**Mục đích sử dụng**

* Mục đích sử dụng của mảng nhiều chiều là để lưu trữ các dữ liệu dạng bảng.
* Phổ biến nhất của mảng nhiều chiều là mảng 2 chiều.
* Ví dụ lưu trữ hình ảnh, bảng tính xcel, bàn cờ, tọa độ hai chiều, thông tin bảng điểm, thông tin nhân viên…

**Cú pháp khai báo tổng quát**

* Cú pháp khai báo mảng hai chiều tổng quát là: **type[][] name;**
* Trong đó:
  + Phần type là kiểu của mảng. Có thể là bất kì kiểu dữ liệu hợp lệ nào trong Java. Ví dụ như int, float, boolean, String…
  + Sau phần kiểu mảng là hai cặp móc vuông [][] đi liền nhau. Đây là dấu hiệu nhận biết mảng hai chiều.
  + Sau đó là dấu cách và phần name. Name là tên mảng. Đặt theo quy tắc đặt tên mảng 1 chiều.
  + Kết thúc khai báo là dấu chấm phẩy.
* Mọi mảng đều phải được cấp phát hoặc khởi tạo trước khi có thể sử dụng.
* Ví dụ:

int[][] table; *// lưu trữ một bảng các số nguyên*

double[][] ballances; *// lưu trữ số dư của từng tháng tương ứng*

int[][] matrix; *// lưu trữ một ma trận*

float[][] coordinates; *// mảng lưu trữ tọa độ 2 chiều*

**Khởi tạo mảng**

* Ta có thể khởi tạo mảng theo nhiều cách khác nhau:

*// cách 1:*

**type[][] name = new type[row][col];**

*// cách 2:*

**type[][] name = new type[][] {**

**{v[0, 0], v[0, 1], ..., v[0, col-1]},**

**...,**

**{v[row-1, 0], v[row-1, 1], ..., v[row-1, col-1]}**

**};**

*// cách 3:*

**type[][] name = {          {v[0, 0], v[0, 1], ..., v[0, col-1]},**

**...,**

**{v[row-1, 0], v[row-1, 1], ..., v[row-1, col-1]}**

**};**

* Trong đó:
  + Nếu khởi tạo theo cách 1, tất cả các phần tử của mảng sẽ được khởi tạo giá trị mặc định của kiểu. Các kiểu số là 0, kiểu boolean là false, kiểu tham chiếu là null. Tổng số phần tử trong mảng là row x col.
  + Nếu khởi tạo theo cách 2 hoặc 3 thì thường ta sẽ biết trước tập dữ liệu. Và tập dữ liệu tương đối nhỏ. Số cặp {} bên trong {} sẽ là số hàng của mảng. Số phần tử trong mỗi cặp {} bên trong là số phần tử trên từng hàng tương ứng, gọi là số cột.
  + Cách 3 là cú pháp rút gọn của cách 2. Tổng số phần tử khởi tạo là tổng số phần tử của mảng.
* Mảng sau khi cấp phát hoặc khởi tạo là không thể thay đổi số lượng phần tử.

**Truy xuất giá trị trong mảng**

* Để truy xuất một phần tử mảng, ta dùng tên mảng và chỉ số hàng, cột. Cú pháp là: name[row\_index][col\_index];
* Lưu ý rằng chỉ số hàng, cột theo quy tắc của mảng 1 chiều. Luôn bắt đầu từ 0 và kết thúc ở row-1 và col-1 tương ứng cho hàng, cột.
* Số hàng, số cột luôn là số nguyên dương.
* Nếu truy cập phần tử ngoài mảng thì sẽ bị ngoại lệ như mảng 1 chiều.
* Ví dụ:

int[][] matrix = new int[][]{

        {1, 2, 3},

        {4, 5, 6},

        {7, 8, 9}

};

System.*out*.println(matrix[0][0]); *// lấy giá trị phần tử đầu tiên*

System.*out*.println(matrix[2][2]); *// lấy giá trị phần tử cuối*

matrix[0][2] = 100; *// thay đổi giá trị hàng 0 cột 2*

matrix[3][3] = 200; *// xảy ra ngoại lệ vì chỉ số hàng cột chỉ đến 2, 2*

* Để truy xuất toàn mảng hoặc một phần của mảng, ta thường dùng vòng lặp for lồng nhau.
* Cú pháp lấy số hàng của mảng hai chiều: **name.length**
* Cú pháp lấy số phần tử trên hàng i: **name[i].length**
* Ví dụ:

int[][] matrix = new int[][]{

        {1, 2, 3},

        {4, 5, 6},

        {7, 8, 9}

};

*// truy xuất từng phần tử mảng, trong đó matrix.length là số hàng*

*// matrix[i].length là số cột trên từng hàng*

for (int i = 0; i < matrix.length; i++) {

    for (int j = 0; j < matrix[i].length; j++) {

        System.*out*.print(matrix[i][j] + " ");

    }

    System.*out*.println(); *// in xuống dòng*

}

Kết quả:

1 2 3

4 5 6

7 8 9

* Nếu dùng foreach thì phải duyệt toàn bộ mảng. Cú pháp sẽ khác đôi chút với foreach ở mảng 1 chiều.
* Ví dụ:

int[][] matrix = new int[][]{

        {1, 2, 3},

        {4, 5, 6},

        {7, 8, 9}

};

for (var row : matrix) { *// duyệt từng hàng trong ma trận*

for (var element : row) { *// duyệt từng phần tử trên hàng*

System.*out*.print(element + " ");

    }

    System.*out*.println(); *// in xuống dòng*

}

Kết quả:

1 2 3

4 5 6

7 8 9

* Cú pháp tương tự áp dụng cho các mảng hai chiều khác.

**Mảng zigzag**

* Mảng từ đầu tới giờ ta xét là mảng có số phần tử trên từng hàng là như nhau.
* Trong Java cho phép tạo mảng hai chiều mà số cột trên từng hàng là khác nhau. Tức là số phần tử trên từng hàng khác nhau.
* Thực chất Java chỉ có mảng 1 chiều. Mảng nhiều chiều là mảng được tạo ra từ mảng của mảng. Tức là mỗi hàng trong mảng hai chiều có bản chất là một mảng 1 chiều.
* Để tạo mảng zigzag ta thực hiện qua hai bước:
  + Cấp phát số hàng
  + Cấp phát số phần tử trên từng hàng(số cột của từng hàng)

Ví dụ:

type[][] name = new type[row][]; *// cấp phát số hàng*

for (int i = 0; i < row; i++) { *// cấp phát số p.tử*

name[i] = new type[col\_i];  *// cho từng hàng*

}

* Sau đó sử dụng mảng như bình thường:

*// giả sử tạo mảng có số phần tử trên từng hàng tăng dần*

int[][] triangle = new int[4][];

for (int i = 0; i < triangle.length; i++) {

    triangle[i] = new int[i + 1];

}

*// gán các giá trị cho mảng:*

for (int i = 0; i < triangle.length; i++) {

    for (int j = 0; j < triangle[i].length; j++) {

        triangle[i][j] = i + j;

    }

}

*// hiện giá trị từng phần tử mảng zigzag*

for (var row : triangle) { *// duyệt từng hàng trong ma trận*

for (var element : row) { *// duyệt từng phần tử trên hàng*

System.*out*.print(element + " ");

    }

    System.*out*.println(); *// in xuống dòng*

}

Kết quả:

0

1 2

2 3 4

3 4 5 6

**Ví dụ minh họa**

Ví dụ lưu trữ hình nhật đặc kích thướng m x n bằng các dấu \* sau đó hiển thị ra màn hình.

import java.util.Scanner;

public class Example {

    public static void main(String[] args) {

        var input = new Scanner(System.*in*);

        System.*out*.println("Nhập vào chiều cao, chiều rộng hình CN: ");

        var height = input.nextInt();

        var width = input.nextInt();

*// cấp phát mảng với số hàng và số cột cho trước*

String[][] rectangle = new String[height][width];

*// gán các giá trị cho mảng:*

*fillRect*(rectangle);

*// vẽ hình chữ nhật ra màn hình*

*drawRect*(rectangle);

    }

*/\*\**

*\* phương thức vẽ các dấu \* và xuống dòng ra màn hình*

*\**

*\* @param rect mảng chứa các phần tử cần vẽ ra*

*\*/*

private static void drawRect(String[][] rect) {

        for (int i = 0; i < rect.length; i++) {

            for (int j = 0; j < rect[i].length; j++) {

                System.*out*.print(rect[i][j]);

            }

            System.*out*.println(); *// in xuống dòng*

}

    }

*/\*\**

*\* phương thức gán giá trị cho từng phần tử của mảng*

*\* hai chiều lưu trữ hình chữ nhật bằng các dấu \**

*\**

*\* @param rect mảng chứa giá trị các dấu \**

*\*/*

private static void fillRect(String[][] rect) {

        for (int i = 0; i < rect.length; i++) {

            for (int j = 0; j < rect[i].length; j++) {

                rect[i][j] = " \* ";

            }

        }

    }

}

Nhập vào chiều cao, chiều rộng hình CN:

4 5

\*  \*  \*  \*  \*

\*  \*  \*  \*  \*

\*  \*  \*  \*  \*

\*  \*  \*  \*  \*