



THỰC HỌC – THỰC NGHIỆP



Conceive Design Implement Operate

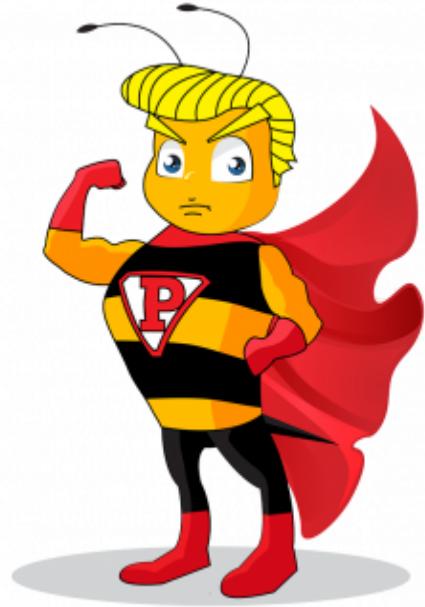
NHẬP MÔN LẬP TRÌNH

ĐẠI CƯƠNG LẬP TRÌNH

BÀI 6: MẢNG



- Hiểu được cấu trúc của mảng
- Phân biệt được mảng 1 chiều và mảng nhiều chiều
- Thực hiện được các thao tác mảng 1 chiều và 2 chiều
 - Khai báo
 - Truy xuất phần tử
 - Gán dữ liệu cho mảng
 - Duyệt mảng
 - Sắp xếp các phần tử mảng



PHẦN 1: MẢNG 1 CHIỀU

- Tìm hiểu về mảng
- Cách khai báo mảng
- Nhập dữ liệu vào mảng
- Truy xuất dữ liệu trong mảng



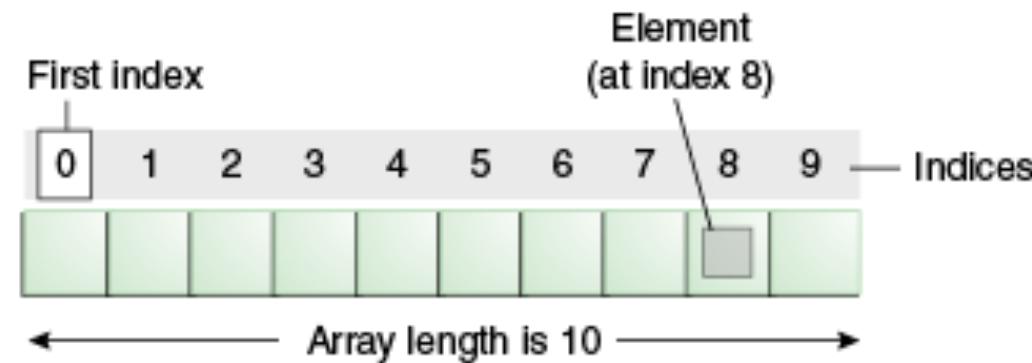
Ví dụ

- ❖ Chương trình cần lưu trữ **3** số nguyên?
=> Khai báo **3** biến **int a1, a2, a3;**
- ❖ Chương trình cần lưu trữ **100** số nguyên?
=> Khai báo **100** biến kiểu số nguyên!
- ❖ Người dùng muốn nhập **n** số nguyên?
=> Không thực hiện được!

Giải pháp

- ❖ Kiểu dữ liệu mới cho phép **lưu trữ một dãy** các số nguyên và **dễ dàng truy xuất**.

- ❑ Mảng là tập hợp các phần tử cùng kiểu.
- ❑ Mảng có số lượng phần tử cố định và được cấp phát vùng nhớ liên tục.
- ❑ Truy xuất các phần tử mảng bằng chỉ số, bắt đầu là 0
- ❑ Ví dụ:
 - ❖ int a[10] = {5,8,22,1,7,6,11,25,33,9};



❑ Lợi ích của mảng

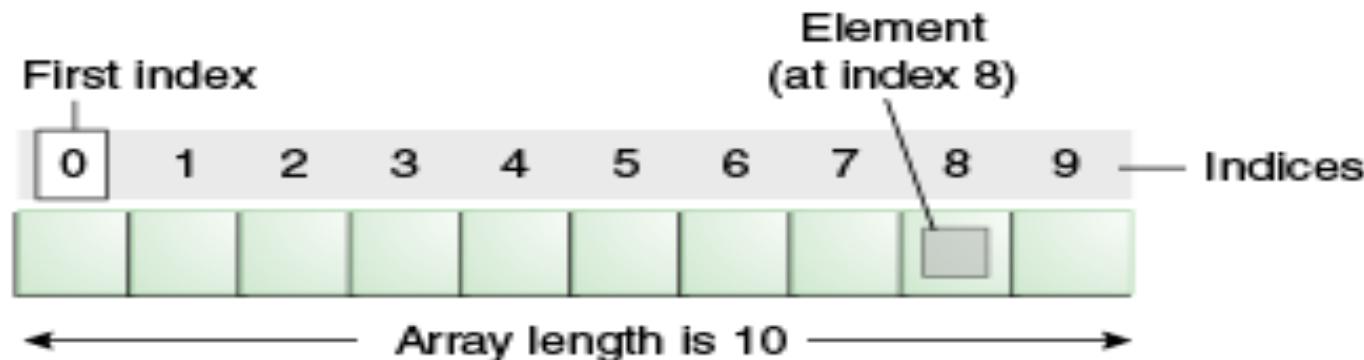
- ❖ Sử dụng mảng để nắm giữ nhiều giá trị thay vì phải khai báo nhiều biến.
- ❖ Truy xuất nhanh
- ❖ Dễ dàng đọc dữ liệu từ các phần tử và sắp xếp

❑ Bất lợi của mảng

- ❖ Số phần tử cố định nên không thể bổ sung hoặc bớt
- ❖ Cấp phát liên tục nên cần phải có vùng bộ nhớ liên tục đủ lớn để cấp phát.

- Mảng một chiều
- Mảng nhiều chiều

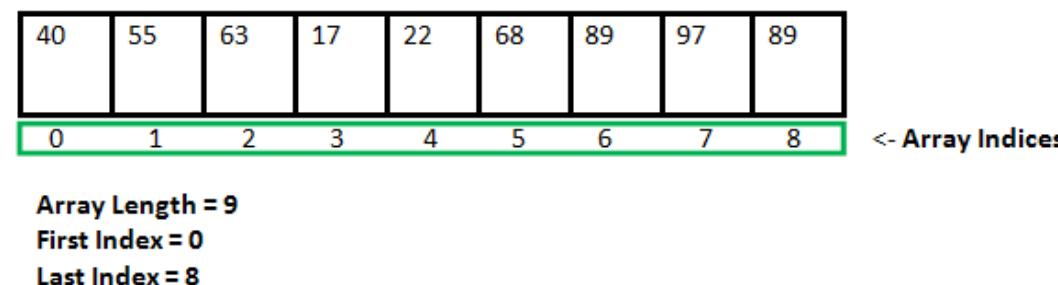
1 chiều



2 chiều

	0	1	2	3	
0	a[0][0]	a[0][1]	a[0][2]	a[0][3]	0
1	a[1][0]	a[1][1]	a[1][2]	a[1][3]	1
2	a[2][0]	a[2][1]	a[2][2]	a[2][3]	2

- ❑ Là một **kiểu dữ liệu có cấu trúc** do người lập trình định nghĩa.
- ❑ Biểu diễn một **dãy các biến có cùng kiểu**. Ví dụ: dãy các số nguyên, dãy các ký tự...
- ❑ Kích thước được **xác định ngay khi khai báo** và **không bao giờ thay đổi**.
- ❑ Ngôn ngữ lập trình C luôn chỉ định **một khối nhớ liên tục** cho một biến kiểu mảng.



❖ Cú pháp:

<kiểu cơ sở> <tên biến mảng> [<số phần tử>];

❖ Ví dụ:

```
int Mang1Chieu[10];
```

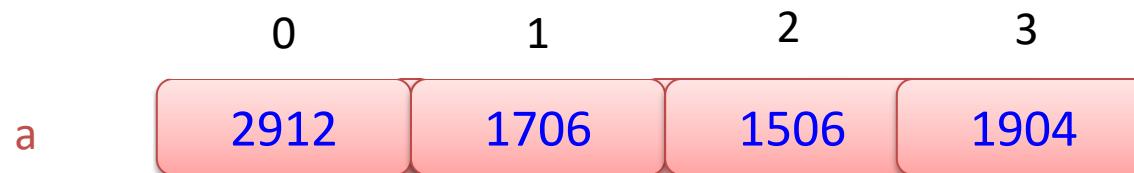
Mang1Chieu



❖ Gồm các cách sau

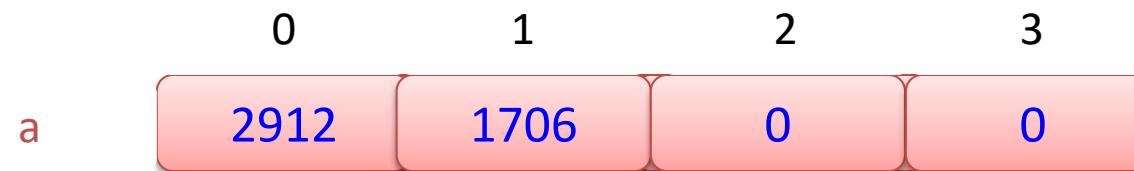
- ❑ Khởi tạo giá trị cho mọi phần tử của mảng

```
int a[4] = {2912, 1706, 1506, 1904};
```



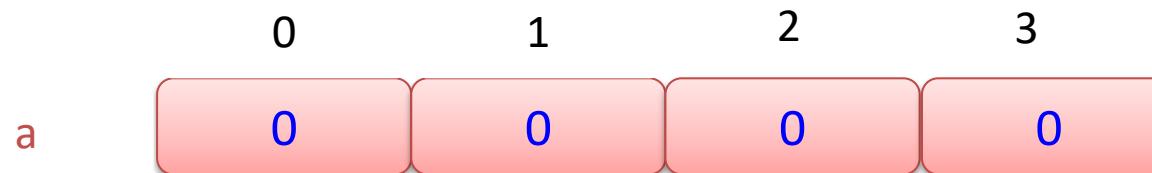
- ❑ Khởi tạo giá trị cho một số phần tử đầu mảng

```
int a[4] = {2912, 1706};
```



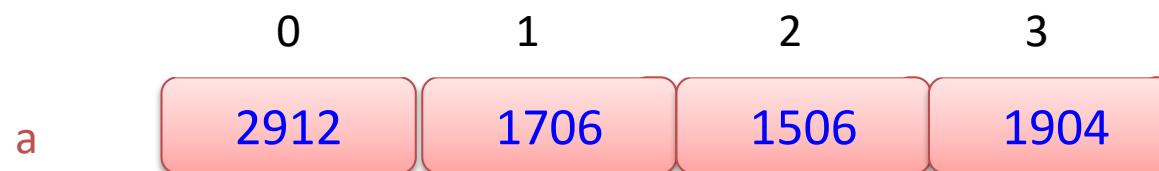
- Khởi tạo giá trị 0 cho mọi phần tử của mảng

```
int a[4] = {0};
```



- Tự động xác định số lượng phần tử

```
int a[] = {2912, 1706, 1506, 1904};
```



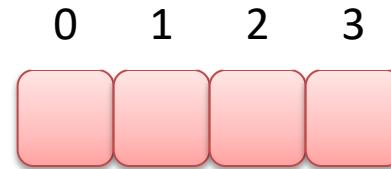
❖ Thông qua chỉ số

<tên biến mảng> [<gt cs1>] [<gt cs2>] ... [<gt csn>]

❖ Ví dụ

- ❑ Cho mảng như sau

int a[4];



- ❑ Các truy xuất

➤ Hợp lệ: a[0], a[1], a[2], a[3]

➤ Không hợp lệ: a[-1], a[4], a[5], ...

- ☐ Có thể sử dụng bất kỳ vòng lặp nào để duyệt mảng. Tuy nhiên vòng lặp thường được sử dụng để duyệt mảng là for.

```
int i = 0;
int marks[5]; // khai báo mảng
marks[0] = 80; // khởi tạo mảng
marks[1] = 60;
marks[2] = 70;
marks[3] = 85;
marks[4] = 75;

//duyệt mảng

for (i = 0; i < 5; i++) {
    printf("%d \n", marks[i]);
}
```

```
int i = 0;
int marks[5]={20, 30, 40, 50, 60};

//duyệt mảng

for (i = 0; i < 5; i++) {
    printf("%d \n", marks[i]);
}
```

- ❖ Nhập và xuất giá trị của mảng, sau đó tìm 1 giá trị trong mảng

```
int a[10],x;  
  
for(int i = 0; i < 10; i++) { //nhập giá trị vào mảng  
    scanf("%d", &a[i]);  
}  
  
for (int i = 0; i < 10; i++) { //hiển thị giá trị mảng  
    printf("%d", a[i]);  
}  
  
printf("Nhập phần tử cần tìm x = ");  
scanf("%d", &x);  
  
for(int i = 0; i < 10; i++) {  
    if(x == a[i]) {  
        printf("%d có trong mảng", x);  
        break;  
    }  
}
```



NHẬP XUẤT DỮ LIỆU TRONG MẢNG

- Hiểu được cấu trúc của mảng
- Thực hiện được các thao tác mảng Khai báo
 - Truy xuất phần tử
 - Gán dữ liệu cho mảng
 - Duyệt mảng





CHÈN QUIZ



PHẦN 2: SẮP XẾP MẢNG & MẢNG 2 CHIỀU

- ❖ Sắp xếp các sản phẩm theo giá từ thấp đến cao
- ❖ Sắp xếp sinh viên theo điểm từ cao tới thấp
- ❖ Sắp xếp các bộ phim theo số lượt xem



- ❖ Trong lập trình: Sắp xếp là sắp đặt các phần tử của một cấu trúc theo thứ tự tăng dần (hay giảm dần).
- ❖ Với một cấu trúc đã được sắp xếp chúng ta rất thuận tiện khi thực hiện các tác vụ trên cấu trúc như tìm kiếm, trích lọc, duyệt cấu trúc
- ❖ Để sắp xếp, người ta đưa ra nhiều thuật toán sắp xếp.

- ❖ Bubble sort.
- ❖ Quick sort.
- ❖ Simple selection sort.
- ❖ Heap sort.
- ❖ Simple insertion sort.
- ❖ Shell sort.
- ❖ Merge sort.
- ❖ Radix sort.

- ❖ Thuật toán Bubble Sort sẽ duyệt danh sách nhiều lần, trong mỗi lần duyệt sẽ lần lượt so sánh cặp nút thứ i và thứ $i+1$ và đổi chỗ hai nút này nếu chúng không đúng thứ tự.

6 5 3 1 8 7 2 4

- ❖ Chúng ta có thể sắp xếp tăng dần hoặc giảm dần các phần tử trong mảng

```
int a[7]={8,2,6,2,9,1,5};  
int i , j, temp;  
for(i=0;i<6;i++){  
    for(j=i+1;j<7;j++){  
        if(a[i]>a[j])  
        {  
            temp=a[i];  
            a[i]=a[j];  
            a[j]=temp;  
        }  
    }  
}
```

Nếu thay đổi toán tử so sánh thành < thì thuật toán trở thành sắp xếp giảm dần.

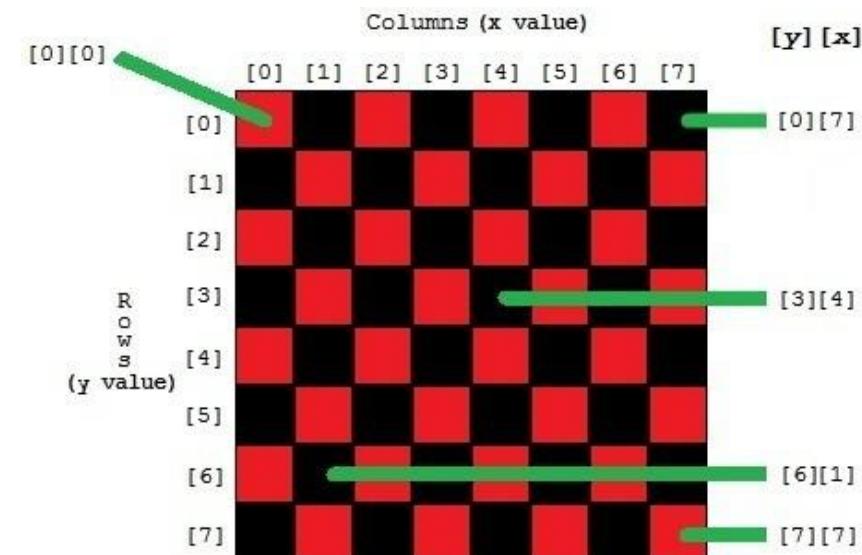


SẮP XẾP MẢNG SỐ NGUYÊN

- ❖ Mảng 2 chiều trong C được biểu diễn dưới dạng hàng và cột, còn được gọi là ma trận.
- ❖ Mảng 2 chiều còn được gọi là mảng chứa n phần tử mảng 1 chiều



- ❖ Bàn cờ vua, cờ tướng
- ❖ Xử lý đồ họa (2D,3D)
- ❖ Ứng dụng trong rất nhiều các chương trình cần sự tính toán
 - ❖ Xác suất thống kê
 - ❖ Toán kinh tế
 - ❖ ...



❖ Cú pháp khai báo

<kiểu cơ sở> <tên biến mảng>[<N1>][<N2>]...[<Nn>];

<N1>, ..., <Nn> : số lượng phần tử của mỗi chiều.

❖ Ví dụ:

```
int Mang2Chieu[3][5];
```

Mang2Chieu	0	1	2	3	4
0					
1					
2					

- ❖ Khởi tạo mảng 2 chiều bằng cách gán giá trị.

```
int i = 0;  
  
int j = 0;  
  
int ma_tran[4][3] = {{ 1, 2, 3 }, {2, 3, 4}, {3, 4, 5}, {4, 5, 6}};  
  
for (i = 0; i < 4; i++) { //duyệt mảng  
    for (j = 0; j < 3; j++) {  
        printf("%d ", ma_tran[i][j]);  
    }  
    printf("\n");  
}
```

- ❖ Khởi tạo mảng 2 chiều bằng cách nhập giá trị từ bàn phím.

```
int i, j;  
int ma_tran[4][3];  
printf("Nhập mảng: \n");  
  
for (int i = 0; i < 4; i++) { //nhập mảng  
    for (int j = 0; j < 3; j++) {  
        printf("Nhập a[%d][%d] = ", i, j);  
        scanf("%d", &ma_tran[i][j]);  
    }  
    printf("\n");  
}
```

- ❑ Dùng 2 vòng lặp for lồng với nhau để duyệt mảng 2 chiều.

```
int i = 0;  
int j = 0;  
int ma_tran[4][3] = {{ 1, 2, 3 }, {2, 3, 4}, {3, 4, 5}, {4, 5, 6}};  
for (i = 0; i < 4; i++) {  
    for (j = 0; j < 3; j++) {  
        printf("%d ", ma_tran[i][j]);  
    }  
    printf("\n");  
}
```

❖ Thông qua chỉ số

<tên biến mảng> [<giá trị cs1>] [<giá trị cs2>]

❖ Ví dụ

❑ Cho mảng 2 chiều như sau

int a[3][4];

❑ Các truy xuất

➤ Hợp lệ: a[0][0], a[0][1], ..., a[2][2], a[2][3]

➤ Không hợp lệ: a[-1][0], a[2][4], a[3][3]

	0	1	2	3
0				
1				
2				



NHẬP XUẤT MẢNG 2 CHIỀU

- Thuật toán sắp xếp
- Sắp xếp mảng 1 chiều
- Mảng 2 chiều
 - Khai báo
 - Truy xuất phần tử
 - Gán dữ liệu cho mảng
 - Duyệt mảng





CHÈN QUIZ

