



---

# Mảng

Khóa học: NỀN TẢNG LẬP TRÌNH

---

# Kiểm tra bài trước

Hỏi và trao đổi về các khó khăn gặp phải trong bài “Cấu trúc lặp”

Tóm tắt lại các phần đã học từ bài “Cấu trúc lặp”

# Mục tiêu

---

- Trình bày được khái niệm mảng
- Mô tả được cú pháp khai báo và sử dụng mảng
- Mô tả được cách sử dụng vòng lặp for để duyệt mảng
- Mô tả được cách sử dụng vòng lặp for/in để duyệt mảng
- Khai báo và sử dụng được mảng một chiều
- Trình bày được khái niệm mảng đa chiều
- Mô tả được cú pháp khai báo và sử dụng mảng hai chiều
- Sử dụng for để làm việc với mảng
- Sử dụng for-in để làm việc với mảng

---

# Thảo luận

Khai báo mảng

Khởi tạo mảng

Gán giá trị cho mảng

# Lưu trữ nhiều dữ liệu

---

- Trong các chương trình máy tính, có những trường hợp chúng ta phải lưu trữ rất nhiều dữ liệu
- Chẳng hạn, cần lưu trữ danh sách tên của hàng trăm sinh viên, cùng điểm thi của từng người
- Việc khai báo hàng trăm biến để lưu trữ các dữ liệu kiểu này rất mất thời gian và không thực tế
- JavaScript và phần lớn các ngôn ngữ khác hỗ trợ một cấu trúc dữ liệu là mảng để giải quyết các tình huống này



# Mảng là gì



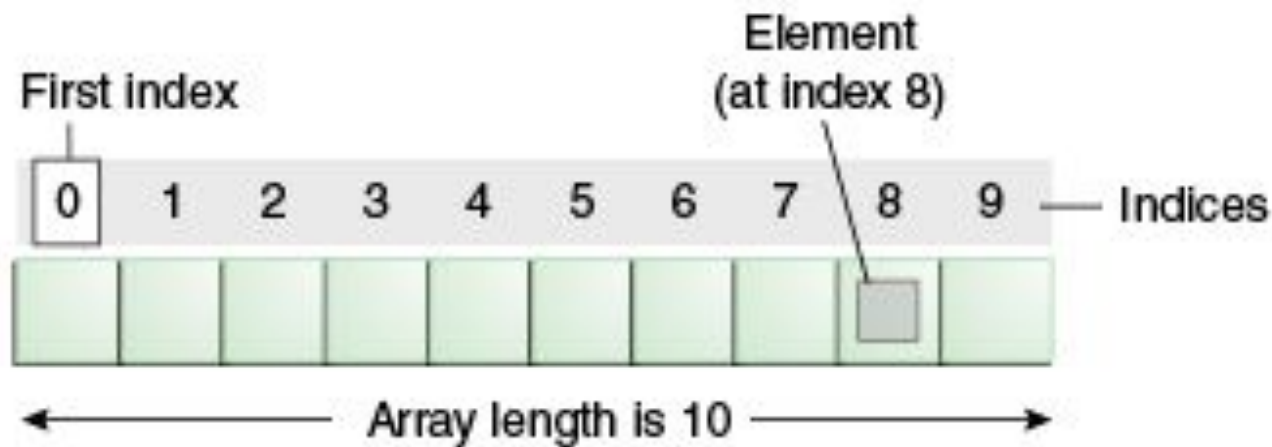
- Mảng là một loại biến đặc biệt, có thể lưu được nhiều giá trị thay vì chỉ một giá trị như các biến thông thường
- Mỗi giá trị trong mảng được gọi là một phần tử
- Các phần tử được lưu trữ ở các vị trí kế tiếp nhau trong bộ nhớ
- Chẳng hạn, mảng `numbers` chứa 12 phần tử là các số nguyên

```
numbers[0]  
numbers[1]  
numbers[2]  
numbers[3]  
numbers[4]  
numbers[5]  
numbers[6]  
numbers[7]  
numbers[8]  
numbers[9]  
numbers[10]  
numbers[11]
```

-45
6
0
72
1543
-89
0
62
-3
1
6453
78

# Các khái niệm của mảng

- Tên mảng: Tuân thủ theo quy tắc đặt tên của biến
- Phần tử: Các giá trị được lưu trữ trong mảng
- Chỉ số: Vị trí của các phần tử trong mảng. Chỉ số bắt đầu từ 0.
- Độ dài: Số lượng các phần tử của mảng



# Khai báo mảng: Cách 1

---



- Cú pháp:

```
let array_name = [item1, item2, ...];
```

- Ví dụ:

```
let array_name = ["Saab", "Volvo", "BMW"];
```





# Khai báo mảng: Cách 2

---

- Sử dụng từ khóa `new`
- Cú pháp:

```
let array_name = new Array(item1, item2, ...);
```

- Ví dụ:

```
let array_name = new Array("Saab", "Volvo", "BMW");
```



---

# Demo

Khai báo mảng sử dụng 2 cách

---

# Thảo luận

Khai báo mảng

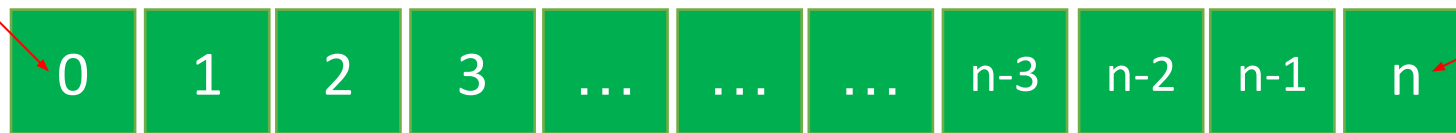
Khởi tạo mảng

Gán giá trị cho mảng

# Phần tử và chỉ số của mảng

- Mỗi phần tử (item) được xác định bằng một số thứ tự còn gọi là chỉ số (index) duy nhất trong mảng
- Chỉ số là một số nguyên dương
- Chỉ số của phần tử đầu tiên là 0
- Chỉ số của phần tử cuối cùng là  $n - 1$ , trong đó  $n$  là độ dài của mảng
- Có thể truy xuất đến phần tử của mảng thông qua chỉ số

Phần tử đầu  
tiên



Phần tử cuối  
cùng

←  $n$ : độ dài của mảng →



# Truy xuất phần tử trong mảng

---

- Mỗi phần tử trong mảng được thao tác giống như một biến
- Truy xuất các phần tử thông qua chỉ số đặt trong dấu []
- Ví dụ, gán giá trị cho một phần tử đầu tiên của mảng

```
cars[0] = "Opel";
```

- Ví dụ, lấy giá trị của một phần tử đầu tiên của mảng

```
let name = cars[0];
```

# Vòng lặp for-in

---

- Vòng lặp for-in (còn gọi là *enhanced for*) được sử dụng để duyệt qua các phần tử của một collection, chẳng hạn như mảng, danh sách...
- Cú pháp: 

```
for (let var_name in collection) {  
  
}
```

Trong đó:

- var\_name: Biến đại diện lần lượt cho từng phần tử của collection trong mỗi lần lặp
- collection: đối tượng cần lặp



# for-in: Ví dụ

---

- Duyệt qua các phần tử của một mảng:

```
let array = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10];  
  
for (let element in array) {  
    console.log("element = " + array[element]);  
}
```

- Tương đương với câu lệnh for:

```
for (let i = 0; i < numbers.length; i++) {  
    console.log("element = " + numbers[i]);  
}
```

# Duyệt các phần tử của mảng với for



- Sử dụng vòng lặp for để duyệt qua tất cả các phần tử của mảng

```
for (let i = 0; i < cars.length; i++) {  
    text += cars[i] + "<br>";  
}
```

BMW  
Volvo  
Saab  
Ford  
Fiat  
Audi





# Khác biệt khi sử dụng for và for-in

---

- Có thể sử dụng index khi duyệt bằng for, không có index nếu sử dụng for-in
- Có thể sử dụng for để duyệt theo các chiều khác nhau (từ đầu đến cuối, từ cuối đến đầu...)
- For in chỉ duyệt từ đầu đến cuối mảng

---

# Các thuộc tính và phương thức của mảng

# Độ dài của mảng

---



- Thuộc tính *length* là độ dài của mảng
- Ví dụ:

```
let fruits = ["Banana", "Orange", "Apple", "Mango"];  
alert(fruits.length);
```

Có giá trị là 4





# Thêm phần tử vào mảng: push()

---

- Phương thức push() cho phép thêm phần tử vào cuối mảng
- Ví dụ:

```
let fruits = ["Banana", "Orange", "Apple", "Mango"];  
fruits.push("Lemon");
```

Phần tử mới được thêm vào tại vị trí 5 (index 4)



# Thêm phần tử vào mảng: length

- Có thể thêm phần tử vào mảng bằng cách gán giá trị cho phần tử ở vị trí length
- Ví dụ:

```
let fruits = ["Banana", "Orange", "Apple", "Mango"];  
fruits[fruits.length] = "Lemon";
```

Độ dài 4



Phần tử mới được thêm vào tại vị trí 5 (index 4)



# join()



- Phương thức join() chuyển tất cả các phần tử trong mảng thành chuỗi và nối chúng lại với nhau
- Ví dụ:

```
let a = [1, 2, 3];           // Khởi tạo mảng với 3 phần tử
a.join();                   // => "1,2,3"
a.join(" ");                // => "1 2 3"
a.join("");                 // => "123"
let b = new Array(10);      // Tạo mảng có độ dài 10, và không có phần tử nào
b.join("-");                // => "-----" (một mảng có 9 kí tự -)
```



# reverse()

---

- Phương thức reverse() đảo ngược các phần tử trong mảng, trả về một mảng các phần tử bị đảo ngược
- Ví dụ:

```
let a = [1, 2, 3];  
a.reverse().join();    // => "3,2,1" và a có giá trị là [3,2,1]
```

# sort()

---



- Sắp xếp các phần tử của mảng sử dụng phương thức sort()
- Ví dụ 1:

```
let a = new Array("banana", "cherry", "apple");  
a.sort();  
let s = a.join(", "); // => "apple, banana, cherry"
```



# sort()

---



- Ví dụ 2:

```
let a = [33, 4, 1111, 222];  
a.sort();           // Alphabetical order: 1111, 222, 33, 4  
a.sort(function(a,b) { // Numerical order: 4, 33, 222, 1111  
    return a-b;      // Returns < 0, 0, or > 0, depending on order  
});  
a.sort(function(a,b) { // Reverse numerical order  
    return b-a;  
});
```



# shift() và unshift()

- Phương thức shift() để xóa phần tử vào đầu mảng
- Phương thức unshift() để nối phần tử đầu tiên của mảng
- Ví dụ:

```
let a = [];           // a: []
a.unshift(1);          // a: [1]
a.unshift(22);         // a: [22, 1]
a.shift();             // a: [1]
a.unshift(3, [4, 5]);  // a: [3, [4, 5], 1]
a.shift();             // a: [[4, 5], 1]
a.shift();             // a: [1]
a.shift();             // a: []
```



---

# Demo

Một số phương thức thao tác với mảng



---

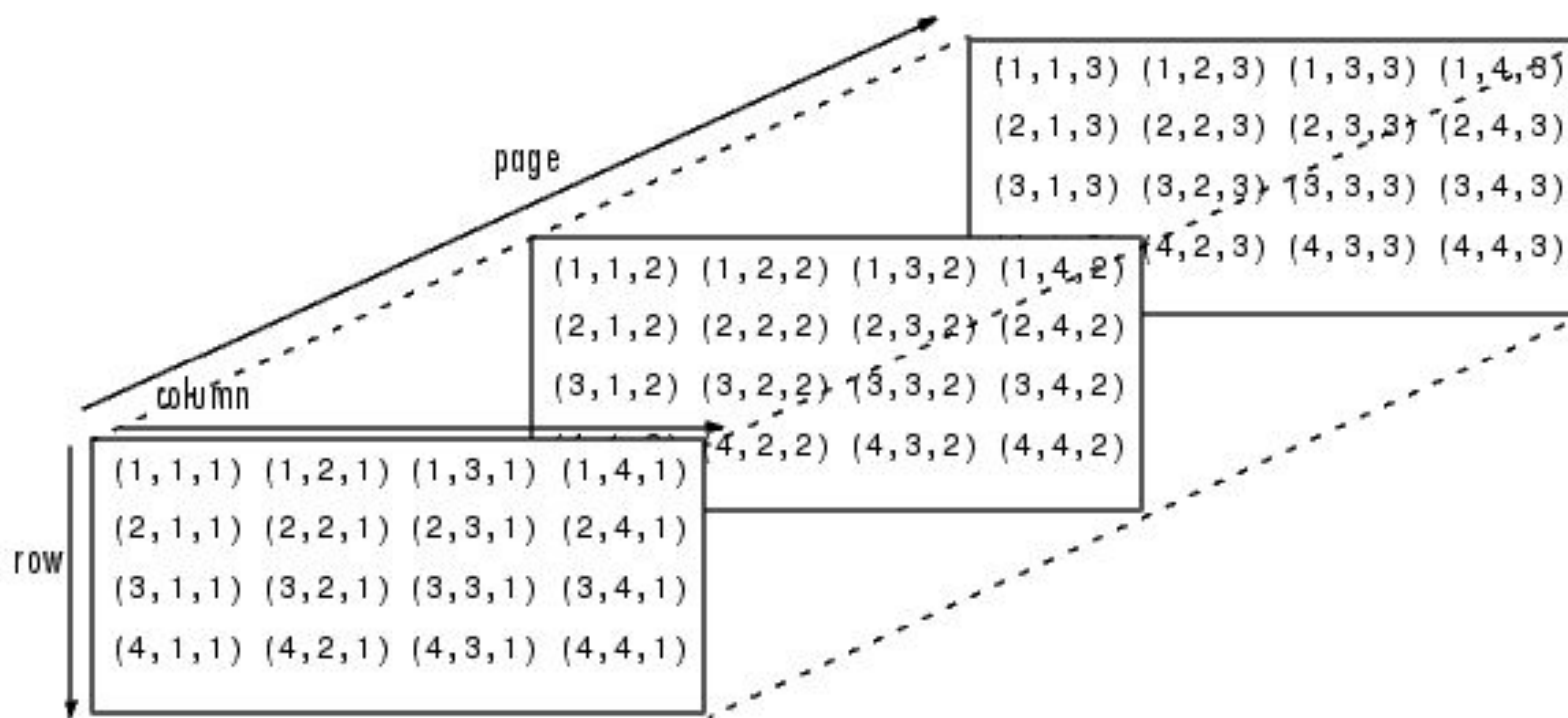
# Thảo luận

Mảng hai chiều

# Mảng đa chiều



- Mảng đa chiều là mảng có các phần tử là các mảng khác
- Có thể có mảng 2 chiều, 3 chiều... hoặc nhiều hơn
- Mảng càng nhiều chiều thì độ phức tạp khi xử lý càng cao



# Số chiều của mảng

---



- Mảng một chiều cần 1 chỉ số để xác định vị trí của phần tử mảng

chiều



5

8

9

2

10

3

# Số chiều của mảng

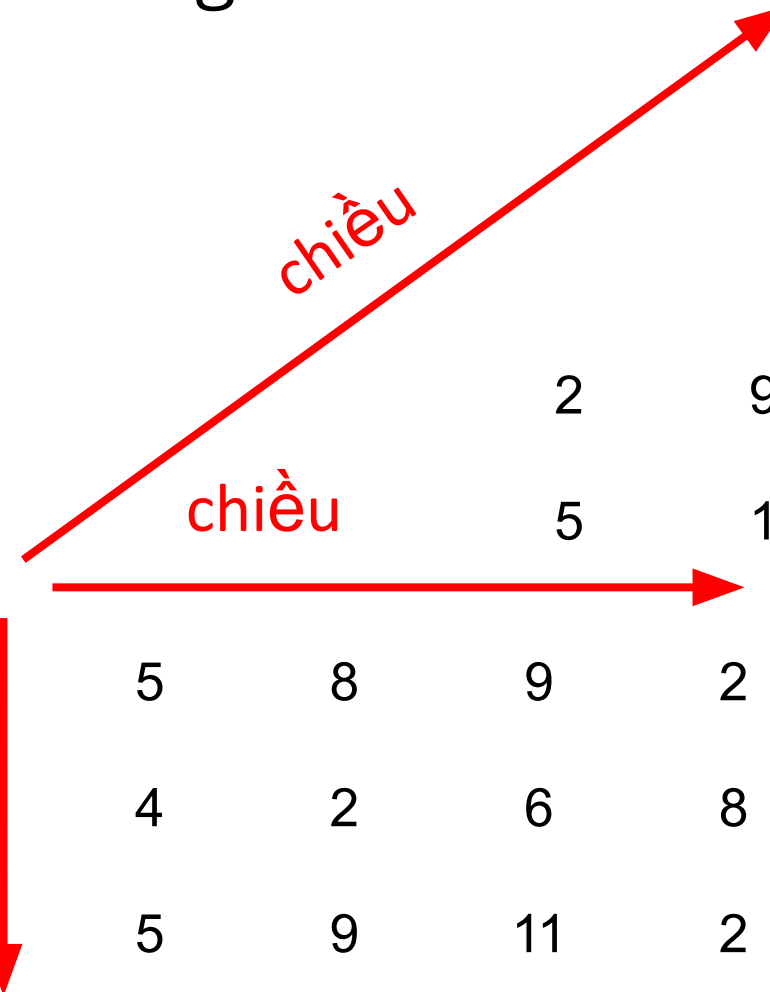


- Mảng hai chiều cần 2 chỉ số để xác định vị trí của phần tử mảng

5	8	9	2	10	3
4	2	6	8	1	0
5	9	11	2	4	6

# Số chiều của mảng

- Mảng 3 chiều cần 3 chỉ số để xác định vị trí của phần tử mảng



				3	4	6	8	20	3
				7	4	1	3	3	5
		2	9	4	6	6	7	4	6
		5	1	5	6	8	3		
	5	8	9	2	10	3	6	1	
	4	2	6	8	1	0			
	5	9	11	2	4	6			



# Mảng hai chiều

- Mảng hai chiều là mảng nhiều chiều được sử dụng phổ biến
- Mảng hai chiều bao gồm các phần tử, mỗi phần tử là một mảng một chiều.
- Mảng hai chiều là một bảng gồm  $n$  dòng và  $m$  cột:

	Cột 0	Cột 1	Cột 2	Cột 3
Dòng 0	<code>a[0][0]</code>	<code>a[0][1]</code>	<code>a[0][2]</code>	<code>a[0][3]</code>
Dòng 1	<code>a[1][0]</code>	<code>a[1][1]</code>	<code>a[1][2]</code>	<code>a[1][3]</code>
Dòng 2	<code>a[2][0]</code>	<code>a[2][1]</code>	<code>a[2][2]</code>	<code>a[2][3]</code>

Tên mảng ↑ chỉ số dòng ↑ chỉ số cột

- Số phần tử trong mảng là  $n * m$

# Mảng hai chiều: Ví dụ



- Ví dụ: mảng 2 chiều gồm 3 dòng, mỗi dòng có 5

		0	1	2	3	4
index	0	$c[0][0] = 11$	$c[0][1] = 2$	$c[0][2] = 7$	$c[0][3] = 9$	$c[0][4] = 5$
	1	$c[1][0] = 23$	$c[1][1] = 34$	$c[1][2] = 65$	$c[1][3] = 67$	$c[1][4] = 2$
	2	$c[2][0] = 38$	$c[2][1] = 28$	$c[2][2] = 43$	$c[2][3] = 72$	$c[2][4] = 65$

- Tổng số phần tử trong mảng  $3 * 5 = 15$
- **Index:** các chỉ số dòng và cột

# Phần tử và chỉ số mảng 2 chiều

---



- Một phần tử được xác định bằng chỉ số dòng và chỉ số cột trong mảng
- Ví dụ mảng *numbers* với 3 cột x 5 dòng, có tổng cộng 15 phần tử lần lượt là:
  - *numbers[0][0]*
  - *numbers[1][0]*
  - ...
  - *numbers[2][4]*

# Khởi tạo mảng hai chiều: Cách 1

- Sử dụng dấu ngoặc vuông để khởi tạo mảng với các phần tử
- Ví dụ, khởi tạo mảng có 3 dòng và 3 cột:

```
let cities = [  
  ["Hanoi", "Saigon", "DaNang"],  
  ["New York", "California", "Miami"],  
  ["Tokyo", "Nagoya", "Osaka"]  
]
```

	0	1	2
0	Hanoi	Saigon	DaNang
1	New York	California	Miami
2	Tokyo	Nagoya	Osaka



# Khởi tạo mảng hai chiều: Cách 2

- Sử dụng từ khóa new để khởi tạo mảng
- Ví dụ, khởi tạo mảng với 3 dòng và 3 cột

```
let cities = new Array(3);  
for (let i = 0; i < 3; i++) {  
    cities[i] = new Array(3);  
}  
  
cities[0][0] = "Hanoi";    cities[0][1] = "Saigon";    cities[0][2] = "DaNang";  
cities[1][0] = "New York"; cities[1][1] = "California"; cities[1][2] = "Miami";  
cities[2][0] = "Tokyo";    cities[2][1] = "Nagoya";    cities[2][2] = "Osaka";
```



# Duyệt mảng đa chiều

---

- Sử dụng vòng lặp lồng nhau để duyệt mảng đa chiều
- Sử dụng 2 vòng lặp lồng nhau để duyệt mảng 2 chiều
- Ví dụ:

```
for (let i = 0; i < cities.length; i++) {  
    for (let j = 0; j < cities[i].length; j++) {  
        document.write(cities[i][j] + "<br/>")  
    }  
}
```



---

# Demo

Cách sử dụng mảng hai chiều



---

# Thảo luận

Các thao tác với mảng hai chiều



# Khởi tạo giá trị ngẫu nhiên

---



- Ví dụ:

```
let matrix = new Array(10).fill(new Array(10));

for (let row = 0; row < matrix.length; row++) {
  for (let column = 0; column < matrix[row].length; column++) {
    matrix[row][column] = Math.floor(Math.random() * 100) + 1;
  }
}
```

# Hiển thị các phần tử của mảng

---



- Ví dụ:

```
for (let row = 0; row < matrix.length; row++) {  
    for (let column = 0; column < matrix[row].length; column++) {  
        console.log(matrix[row][column] + " ");  
    }  
}
```

# Tóm tắt bài học

---

- Mảng cho phép lưu trữ nhiều giá trị cùng kiểu
- Các khái niệm của mảng: Tên mảng, kiểu dữ liệu, kích thước, phần tử, chỉ số
- Tên của mảng tuân theo quy tắc của tên biến
- Chỉ số của phần tử đầu tiên là 0
- Chỉ số của phần tử cuối cùng là  $\text{length} - 1$
- Có thể sử dụng vòng lặp for và for-each để duyệt mảng
- Để sao chép mảng thì cần sao chép lần lượt từng phần tử của mảng



# Tóm tắt bài học

---

- Mảng đa chiều được sử dụng phổ biến là 2 chiều
- Số lượng “chiều” của mảng bằng với số lượng chỉ số để truy xuất đến một phần tử của mảng
- Ragged array là mảng hai chiều trong đó kích thước của các dòng là không bằng nhau
- Có thể sử dụng 2 vòng lặp lồng nhau để duyệt qua các phần tử của mảng hai chiều

# Hướng dẫn

Hướng dẫn làm bài thực hành và bài tập

Chuẩn bị bài tiếp theo: *Hàm*