Bài 12: MẢNG TRONG JAVA

Xem bài học trên website để ủng hộ Kteam: Mảng trong Java

Mọi vấn đề về lỗi website làm ảnh hưởng đến bạn hoặc thắc mắc, mong muốn khóa học mới, nhằm hỗ trợ cải thiện Website. Các bạn vui lòng phản hồi đến Fanpage How Kteam nhé!

Dẫn nhập

Ở những bài về BIẾN và KIỂU DỮ LIỆU ta đã học cách lưu trữ các giá trị vào những biến đơn lẻ. Tuy nhiên, có những lúc ta muốn lưu nhiều giá trị chung kiểu dữ liệu vào một biến nhất định. Ta gọi đó làm mảng, trong bài này Kteam sẽ giải thích **công dụng về mảng**.

Nội dung

Để đọc hiểu bài này, tốt nhất các bạn nên có kiến thức cơ bản về các phần sau:

- CÁC BIẾN TRONG JAVA.
- CÁC KIỂU DỮ LIỆU TRONG JAVA.
- CÁC HẠNG TOÁN TỬ TRONG JAVA
- CẤU TRÚC RẼ NHÁNH TRONG JAVA
- VÒNG LẶP WHILE TRONG JAVA
- VÒNG LẶP FOR TRONG JAVA

Bài này chúng ta sẽ tìm hiểu những vấn đề sau:

- Mảng là gì? Ưu nhược của mảng
- Cấu trúc của mảng

Mảng là gì? Ưu nhược của mảng

Mảng là gì?

Mảng là tập hợp các đối tượng có cùng kiểu dữ liệu và được lưu trữ gần nhau trong bộ nhớ. Mỗi đối tượng hay được gọi là phần tử, các phần từ được phân biệt bằng vị trí (hay chỉ số phần tử), được bắt đầu từ vị trí 0.

Việc sử dụng **mảng** mang ý nghĩa lưu những giá trị liên quan với nhau. Ví dụ như lưu điểm kiểm tra của 30 học sinh trong lớp, như vậy ta muốn lấy điểm của học sinh nào đó thì chỉ cần viết vị trí của học sinh đó trong danh sách.

Ưu nhược của mảng

Ưu điểm:

- Tối ưu code: Gom các phần tử liên quan vào chung một với nhau giúp code gọn gàng hơn.
- Có thể truy cập ngẫu nhiên: Do các vị trí ô lưu trữ liên tiếp ta có thể truy cập ngẫu nhiên bằng chỉ số phần tử dễ dàng và nhanh chóng.
- Dễ thao tác, quản lý và nâng cấp: Như muốn thay đổi các giá trị theo 1 quy luật thì ta sẽ tận dụng sử dụng những vòng lặp lập trình.

Nhược điểm:

- Giới hạn kích thước: Khi sử dụng mảng ta phải khai báo kích thước lưu trữ của mảng và không thể thay đổi kích thước trong lúc chạy.
- Vùng lưu trữ phải liên tiếp: Đây cũng là vừa ưu vừa nhược điểm. Vì yêu cầu các ô nhớ liên tiếp nên phải tốn không gian bộ nhớ, hoặc đủ ô nhớ nhớ nhưng các ô nhớ không tiếp nên không thể khai báo được.

Cấu trúc của mảng

Mảng có hai loại: Mảng một chiều và mảng đa chiều

Mảng 1 chiều

Cú pháp khai báo:

```
<kiểu dữ liệu> [] <tên mảng>;
```

Cú pháp cấp phát bộ nhớ để tạo mảng:

```
<tên mảng> = new <kiểu dữ liệu>[kích cỡ mảng];
```

Cú pháp rút gọn hơn:

```
<kiểu dữ liệu> [] <tên mảng> = new <kiểu dữ liệu>[kích cỡ mảng];
```

Ví dụ: Khai báo mảng có 3 phần tử, đưa giá trị. Thử in mảng a và các giá trị mảng a

```
public class HelloWorld{

    public static void main(String []args){
      int[] a;
```

```
        a = new int[3];
        a[0] = 5;
        a[1] = 2;
        a[2] = 1;
        System.out.println(a);
        for (int i=0; i<a.length; i++){
            System.out.println(a[i]);
        }
    }
}
```

```
$javac HelloWorld.java
$java  HelloWorld
[I@2a139a55
5
2
1
```

Ở ví dụ trên. Kteam sử dụng **a.length** là một thuộc tính của mảng giúp ta có thể biết kích cỡ của mảng, nó giúp ta sử dụng vòng lặp for.

Ta thấy khi ta thử in ra mảng a thì nó cho kết quả **[I@2a139a55**. Vì mảng không phải kiểu dữ liệu nguyên thủy (ở bài kiểu dữ liệu Kteam đã giải thích), mảng là kiểu dữ liệu tham chiếu. Ý nghĩa kết quả như sau: **[I** có nghĩa đây là mảng thuộc kiểu int (tượng trưng bằng chữ **I**), **@2a139a55** thì tùy thuộc JVM đưa ra, nhưng thường là địa chỉ lưu trữ đối tượng.

Cú pháp khởi tạo cho mảng:

> **<kiểu dữ liệu> [] <tên mảng> = {<giá trị>,...}**

Ví dụ:

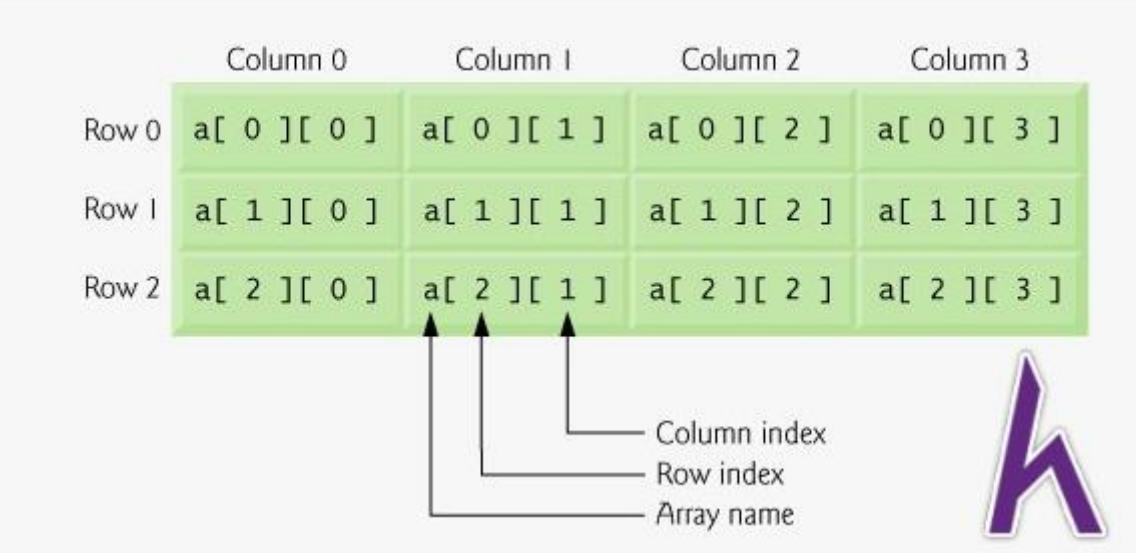
```
public class HelloWorld{

    public static void main(String []args){
      char[] a = {'H', 'o', 'w', 'K','t','e','a','m'};
      System.out.print(a);
    }
}
```

```
$javac HelloWorld.java
$java  HelloWorld
HowKteam
```

Riêng kiểu dữ liệu **char** đặc biệt hơn là có thể in ra toàn bộ giá trị bằng cách print trực tiếp mảng.

Mảng đa chiều

Cũng như mảng một chiều dùng để lưu các giá trị có nét tương đồng. Thì mảng đa chiều chỉ là **tăng số chiều lưu trữ nhiều chiều hơn**, hay còn gọi là ma trận. Thường thường ta hay sử dụng mảng 2 chiều. Kteam sẽ nói 2 chiều là chính cho các bạn dễ hình dung. Trong ma trận 2 chiều, ta hay gọi chiều thứ 1 là hàng, còn chiều thứ 2 là cột.



Cú pháp khai báo:

> **<kiểu dữ liệu> [][] <tên mảng> = new <kiểu dữ liệu>[kích cỡ hàng] [kích cỡ cột];**

Cú pháp khởi tạo cho mảng:

> **<kiểu dữ liệu> [][] <tên mảng> = {{các giá trị hàng 1}, {các giá trị hàng 2},... {các giá trị hàng n}}**

Ví dụ: Tạo ma trận 3 hàng 2 cột và in các giá trị ra theo hàng và cột.

```
public class HelloWorld{

    public static void main(String []args){
      int[][] a= {{1,2,3},{4,5,6},{7,8,9}};
      for (int i=0; i<3; i++){
         for(int j=0; j<3; j++){
            System.out.print(a[i][j]+" ");
         }
         System.out.println();
      }
    }
}
```

```
$javac HelloWorld.java
$java  HelloWorld
 1 2 3
 4 5 6
 7 8 9
```

Lưu ý

Thực ra trong Java có thể khai báo mảng theo 2 cách sau: **int[] a** hoặc **int a[]**. Chức năng đều như nhau, nên các bạn có thể cách nào tùy ý. Tuy nhiên, do

người ta thường khuyên viết cách đầu hơn nên Kteam quyết định viết bài theo cách đó.

Kết

Như vậy chúng ta đã tìm hiểu mảng trong Java

Ở bài sau, Kteam sẽ giới thiệu đến bạn về VÒNG LẶP FOR-EACH JAVA

Cảm ơn các bạn đã theo dõi bài viết. Hãy để lại bình luận hoặc góp ý của mình để phát triển bài viết tốt hơn. Đừng quên "**Luyện tập – Thử thách – Không ngại khó**".