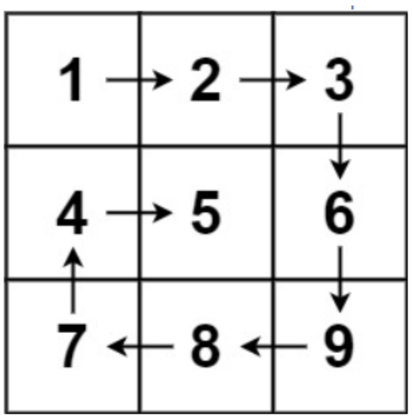
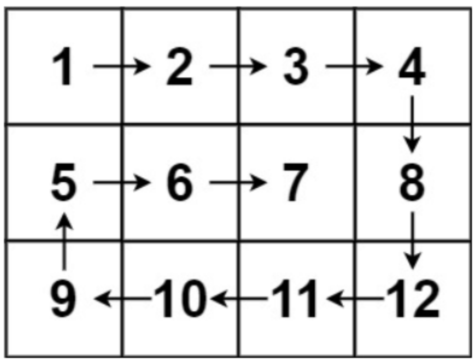
周作業1027

1. 使用者輸入二維陣列的列數與行數：r, c，然後程式以隨機方式產生此二維陣列內容(，每個元素值為小於200的整數)，若此陣列包含saddle point, 則印出此陣列內容、與saddle point 的列編號、行編號、以及數值，若此陣列不包含saddle points，則程式重新產生相同大小的二維陣列，直到找到有saddle point 的陣列為止。saddle point 為陣列元素，它是同一列所有元素的最小值、也是同一行元素的最小值。
2. 讀入一陣列內容、以及目標合 sum，找出此陣列中哪一段子陣列的元素總和加總等於 sum。例如：輸入陣列 {6, 2, 13, 9, 4, 2, 8, 77, 45, 6}, 若 sum=87, 程式輸出：2+8+77=87；若sum=30, 程式輸出：2+13+9+4+2=30；若程式找不到符合的子陣列加總等於sum，則印出 No-match
3. 讀入任意形狀的二維陣列，以spiral order 印出陣列內容，例如：



```
Input: matrix = [[1,2,3],[4,5,6],[7,8,9]]
Output: [1,2,3,6,9,8,7,4,5]
```



```
Input: matrix = [[1,2,3,4],[5,6,7,8],[9,10,11,12]]
Output: [1,2,3,4,8,12,11,10,9,5,6,7]
```

第一題:

```
/* 1027 楊育哲 周作業
 * 實作第一題: 找隨機r*c陣列裡鞍點
 */
import java.util.Scanner;
public class h1_1027_w {
    public static void main(String args[]){
        Scanner sc = new Scanner(System.in);
        int r=0, c=0, current=0;
        System.out.println("依序輸入兩個數字r, c:");
        r = sc.nextInt();
        c = sc.nextInt();
        int[][] arr=new int[r][c];
        int[][] saddlePoints=new int[r*c][2];
        boolean flag=false;
        while(!flag){//找到符合的二維陣列前重複執行
            for(int i=0; i<r; i++){
                for(int j=0; j<c; j++) arr[i][j]=(int)(Math.random()*201);
            }
            for(int i=0; i<r; i++){
                int min=201, minIndex=0;
                for(int j=0; j<c; j++){
                    if(arr[i][j]<min){
                        min=arr[i][j];
                        minIndex=j;
                    }
                }
                boolean check=true;
                for(int k=0; k<r; k++)
                    if(arr[k][minIndex]<min) check=false;
                if(check){//找到鞍點, while跳脫條件達成、將座標加入鞍點表
                    flag=true;
                    saddlePoints[current][0]=i;
                    saddlePoints[current++][1]=minIndex;
                }
            }
        }
        for(int i=0; i<r; i++){ //輸出目標二維陣列
            for(int j=0; j<c; j++) System.out.printf("%3d, ", arr[i][j]);
            System.out.println("");
        }
        for(int i=0; i<current; i++) //輸出所有saddle point
            System.out.printf("saddlePoint%d: arr[%d, %d]=%d\n", i+1, saddlePoints[i][0], saddlePoints[i][1], arr[saddlePoints[i][0]][s
    }
}
```

程式解說: 輸入兩個數字r, c，重複隨機建立二維陣列，直到找到有鞍點在的陣列。

```
for(int i=0; i<r; i++){
    int min=201, minIndex=0;
    for(int j=0; j<c; j++){
        if(arr[i][j]<min){
            min=arr[i][j];
            minIndex=j;
        }
    }
    boolean check=true;
    for(int k=0; k<r; k++)
        if(arr[k][minIndex]<min) check=false;
    if(check){//找到鞍點，while跳脫條件達成、將座標加入鞍點表
        flag=true;
        saddlePoints[current][0]=i;
        saddlePoints[current++][1]=minIndex;
    }
}
```

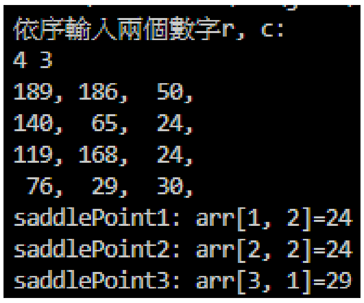
←while內的程式片段，負責找有無鞍點，若有則加進saddlePoints裡。

運作情形為: 歷遍每排，確認其中一排最小的值亦是那列最小的值。

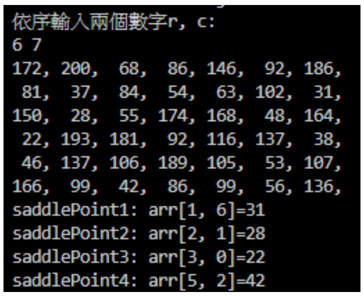
check即為'確認'是否符合鞍點的布林變數。

而flag則為確認有至少有一鞍點的布林變數。

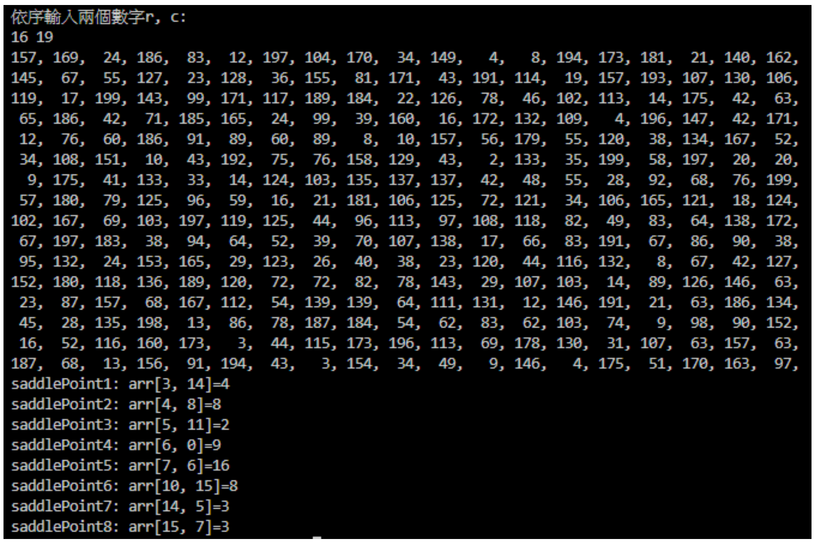
ex.



```
依序輸入兩個數字r, c:
4 3
189, 186,  50,
140,  65,  24,
119, 168,  24,
 76,  29,  30,
saddlePoint1: arr[1, 2]=24
saddlePoint2: arr[2, 2]=24
saddlePoint3: arr[3, 1]=29
```



```
依序輸入兩個數字r, c:
6 7
172, 200,  68,  86, 146,  92, 186,
 81,  37,  84,  54,  63, 102,  31,
150,  28,  55, 174, 168,  48, 164,
 22, 193, 181,  92, 116, 137,  38,
 46, 137, 106, 189, 105,  53, 107,
166,  99,  42,  86,  99,  56, 136,
saddlePoint1: arr[1, 6]=31
saddlePoint2: arr[2, 1]=28
saddlePoint3: arr[3, 0]=22
saddlePoint4: arr[5, 2]=42
```



```
依序輸入兩個數字r, c:
16 19
157, 169,  24, 186,  83,  12, 197, 104, 170,  34, 149,   4,   8, 194, 173, 181,  21, 140, 162,
145,  67,  55, 127,  23, 128,  36, 155,  81, 171,  43, 191, 114,  19, 157, 193, 107, 130, 106,
119,  17, 199, 143,  99, 171, 117, 189, 184,  22, 126,  78,  46, 102, 113,  14, 175,  42,  63,
 65, 186,  42,  71, 185, 165,  24,  99,  39, 160,  16, 172, 132, 109,   4, 196, 147,  42, 171,
 12,  76,  60, 186,  91,  89,  60,  89,   8,  10, 157,  56, 179,  55, 120,  38, 134, 167,  52,
 34, 108, 151,  10,  43, 192,  75,  76, 158, 129,  43,   2, 133,  35, 199,  58, 197,  20,  20,
  9, 175,  41, 133,  33,  14, 124, 103, 135, 137, 137,  42,  48,  55,  28,  92,  68,  76, 199,
 57, 180,  79, 125,  96,  59,  16,  21, 181, 106, 125,  72, 121,  34, 106, 165, 121,  18, 124,
102, 167,  69, 103, 197, 119, 125,  44,  96, 113,  97, 108, 118,  82,  49,  83,  64, 138, 172,
 67, 197, 183,  38,  94,  64,  52,  39,  70, 107, 138,  17,  66,  83, 191,  67,  86,  90,  38,
 95, 132,  24, 153, 165,  29, 123,  26,  40,  38,  23, 120,  44, 116, 132,   8,  67,  42, 127,
152, 180, 118, 136, 189, 120,  72,  72,  82,  78, 143,  29, 107, 103,  14,  89, 126, 146,  63,
 23,  87, 157,  68, 167, 112,  54, 139, 139,  64, 111, 131,  12, 146, 191,  21,  63, 186, 134,
 45,  28, 135, 198,  13,  86,  78, 187, 184,  54,  62,  83,  62, 103,  74,   9,  98,  90, 152,
 16,  52, 116, 160, 173,   3,  44, 115, 173, 196, 113,  69, 178, 130,  31, 107,  63, 157,  63,
187,  68,  13, 156,  91, 194,  43,   3, 154,  34,  49,   9, 146,   4, 175,  51, 170, 163,  97,
saddlePoint1: arr[3, 14]=4
saddlePoint2: arr[4, 8]=8
saddlePoint3: arr[5, 11]=2
saddlePoint4: arr[6, 0]=9
saddlePoint5: arr[7, 6]=16
saddlePoint6: arr[10, 15]=8
saddlePoint7: arr[14, 5]=3
saddlePoint8: arr[15, 7]=3
```

第二題:

```
/* 1027 楊育哲 周作業
 * 實作第二題: 取一數串的子數串，確認其可以相加乘所求數字
 */
import java.util.Scanner;
public class h2_1027_w {
    public static void main(String args[]){
        Scanner sc = new Scanner(System.in);
        System.out.println("輸入一數串(以空格分開):");
        String s=sc.nextLine();
        System.out.print("輸入一數字(target):");
        int L=1, current=-1, target=sc.nextInt();
        for(int i=0; i<s.length(); i++) if(s.charAt(i)==' ') L++;
        int[] arr=new int[L];
        for(int i=0; i<L; i++){
            int num=0;
            while(current<s.length()-1&&s.charAt(++current)!=' '){
                num = num*10+(int)(s.charAt(current)-'0');
            }
            arr[i] = num;
        }
        boolean match=false;
        int[][] ans=new int[L][];
        current=0;
        for(int i=0; i<L; i++){
            for(int j=i; j<L; j++){
                int num=0;
                for(int k=i; k<=j; k++) num+=arr[k];
                if(num==target){
                    match = true;
                    ans[current] = new int[j-i+2];
                    ans[current][0]=j-i+1;
                    for(int k=i; k<=j; k++) ans[current][k-i+1]=arr[k];
                    current++;
                }
            }
        }
        if(match){//若找到符合之子串，將全部符合的子字串都列印出
            for(int i=0; i<current; i++){
                for(int j=0; j<ans[i][0]-1; j++) System.out.printf("%d+", ans[i][j+1]);
                System.out.printf("%d=%d\n", ans[i][ans[i][0]], target);
            }
        }else System.out.println("No-match");
    }
}
```

程式解說: 歷遍所有仔字串，確認其可以相加成目標之target。

```
int L=1, current=-1, target=sc.nextInt();
for(int i=0; i<s.length(); i++) if(s.charAt(i)==' ') L++;
int[] arr=new int[L];
for(int i=0; i<L; i++){
    int num=0;
    while(current<s.length()-1&&s.charAt(++current)!=' '){
        num = num*10+(int)(s.charAt(current)-'0');
    }
    arr[i] = num;
}
```

←計算輸入字串長度L，並將字串拆成一長度為L的數列

(寫法和當天作業的第一題字串輸入處理片段一樣)

```
boolean match=false;
int[][] ans=new int[L][];
current=0;
for(int i=0; i<L; i++){
    for(int j=i; j<L; j++){
        int num=0;
        for(int k=i; k<=j; k++) num+=arr[k];
        if(num==target){
            match = true;
            ans[current] = new int[j-i+2];
            ans[current][0]=j-i+1;
            for(int k=i; k<=j; k++) ans[current][k-i+1]=arr[k];
            current++;
        }
    }
}
```

←尋過所有子字串，若其可以相加成target，則加進ans陣列中，

並把match設成true，表示有找到符合子字串。

ex.

```
輸入一數串(以空格分開):
6 2 13 9 4 2 8 77 45 6
輸入一數字(target):87
2+8+77=87
```

```
輸入一數串(以空格分開):
6 2 13 9 4 2 8 77 45 6
輸入一數字(target):30
6+2+13+9=30
2+13+9+4+2=30
```

第三題:

```java
/* 1027 楊育哲 周作業
 * 實作第三題: 以spiral order印出陣列內容，陣列由使用者輸入，其為任意形狀
 * matrix = [[1,2,3],[4,5,6],[7,8,9]]
 * matrix = [[1,2,3,4],[5,6,7,8],[9,10,11,12]]
 */
import java.util.Scanner;
public class h3_1027_w {
    public static void main(String args[]){
        Scanner sc = new Scanner(System.in);
        System.out.println("輸入符合規定的二維陣列(陣列區只包含'['、']'、','、數字, 且得每排、排列個數一樣):");
        String s = sc.nextLine();
        int index=0;
        while(s.charAt(index)!='[') index++;
        int current=index+1, r=0, c=1;
        while(current<s.length())
            if(s.charAt(current++)=='[') r++;
        current=index;
        while(s.charAt(current)!=']')
            if(s.charAt(current++)==',') c++;
        int[][] arr=new int[r][c];
        current=index+2;
        for(int i=0; i<r; i++){
            for(int j=0; j<c; j++){
                int num=0;
                while(current<s.length()&&s.charAt(current)!=','&&s.charAt(current)!=']'){
                    num = num*10+(int)(s.charAt(current++)-'0');
                }
                arr[i][j] = num;
                current+=1;
            }
            current+=2;
        }
        current = 0;
        int end=r*c, ceil=0;
        int[] cood={0, 0}, move={0, 0};
        int state=0;//0:r,右至左 | 1:c,上至下 | 2:r,左至右 | 3:c,下至上
        while(current<end){
            index = 0;
            ceil = (state%2==1)? r:c;
            move[0] = (state%2==1)? -1*(state-2):0;
            move[1] = (state%2==1)? 0:-1*(state-1);
            while(index<ceil-1){
                System.out.printf("%3d ", arr[cood[0]][cood[1]]);
                cood[0]+=move[0];
                cood[1]+=move[1];
                current++;
                index++;
            }
            state = (state+1)%4;
            if(state==3){
                r--;
                c--;
            }
        }
    }
}
```

程式解說:

←左方程式片段負責將輸入字串拆成二維陣列

```java
int index=0;
while(s.charAt(index)!='[') index++;
int current=index+1, r=0, c=1;
while(current<s.length())
    if(s.charAt(current++)=='[') r++;
current=index;
while(s.charAt(current)!=']')
    if(s.charAt(current++)==',') c++;
int[][] arr=new int[r][c];
current=index+2;
for(int i=0; i<r; i++){
    for(int j=0; j<c; j++){
        int num=0;
        while(current<s.length()&&s.charAt(current)!=','&&s.charAt(current)!=']'){
            num = num*10+(int)(s.charAt(current++)-'0');
        }
        arr[i][j] = num;
        current+=1;
    }
    current+=2;
}
```

```
current = 0;
int end=r*c, ceil=0;
int[] cood={0, 0}, move={0, 0};
int state=0;//0:r,右至左 | 1:c,上至下 | 2:r,左至右 | 3:c,下至上
while(current<end){
    index = 0;
    ceil = (state%2==1)? r:c;
    move[0] = (state%2==1)? -1*(state-2):0;
    move[1] = (state%2==1)? 0:-1*(state-1);
    while(index<ceil-1){
        System.out.printf("%3d ", arr[cood[0]][cood[1]]);
        cood[0]+=move[0];
        cood[1]+=move[1];
        current++;
        index++;
    }
    state = (state+1)%4;
    if(state==3){
        r--;
        c--;
    }
}
```

←左方程式片段負責將二維陣列輸出成spiral order形式

ex.

```
輸入符合規定的二維陣列(陣列區只包含'['、']'、','、數字, 且得每排、排列個數一樣):
matrix = [[1,2,3],[4,5,6],[7,8,9]]
1   2   3   6   9   8   7   4   5
輸入符合規定的二維陣列(陣列區只包含'['、']'、','、數字, 且得每排、排列個數一樣):
matrix = [[1,2,3,4],[5,6,7,8],[9,10,11,12]]
1   2   3   4   8  12  11  10   9   5   6   7
```