผลลัพธ์สุดท้ายของโปรแกรม

4 กำหนดให้  $A[0...n-1]$  เป็นชุดข้อมูลที่เป็นจำนวนเต็ม  $n$  จำนวน จงเขียนโปรแกรมแบบ OOP ในการเรียงข้อมูล  $A[0...n-1]$  จากน้อยไปมาก พร้อมทั้งหาความถี่สะสมของข้อมูลแต่ละตัว ตัวอย่างเช่น  $A = [9\ 5\ 9\ 5\ 8]$

ข้อมูลที่เรียงจากน้อยไปมาก	5	8	9
ความถี่ของข้อมูลแต่ละตัว	2	3	5

ให้นักศึกษานิยาม Class ชื่อ AscendSortFreq ที่ประกอบไปด้วย

- Data fields ชนิด double[ ] A ซึ่งแสดงชุดข้อมูล จำนวน  $n$  ตัว
- Constructor ที่กำหนดค่าให้แก่ double[ ] A จำนวน  $n$  ตัว
- Method AscendSort(double[ ] A) ที่ return array B ที่เรียงจากน้อยไปมาก
- Method SortCommuFreq(double[ ] B) ที่ return array C ความถี่ของข้อมูลแต่ละตัว

5. [MatrixMultiplication] ให้เขียน class MatrixMultiplication ซึ่งทำการคำนวณหาผลคูณของสองเมทริกซ์ การคูณเมทริกซ์ทำได้เมื่อจำนวนคอลัมน์ของเมทริกซ์แรกมีค่าเท่ากับจำนวนแถวของเมทริกซ์ที่สอง ดังนั้นถ้าเมทริกซ์ A เป็นเมทริกซ์ขนาด  $N \times L$  และเมทริกซ์ B มีขนาดเป็น  $L \times M$  แล้วผลลัพธ์ของการคูณเมทริกซ์คือ  $C = A \times B$  จะมีขนาดเป็น  $N \times M$  โดยที่สมาชิกแต่ละตัวของเมทริกซ์ C มีค่าดังสมการต่อไปนี้

$$C_{ik} = a_{i1}b_{1k} + a_{i2}b_{2k} + a_{i3}b_{3k} + \dots + a_{iL}b_{Lk}$$

ตัวอย่าง ให้  $A = \begin{bmatrix} 1 & 2 \\ -1 & 0 \\ 3 & 2 \end{bmatrix}, B = \begin{bmatrix} 1 & 5 & 2 \\ -2 & 0 & 1 \end{bmatrix}$

วิธีทำ  $AB = \begin{bmatrix} 1 & 2 \\ -1 & 0 \\ 3 & 2 \end{bmatrix} \begin{bmatrix} 1 & 5 & 2 \\ -2 & 0 & 1 \end{bmatrix}$

ใช้หลักการแถว คูณ หลัก

$$= \begin{bmatrix} (1)(1) + (2)(-2) & (1)(5) + (2)(0) & (1)(2) + (2)(1) \\ (-1)(1) + (0)(-2) & (-1)(5) + (0)(0) & (-1)(2) + (0)(1) \\ (3)(1) + (2)(-2) & (3)(5) + (2)(0) & (3)(2) + (2)(1) \end{bmatrix}$$

$$= \begin{bmatrix} -3 & 5 & 4 \\ -1 & -5 & -2 \\ -1 & 15 & 8 \end{bmatrix}$$

ข้อมูลนำเข้า ขนาดของเมทริกซ์ A และข้อมูลในเมทริกซ์ A

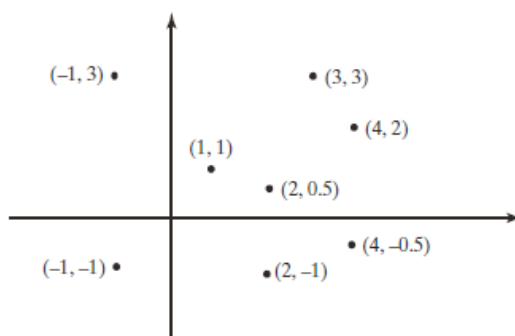
ขนาดของเมทริกซ์ B และข้อมูลในเมทริกซ์ B

ข้อมูลส่งออก ผลคูณของสองเมทริกซ์

ข้อมูลนำเข้า	ข้อมูลส่งออก
3 2	-3 5 4
1 2	-1 -5 -2
-1 0	-1 15 8

3 2	
2 3	
1 5 2	
-2 0 1	

6. จงเขียนโปรแกรมเพื่อคำนวณหาจุดที่ใกล้เคียงกันมากที่สุด



ข้อมูลนำเข้า บรรทัดแรกรับจำนวนจุด n จุด  
n บรรทัดถัดไปแสดงพิกัดของจุดแต่ละจุด

ข้อมูลส่งออก ระยะทางคู่จุดที่ใกล้เคียงกันมากที่สุด

ข้อมูลนำเข้า	ข้อมูลส่งออก
8 3 3 3 1- 1 1 2 4 0.5 2 1- 1- 0.5- 4 2 -1	1.12

## 7. minTwoSet

กำหนดให้ จำนวนเต็ม n จำนวน โดยมีอยู่ m จำนวนอยู่ในกลุ่ม A และ n-m คนอยู่ในกลุ่ม B จงเขียนโปรแกรมหาคำตอบของผลต่างของผลรวมในแต่ละกลุ่มน้อยที่สุดที่สามารถหาได้จากสมการ

$$\min \left( \left| \sum_{i=1, i \in A}^m w_i - \sum_{j=1, j \in B}^{n-m} w_j \right| \right)$$

$w_i \in n, w_j \in n$  โดย และ  $| \quad |$  คือ ค่าสัมบูรณ์ เช่น  $|-3| = 3$  หรือ  $|3| = 3$

เขียนโปรแกรมหาค่าผลต่างของผลรวมในแต่ละกลุ่มน้อยที่สุด (เขียนแบบ OOP)

ข้อมูลนำเข้า อ่านจาก Standard Input

บรรทัดแรก ระบุ จำนวน n โดย  $3 \leq n < 1000$

บรรทัดสอง ระบุค่า  $w_i$  จำนวนเต็มบวก n จำนวนที่ไม่ซ้ำกัน โดยแต่ละจำนวนคั่นด้วยช่องว่าง

## ข้อมูลส่งออก ส่งออกไปยัง Standard Output

แสดงผลลัพธ์ของผลต่างของผลรวมในแต่ละกลุ่มน้อยที่สุด

## ตัวอย่างข้อมูล

ข้อมูลนำเข้า	ข้อมูลส่งออก
4 3 101 99 1	0
5 14 16 47 25 2	6
6 7 2 1 3 6 0	1

8. [Find Pokemon] ญาณู้าผู้เล่นหน้าใหม่ของเกมสโโปเกมอนต้องการจับตัวปิกาจู เนื่องจากปิกาจูเป็นตัวที่หาได้ยากมาก ดังนั้นญาณู้าจึงไปรวบรวมสถิติการเกิดตัวโปเกมอน ณ พิกัดต่าง ๆ โดยพบว่าในแต่ละพิกัดจะมีความถี่ของการเกิดตัวโปเกมอนตัวต่าง ๆ ที่ไม่เท่ากัน โดยมีการเก็บข้อมูลเป็นรูปขนาด HxW ช่อง โดยญาณู้าต้องการหาปิกาจูจากรูปนี้ ตัวอย่างของรูปขนาด 4x5 แสดงเป็นตารางด้านล่าง กำหนดตารางชื่อ A ตัวเลขในแต่ละช่องแสดงความถี่ของการเกิดตัวโปเกมอนในช่องนั้น

5	1	2	10	4
4	30	3	0	100
3	25	10	4	10
3	20	4	8	5

ในการหาตำแหน่งของปิกาจูจะมีเงื่อนไข 3 ข้อดังนี้

1. ปิกาจูจะปรากฏเป็น 2 ช่องติดกันพอดี
2. สองช่องที่เป็นบริเวณที่มีปิกาจูควรมีค่าความถี่ของการเกิดตัวโปเกมอนในช่องนั้น ต่างกันไม่เกิน 10
3. ตำแหน่งของปิกาจูน่าจะเป็นตำแหน่งที่มีความถี่ของการเกิดตัวโปเกมอนสูงก็ต้องเป็นสองช่องที่มีผลรวมของค่าความถี่ของการเกิดตัวโปเกมอนมากที่สุด

จากตารางตำแหน่งที่ตรงตามเงื่อนไขคือ A[2][2] และ A[3][2] จงเขียนโปรแกรมที่รับตารางแสดงความถี่ของการเกิดตัวโปเกมอน จากนั้นให้หาตำแหน่งมุมบนซ้ายของช่องที่น่าจะเกิดปิกาจูมากที่สุด โดยระบุแถวและคอลัมน์ช่องนั้น

## ข้อมูลนำเข้า

บรรทัดแรก ระบุขนาดตาราง HxW

บรรทัดที่ 2 ถึง H+1 แสดงความถี่ของการเกิดตัวโปเกมอนในแถวที่ i โดยระบุเป็นจำนวนเต็มจำนวน W ตัว จำนวนที่ j จะเป็นความถี่ของการเกิดตัวโปเกมอนในช่องที่อยู่ในคอลัมน์ j

## ข้อมูลส่งออก

มีบรรทัดเดียว คือ มุมบนซ้ายของช่องที่น่าจะเกิดตัวปิกาจูมากที่สุดโดยระบุแถวและคอลัมน์ช่องนั้น

ตัวอย่างข้อมูลนำเข้า	ตัวอย่างข้อมูลส่งออก
4 5 5 1 2 10 4 4 30 3 0 100 3 25 10 4 10 3 20 4 8 5	2 2
4 4 0 0 0 0 0 0 0 0 0 1 1 1 1 1 0 0	3 2

9. จงเขียนเมธอด `isConsecutiveFour(int[][] values)` เพื่อตรวจสอบอาร์เรย์สองมิติต่อไปนี้ว่ามีตัวเลขตัวเดียวกันเรียงต่อกัน ครบสี่ตัวหรือไม่ ในแนวตั้ง แนวนอน หรือแนวทะแยง

**public static boolean isConsecutiveFour(int[][] values)**

จากนั้นให้เขียนโปรแกรมทดสอบที่ผู้ใช้สามารถป้อนจำนวนแถวและคอลัมน์ของอาร์เรย์สองมิติและป้อนค่าภายในอาร์เรย์ โดยเมื่อป้อนค่าไปแล้วให้เรียกใช้เมธอด `isConsecutiveFour(int[][] values)` โดยการคืนค่า `true` กรณีที่ภายในอาร์เรย์มีตัวเลข ตัวเรียงต่อกัน กรณีอื่นให้คืน 4ค่าเป็น `false`

**ตัวอย่าง** เมื่อส่งอาร์เรย์ต่อไปนี้เข้าไปที่เมธอดจะคืนค่า `true` ออกมา

0 1 0 3 1 6 1 0 1 6 8 6 0 1 5 6 2 1 8 2 9 6 5 6 1 1 9 1 1 3 6 1 4 0 7 3 3 3 3 4 0 7	0 1 0 3 1 6 1 0 1 6 8 6 0 1 5 5 2 1 8 2 9 6 5 6 1 1 9 1 1 5 6 1 4 0 7 3 5 3 3 4 0 7	0 1 0 3 1 6 1 0 1 6 8 6 0 1 5 6 2 1 6 2 9 6 5 6 6 1 9 1 1 3 6 1 4 0 7 3 6 3 3 4 0 7	0 1 0 3 1 6 1 0 1 6 8 6 0 1 9 6 2 1 8 2 9 6 9 6 1 1 9 1 1 3 9 1 4 0 7 3 3 3 9 4 0 7
--	--	--	--

**ข้อมูลนำเข้า** จำนวนแถวและคอลัมน์ของอาร์เรย์สองมิติและป้อนค่าภายในอาร์เรย์

**ข้อมูลส่งออก** คืนค่า 1 กรณีที่ภายในอาร์เรย์มีตัวเลข ตัวเรียงต่อกัน กรณีอื่นให้คืนค่าเป็น 0 4

ข้อมูลนำเข้า	ข้อมูลส่งออก
7 6 0 1 0 3 1 6 1 0 1 6 8 6 0 1 5 6 2 1 8 2 9 6 5 6 1 1 9 1 1 3 6 1 4 0 7 3 3 3 3 4 0 7	1



## 10. (Car) ขับรถหลบสิ่งกีดขวาง

ในการแสดงขับรถผาดโผนบนถนนที่มีเลนทั้งหมด  $m$  เลน โดยให้หมายเลขประจำเลนจากซ้ายไปขวามีค่าตั้งแต่ 1 จนถึง  $m$  ตามลำดับ นักแสดงขับรถผาดโผนต้องบังคับรถให้แล่นไปบนถนนดังกล่าวให้ปลอดภัยตลอดระยะเวลา  $t$  หน่วย การแสดงเริ่มต้น ณ เวลา 0 ถึง  $t$  นักแสดงขับรถผาดโผนอยู่ในเลนที่  $n$  ในแต่ละ 1 หน่วยเวลา อาจมีสิ่งกีดขวางตกลงมายังถนนบางเลน ทำให้เขาต้องบังคับรถเพื่อหลีกเลี่ยงสิ่งกีดขวาง ซึ่งมีทางเลือกในการบังคับรถอยู่ 3 แบบ ได้แก่ 1 หมายถึง การเปลี่ยนเลนไปทางซ้าย 1 เลนในเวลาถัดไปไปยังเลนที่มีหมายเลขประจำเลนน้อยกว่า 2 หมายถึงการเปลี่ยนเลนไปทางขวา 1 เลนในเวลาถัดไป (ไปยังเลนที่มีหมายเลขประจำเลนมากกว่า) และ 3 หมายถึง การขับอยู่ในเลนเดิม กำหนดให้ถนนเป็นเส้นตรงตลอดทาง จงเขียนโปรแกรมเพื่อบังคับให้รถแล่นไปตามเส้นทางนี้โดยปลอดภัย โดยชุดข้อมูลทดสอบจะมีคำตอบที่ถูกต้องเพียง 1 คำตอบเสมอ

ข้อมูลนำเข้า

- บรรทัดแรกระบุจำนวนเลน  $m$  โดยที่  $2 \leq m \leq 40$
- บรรทัดที่สองระบุหมายเลขเลนเริ่มต้น  $1 \leq n \leq m$
- บรรทัดที่สามระบุระยะเวลา  $t$  โดยที่  $1 \leq t \leq 100$
- บรรทัดที่สี่ถึงบรรทัดที่  $t+3$  แสดงสถานะของถนน ณ เวลา  $t=1, 2, \dots, K$  ตามลำดับ แต่ละบรรทัดระบุตัวเลข  $m$  ตัวเลขแต่ละตัวแสดงสถานะของถนนตั้งแต่เลนที่ 1 ถึงเลนที่  $m$  โดยเลข 0 หมายถึงเลนนั้นไม่มีสิ่งกีดขวาง และเลข 1 หมายถึงมีสิ่งกีดขวางอยู่

ข้อมูลส่งออก

มีอยู่  $t$  บรรทัด แต่ละบรรทัดมีตัวเลข 1 ตัวเพื่อแสดงถึงทางเลือกในการบังคับรถของนักแสดงขับรถผาดโผน ในแต่ละช่วงเวลา บรรทัดที่  $i$  หมายถึงการเปลี่ยนเลนจากเวลาที่  $i-1$  ไปยังเวลาที่  $i$  เมื่อ  $i=1, 2, \dots, t$  โดยที่เลข 1 จะหมายถึงขับไปทางซ้าย 1 เลน, เลข 2 หมายถึงขับไปทางขวา 1 เลน, และเลข 3 หมายถึงขับอยู่ในเลนเดิม

ตัวอย่างที่ 1

ข้อมูลนำเข้า	ข้อมูลส่งออก
7	1
5	1
5	1
0 0 0 0 0 0	1
0 0 0 0 0 0	2
0 0 0 0 0 0	
0 1 1 0 0 0	
1 0 1 1 1 1	

## 11. [Oil Deposits] บ่อน้ำมัน

นักธรณีวิทยาต้องการสำรวจหาแหล่งน้ำมันบนพื้นที่สี่เหลี่ยมขนาดใหญ่เรียกว่า GRID โดยบริษัทได้ตีเส้นใน GRID ให้อยู่ในรูปของตาราง และทำการสำรวจโดยใช้เครื่องมือสำหรับตรวจจับน้ำมันว่าพื้นที่สี่เหลี่ยมที่สำรวจมีน้ำมันอยู่หรือไม่ บริเวณที่มีน้ำมันจะเรียกว่า pocket ถ้ามี pocket ในตารางแต่ละช่องต่อกันไม่ว่าจะเป็นแนวตั้ง แนวนอน แนวทะแยง จะถือว่าเป็นบ่อน้ำมันบ่อเดียวกัน งานที่ต้องทำคือให้ตอบคำถามว่าในพื้นที่ พื้นที่ขนาดใหญ่ 1GRID มีบ่อน้ำมันกี่บ่อ

## ข้อมูลนำเข้า

ข้อมูลนำเข้าประกอบด้วย หรือมากกว่า 1 1GRID แต่ละ GRID จะเริ่มต้นด้วยตัวเลข m และ n ซึ่งแทนด้วยจำนวนแถวและจำนวนคอลัมน์ใน GRID กรณีที่  $m=0$  แสดงว่าไม่มีการรับข้อมูลต่อ กำหนดให้  $1 \leq m \leq 100$  และ  $1 \leq n \leq 100$  แถวต่อไปจำนวน m แถวจะมีแถวละ n ตัวอักษร แต่ละตัวอักษรแทนด้วยสถานะของน้ำมัน ณ ตารางดังกล่าวโดย '\*' แทนไม่มีน้ำมัน '@' แทน oil pocket.

## ข้อมูลส่งออก

สำหรับแต่ละ GRID จะแสดงจำนวนของบ่อน้ำมันที่ค้นพบ

ข้อมูลนำเข้า	ข้อมูลส่งออก
1 1 *	0
3 5 * @ * @ * **@** *@*@*	1
1 8 @@*****@*	2
5 5 *****@ *@@*@ *@**@ @@@@*@ @@@@*@	2