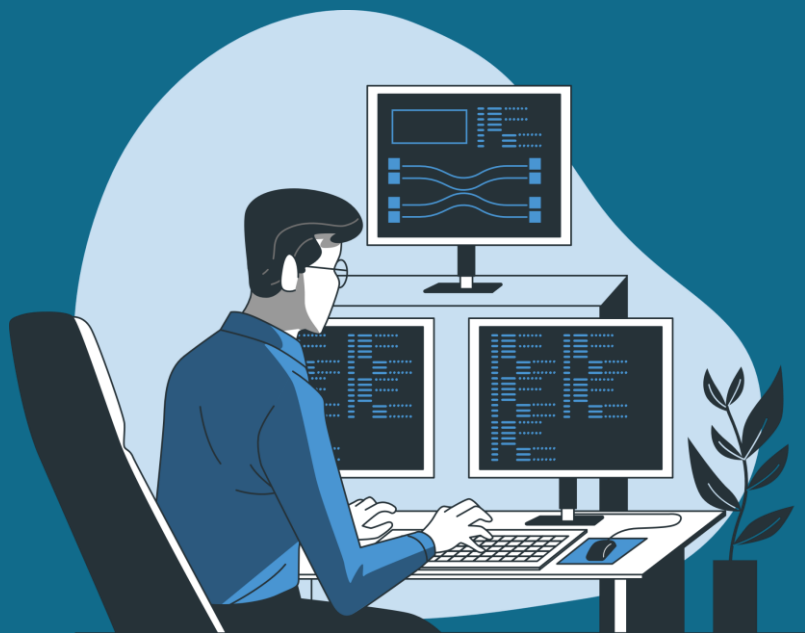




28TECH

Become A Better Developer



MẢNG HAI CHIỀU





Mảng hai chiều:

Mảng 2 chiều được sử dụng trong các bài toán liên quan tới ma trận, bảng số,... Các bạn có thể coi mảng 2 chiều chính là các mảng một chiều được xếp chồng lên nhau.





1. Khai báo mảng 2 chiều:



Khi khai báo mảng 1 chiều, các bạn cần chỉ ra số hàng, số cột của ma trận.

Khai báo mảng 2 chiều
có 3 hàng và 3 cột

EXAMPLE

```
int[][] a = {  
    {1, 2, 3},  
    {4, 5, 6},  
    {7, 8, 9}  
};  
int[][] b = new int[3][3];
```

1	2	3
4	5	6
7	8	9





2. Truy cập vào các phần tử trong mảng 2 chiều:



Để truy cập vào phần tử trong mảng, các bạn dùng chỉ số hàng và chỉ số cột. Chỉ số hàng và cột của mảng 2 chiều được đánh số từ 0 tương tự như mảng 1 chiều.

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

$a[0][2] = 3$

$a[2][1] = 8$





3. Nhập và duyệt mảng 2 chiều:



Để nhập mảng 2 chiều từ bàn phím ta duyệt qua từng hàng, mỗi hàng duyệt qua từng cột.

EXAMPLE

```
public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    int n = sc.nextInt(); //row
    int m = sc.nextInt(); //col
    int[][] a = new int[n][m];
    for(int i = 0; i < n; i++)
        for(int j = 0; j < m; j++)
            a[i][j] = sc.nextInt();
    for(int i = 0; i < n; i++)
        for(int j = 0; j < m; j++)
            System.out.print(a[i][j] + " ");
}
```





4. Các bài toán cơ bản trên mảng 2 chiều:

a. Tìm phần tử lớn nhất, nhỏ nhất trong mảng:

EXAMPLE

```
public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    int n = sc.nextInt(); //row
    int m = sc.nextInt(); //col
    int[][] a = new int[n][m];
    int minVal = Integer.MAX_VALUE, maxVal = Integer.MIN_VALUE;
    for(int i = 0; i < n; i++){
        for(int j = 0; j < m; j++){
            a[i][j] = sc.nextInt();
            minVal = Math.min(minVal, a[i][j]);
            maxVal = Math.max(maxVal, a[i][j]);
        }
    }
    System.out.println(minVal + " " + maxVal);
}
```





4. Các bài toán cơ bản trên mảng 2 chiều:

b. Tính tổng từng hàng của mảng 2 chiều:

```
public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    int n = sc.nextInt(); //row
    int m = sc.nextInt(); //col
    int[][] a = new int[n][m];
    for(int i = 0; i < n; i++)
        for(int j = 0; j < m; j++)
            a[i][j] = sc.nextInt();
    for(int i = 0; i < n; i++){
        int rowSum = 0;
        for(int j = 0; j < m; j++){
            rowSum += a[i][j];
        }
        System.out.print(rowSum + " ");
    }
}
```

EXAMPLE





4. Các bài toán cơ bản trên mảng 2 chiều:

c. Tính tổng, hiệu hai ma trận:



Trong đại số tuyến tính, **ma trận tương tự như một mảng 2 chiều** gồm n hàng và m cột. Để 2 ma trận có thể cộng hoặc trừ cho nhau thì chúng phải **có cùng số hàng và số cột**.

1	2	0
0	4	1

+

1	4	8
9	2	3

=

2	6	8
9	6	4

1	2	0
0	4	1

-

1	4	8
9	2	3

=

0	-2	-8
-9	2	-2





4. Các bài toán cơ bản trên mảng 2 chiều:

c. Tính tổng, hiệu hai ma trận:

```
public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    int n = sc.nextInt(); //row
    int m = sc.nextInt(); //col
    int[][] a = new int[n][m];
    int[][] b = new int[n][m];
    for(int i = 0; i < n; i++)
        for(int j = 0; j < m; j++)
            a[i][j] = sc.nextInt();
    for(int i = 0; i < n; i++)
        for(int j = 0; j < m; j++)
            b[i][j] = sc.nextInt();
    for(int i = 0; i < n; i++){
        for(int j = 0; j < m; j++){
            System.out.print(a[i][j] + b[i][j] + " ");
        }
        System.out.println("");
    }
}
```

EXAMPLE





4. Các bài toán cơ bản trên mảng 2 chiều:

d. Nhân hai ma trận:



Giả sử có 2 ma trận a cỡ $n \times m$, ma trận b cỡ $p \times q$, để ma trận a có thể nhân với ma trận b thì số cột của ma trận a, tức là m phải bằng số hàng của ma trận b, tức là p.

$$a[n][m] \times b[p][q] = c[n][q]$$



Khi đó $m = p$ thì ma trận tích của a với b sẽ là ma trận c có cỡ $n \times q$. Phần tử ở chỉ số (i, j) của ma trận tích c được tính bằng cách nhân từng cặp phần tử ở hàng i của ma trận a với các phần tử ở cột j của ma trận b.





4. Các bài toán cơ bản trên mảng 2 chiều:

d. Nhân hai ma trận:

Nhập hai ma trận:

```
Scanner sc = new Scanner(System.in);
int n = sc.nextInt(); //row
int m = sc.nextInt(); //col
int[][] a = new int[n][m];
int[][] b = new int[n][m];
for(int i = 0; i < n; i++)
    for(int j = 0; j < m; j++)
        a[i][j] = sc.nextInt();
for(int i = 0; i < m; i++)
    for(int j = 0; j < p; j++)
        b[i][j] = sc.nextInt();
```

Nhân và in ra kết quả:

```
int[][] c = new int[n][p];
for (int i = 0; i < n; i++) {
    for (int j = 0; j < p; j++) {
        c[i][j] = 0;
        for (int k = 0; k < m; k++) {
            c[i][j] += a[i][k] * b[k][j];
        }
    }
}
```



5. Kỹ thuật duyệt các ô liền kề:

$i-1, j-1$	$i-1, j$	$i-1, j+1$
$i, j-1$	i, j	$i, j+1$
$i+1, j-1$	$i+1, j$	$i+1, j+1$



5. Kỹ thuật duyệt các ô liền kề:

Duyệt 4 ô chung cạnh với ô (i,j)

EXAMPLE

```
public static void main(String[] args) {  
    Scanner sc = new Scanner(System.in);  
    int[] dx = {-1, 0, 0, 1};  
    int[] dy = {0, -1, 1, 0};  
    int[][] a = {  
        {1, 2, 3},  
        {4, 5, 6},  
        {7, 8, 9}  
    };  
    int i = 1, j = 1;  
    for (int k = 0; k < 4; k++) {  
        int i1 = i + dx[k], j1 = j + dy[k];  
        System.out.print(a[i1][j1] + " ");  
    }  
}
```

OUTPUT

2 4 6 8

1	2	3
4	5	6
7	8	9





5. Kỹ thuật duyệt các ô liền kề:

Duyệt 8 ô chung đỉnh với ô (i,j)

EXAMPLE

```
public static void main(String[] args) {  
    Scanner sc = new Scanner(System.in);  
    int[] dx = {-1, -1, -1, 0, 0, 1, 1, 1};  
    int[] dy = {-1, 0, 1, -1, 1, -1, 0, 1};  
    int[][] a = {  
        {1, 2, 3},  
        {4, 5, 6},  
        {7, 8, 9}  
    };  
    int i = 1, j = 1;  
    for (int k = 0; k < 8; k++) {  
        int i1 = i + dx[k], j1 = j + dy[k];  
        System.out.print(a[i1][j1] + " ");  
    }  
}
```

OUTPUT

1 2 3 4 6 7 8 9

1	2	3
4	5	6
7	8	9





5. Kỹ thuật duyệt các ô liền kề:

Duyệt 8 ô xung quanh nước đi của quân mã

EXAMPLE

```
public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    int[] dx = {-2, -2, -1, -1, +1, +1, +2, +2};
    int[] dy = {-1, +1, -2, +2, -2, +2, -1, +1};
    int[][] a = {
        {1, 2, 3, 4, 5, 6},
        {7, 8, 9, 1, 2, 5},
        {1, 2, 1, 0, 3, 5},
        {1, 2, 1, 3, 4, 9},
        {1, 2, 1, 3, 0, 4},
        {1, 8, 7, 6, 2, 9}
    };
    int i = 2, j = 3;
    for (int k = 0; k < 8; k++) {
        int i1 = i + dx[k], j1 = j + dy[k];
        System.out.print(a[i1][j1] + " ");
    }
}
```

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

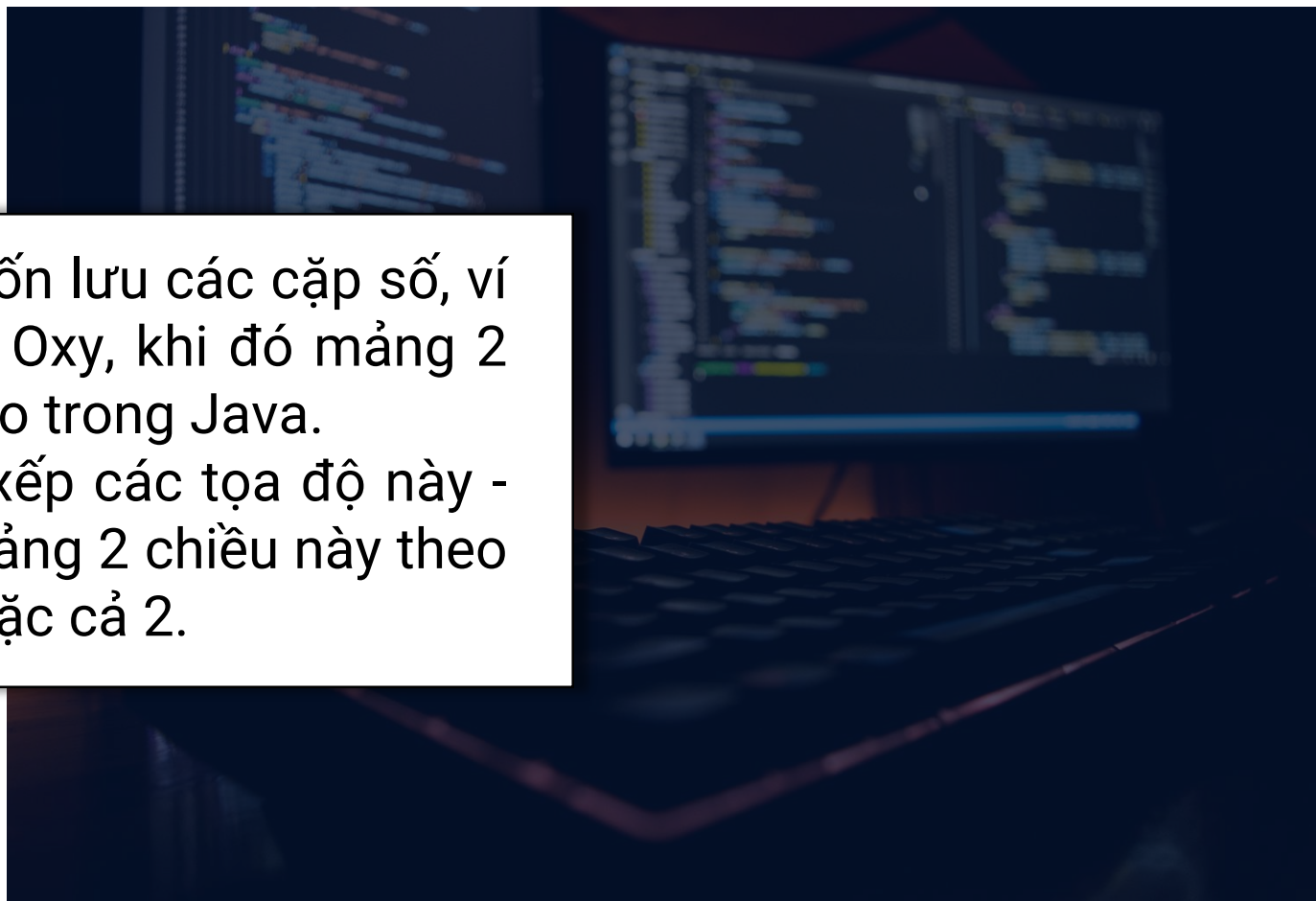
OUTPUT

3 5 8 5 2 9 1 0



6. Sắp xếp mảng hai chiều theo hàng:

- Giả sử bạn gặp 1 bài toán muốn lưu các cặp số, ví dụ tọa độ một điểm trong hệ Oxy, khi đó mảng 2 chiều là một lựa chọn hoàn hảo trong Java.
- Tuy nhiên đôi khi ta cần sắp xếp các tọa độ này - cũng chính là các hàng của mảng 2 chiều này theo phần tử thứ nhất hay thứ 2, hoặc cả 2.





6. Sắp xếp mảng hai chiều theo hàng:

EXAMPLE

```
public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    int n = sc.nextInt();
    Integer[][] a = new Integer[n][2];
    for (int i = 0; i < n; i++) {
        a[i][0] = sc.nextInt(); // hoành độ
        a[i][1] = sc.nextInt(); // tung độ
    }
    Arrays.sort(a, new Comparator<Integer[]>() {
        @Override
        public int compare(Integer[] o1, Integer[] o2) {
            if(o1[0] != o2[0]){
                return o1[0] - o2[0];
            }
            else{
                return o1[1] - o2[1];
            }
        }
    });
}
```

Ví dụ: Cho 1 danh sách các tọa độ Oxy, sắp xếp các tọa độ này theo chiều tăng dần của x, nếu 2 điểm có cùng hoành độ x thì sắp xếp theo tung độ tăng dần. ●

