Bài 1.1: Mảng một chiều

- ✓ Định nghĩa
- ✓ Cú pháp tổng quát
- ✓ Truy xuất giá trị trong mảng
- ✓ Vòng lặp foreach
- ✓ Ví dụ minh họa
- ✓ Bài tập thực hành

Khái niệm

- Mảng là một cấu trúc dữ liệu dùng để lưu trữ tập hợp của các giá trị cùng kiểu
- Các giá trị đơn lẻ cấu thành mảng gọi là phần tử của mảng
- Mỗi phần tử được truy xuất thông qua chỉ số mảng. Chỉ số này là duy nhất
- Chỉ số mảng là số nguyên. Chỉ số phần tử đầu tiên là 0, phần tử cuối cùng là n-1 với n là số phần tử của mảng
- Ví dụ: dùng mảng lưu danh sách sinh viên, danh sách môn học, danh sách các tỉ phú thế giới, danh sách người yêu cũ…

Cú pháp khai báo tổng quát

- Cú pháp: *type[] name;*
- Trong đó:
  - Phần type là kiểu của mảng, có thể là bất kì kiểu dữ liệu hợp lệ nào trong Java: kiểu nguyên thủy, tham chiếu
  - Sau kiểu là cặp móc vuông []. Đây là cú pháp nhận diện 1 biến kiểu mảng so với một biến thông thường. Bạn có thể đặt cặp [] sau tên mảng tuy nhiên khuyến nghị đặt sau kiểu để tránh nhầm lần.
  - Sau móc vuông là dấu cách rồi đến phần name là tên mảng. Quy tắt đặt như đặt tên biến nhưng ở số nhiều.
  - Kết thúc khai báo là dấu chấm phẩy

Ví dụ

```
int[] myNumbers; // khai báo biến mảng kiểu int
float[] avgGrades; // khai báo mảng kiểu float
String[] brands; // khai báo mảng kiểu String
String[] students; // khai báo mảng lưu danh sách các sinh viên
String[] friends; // khai báo mảng lưu danh sách những người bạn
```

Khởi tạo mảng

- Mọi mảng đều phải khởi tạo hoặc gán giá trị trước khi chúng có thể được sử dụng
- Cú pháp khởi tạo mảng:

```
type[] name = new type[num_of_elements];
type[] name = new type[]{elements};
type[] name = {elements};
```

- Trong đó cú pháp giống với khai báo mảng và cộng thêm:
  - Phép gán = và cấp phát new … hoặc cú pháp khởi tạo rút gọn {…};
  - Phần bên trong ngoặc vuông, ở vế phải là số lượng phần tử của mảng sẽ lưu trữ. Giá trị này luôn nguyên dương
  - Kết thúc cú pháp luôn có dấu chấm phẩy

Khởi tạo mảng

```
type[] name = new type[num_of_elements];
type[] name = new type[]{elements};
type[] name = {elements};
```

- Với cách 1: tạo ra một mảng kiểu type có num_of_elements phần tử. Tất cả các phần tử mặc định sẽ là 0 nếu là kiểu số, false nếu là boolean, null nếu là các kiểu tham chiếu
- Với cách 2: tạo ra mảng kiểu type và khởi tạo các phần tử trong cặp ngoặc {}. Số lượng phần tử tương ứng bằng số lượng phần tử trong cặp {}
- Với cách 3: là rút gọn của cách 2. Lúc này số lượng phần tử liệt kê trong {} sẽ là số phần tử của mảng

Khởi tạo mảng

```
type[] name = new type[num_of_elements];
type[] name = new type[]{elements};
type[] name = {elements};
```

- Khi khởi tạo theo cách 2, 3 thì lưu ý rằng tất cả các phần tử trong cặp {} phải cùng kiểu hoặc có kiểu nhỏ hơn kiểu của mảng
- Ví dụ:

```
int[] myNumbers = new int[10] {1, 2, 3, 5}; // error
float[] avgGrades = {1.25f, 3.24, 5.66f, 7.49}; // error
String[] brands = new String[]{"Apple", "Samsung", "Huawei"}; // ok
String[] students = {"Loan", "Hoa", "Nhung", "Phương", "Thanh"}; // ok
```

Truy xuất giá trị trong mảng

- Để sử dụng các phần tử mảng, ta cần có tên mảng và chỉ số của phần tử mảng
- Cú pháp gán giá trị cho 1 phần tử:
  name[index] = value;
- Cú pháp gán phần tử mảng cho một biến:
  variable = name[index];
- Trong đó:
  - Phần name là tên mảng
  - Phần index là chỉ số phần tử mảng [0, n-1]
  - Value là giá trị cùng kiểu với kiểu mảng hoặc kiểu nhỏ hơn
  - Variable là tên biến cùng kiểu hoặc kiểu lớn hơn nào đó

Truy xuất giá trị trong mảng

- Để lấy số phần tử của mảng, cú pháp là: name.length
- Ta có thể gán một biến mảng cho một biến mảng khác cùng kiểu. Khi đó chúng cùng tham chiếu đến 1 mảng
- Kiểu mảng là kiểu đối tượng.

```
String[] brands = new String[]{"Apple", "Samsung", "Huawei", "Oppo"};
int size = brands.length; // lấy số phần tử mảng
System.out.println("Số phần tử của mảng là: " + size);
// gán giá trị cho một phần tử mảng:
brands[2] = "Xiaomi";
System.out.println("Phần tử tại chỉ số 2 của mảng là: " + brands[2]);
// gán mảng cho một biến mảng khác cùng kiểu
String[] other = brands; // gán mảng brands cho mảng khác cùng kiểu String
System.out.println(brands[0]); // hiển thị giá trị phần tử đầu tiên trong brands
System.out.println(other[0]); // hiển thị giá trị phần tử đầu tiên trong other
// kết quả cùng là chuỗi kí tự "Apple"
```

Copy mảng

- Để copy mảng sang một mảng khác ta dùng Arrays.copyOf(source, new_size);
- Trong đó:
  - Tham số thứ nhất là mảng nguồn cần copy
  - Tham số thứ hai là kích thước mảng đích
  - Trả về một mảng mới cùng kiểu với mảng nguồn
- Ví dụ:

```
String[] brands = new String[]{"Apple", "Samsung", "Huawei", "Oppo"};
var copyArr = Arrays.copyOf(brands, 2);
System.out.println(copyArr[1]); // kết quả in ra: Samsung
```

Sử dụng vòng lặp for

- Ta thường kết hợp việc truy xuất cả một mảng hoặc một đoạn phần tử trong mảng bằng việc sử dụng vòng lặp for
- Ví dụ:

```
String[] brands = new String[]{"Apple", "Samsung", "Huawei", "Oppo"};
// duyệt toàn bộ mảng
for (int i = 0; i < brands.length; i++) {
    System.out.println(brands[i]);
}
```

- Cần lưu ý rằng nếu bạn truy cập phần tử mảng với chỉ số ngoài đoạn [0, n-1] thì sẽ nhận được ngoại lệ *ArrayIndexOutOfBoundsException*

Vòng lặp foreach

- Cú pháp:

```
for (type element : collection) {
    // do something
}
```

- Trong đó:
  - Type là kiểu của biến, có thể là bất kì kiểu hợp lệ nào
  - Element là tên đại diện cho 1 phần tử của mảng
  - Dấu hai chấm thể hiện ý nghĩa: là một phần tử thuộc collection
  - Collection là tên của tập hợp, mảng
  - Phần thân của foreach chứa các câu lệnh cần thực hiện bên trong cặp {}
- Chủ yếu sử dụng để duyệt mảng(đọc dữ liệu)

Ví dụ vòng lặp foreach

```java
String[] brands = new String[]{"Apple", "Samsung", "Huawei", "Oppo"};
// duyệt toàn bộ mảng
for (int i = 0; i < brands.length; i++) {
    System.out.println(brands[i]); // hiển thị p.tử thứ i
}
// tương đương:
for (var item : brands) { // sử dụng kiểu tự suy luận var
    System.out.println(item);
}
// hoặc:
for (String item : brands) { // sử dụng kiểu tường minh: String
    System.out.println(item);
}
```

Nội dung tiếp theo

Mảng 2 chiều