


Bài 26: Mảng hai chiều

- ✓ Mục đích sử dụng
- ✓ Cú pháp tổng quát
- ✓ Truy xuất giá trị mảng
- ✓ Mảng zigzag
- ✓ Ví dụ minh họa
- ✓ Bài tập thực hành



Mục đích sử dụng

- 
- Để lưu trữ các dữ liệu có dạng bảng gồm các hàng và các cột
 - Phổ biến nhất và là nội dung của bài học này là mảng hai chiều
 - Ví dụ ứng dụng mảng hai chiều để: lưu trữ ảnh, bảng tính excel, bàn cờ, tọa độ, bảng điểm, thông tin nhân viên...

Cú pháp tổng quát

- Cú pháp: `type[][] name;`
- Trong đó:
 - Type là kiểu của mảng, có thể là bất kì kiểu hợp lệ nào trong Java
 - Sau đó là hai cặp móc vuông `[]` liền nhau để nhận diện khai báo mảng hai chiều
 - Tiếp đó là dấu cách và name là tên mảng hai chiều, đặt theo quy tắc đặt tên cho mảng
 - Kết thúc khai báo là dấu chấm phẩy
- Mọi mảng trong Java phải được cấp phát hoặc khởi tạo trước khi có thể sử dụng

Ví dụ về khai báo mảng

- Sau đây là một số ví dụ về khai báo mảng hai chiều trong Java:

```
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 hai chiều

➤ Có nhiều cách để khởi tạo mảng hai chiều:

// 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]}  
};
```

Khởi tạo mảng hai chiều

- Với cách 1: Tất cả các phần tử của mảng sẽ được khởi tạo giá trị mặc định của kiểu mảng. Các kiểu số là 0, kiểu boolean là false, kiểu tham chiếu là null
- Tổng số phần tử được cấp phát cho mảng là $\text{row} * \text{col}$ phần tử. Row và col phải nguyên dương
- Với cách 2 và 3: thường ta biết trước tập dữ liệu và mảng sẽ tương đối nhỏ. Số cặp {} phân tách nhau bởi dấu phẩy là số hàng, số phần tử trong mỗi cặp {} bên trong là số phần tử của hàng tương ứng
- Cách 3 là rút gọn của cách 2.
- Số lượng phần tử của mảng là tổng số phần tử đã khởi tạo

Lưu ý

- Kích thước mảng sau khi khởi tạo là không thể thay đổi
- Các phần tử trong mảng thường cùng kiểu
- Không khai báo kiểu mảng hoặc khởi tạo mảng với từ khóa var

Truy xuất mảng

- Để truy xuất đến 1 phần tử trong mảng ta dùng tên mảng và chỉ số hàng, cột đặt trong hai móc vuông: `name[row_index][col_index]`
- Quy tắc của chỉ số giống như mảng 1 chiều: bắt đầu từ 0 và kết thúc ở row-1, col-1
- Số hàng và số cột luôn là số nguyên dương
- Nếu truy cập chỉ số ngoài biên mảng sẽ gặp ngoại lệ
- Để lấy số hàng của mảng: `name.length`
- Để lấy số cột của hàng i: `name[i].length`

Ví dụ truy xuất mảng

➤ Sau đây là ví dụ về truy cập và sử dụng phần tử mảng:

```
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
```

Truy xuất mảng

- Thường sử dụng vòng lặp for lồng nhau để truy xuất một đoạn hoặc toàn mảng:

```
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
}
```

Truy xuất mảng

➤ Nếu dùng foreach thì cú pháp như sau:

```
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  
}
```

Mảng zigzag

- Thực chất Java không có mảng nhiều chiều, mảng 2 chiều ở đây là mảng của các mảng 1 chiều
- Mỗi hàng trong mảng 2 chiều là một mảng 1 chiều
- Java cho phép ta tạo mảng hai chiều trong đó mỗi hàng có số lượng phần tử khác nhau, gọi mảng zigzag
- Quá trình tạo mảng zigzag:
 - Bước 1: cấp phát số hàng
 - Bước 2: cấp phát số phần tử trên từng hàng

Mảng zigzag

➤ Ví dụ:

```
// 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
}
```

Ví dụ minh họa

- Ví dụ 1: Vẽ hình chữ nhật đặc bằng các dấu * sau đó lưu vào mảng 2 chiều. Hiện kết quả lên màn hình.

Nhập vào chiều cao, chiều rộng hình CN:

4 5

```
* * * * *  
* * * * *  
* * * * *  
* * * * *
```

- Ví dụ 2: Tạo mảng zigzag lưu tam giác số vuông góc trái dưới dạng:

0

1 2

2 3 4

3 4 5 6

Nội dung tiếp theo

Tổng quan về lớp và đối tượng