

# Mảng

GV. Nguyễn Minh Huy

# Nội dung



- Khái niệm mảng 1 chiều.
- Thao tác mảng 1 chiều.
- Mảng 2 chiều và thao tác.



- **Khái niệm mảng 1 chiều.**
- Thao tác mảng 1 chiều.
- Mảng 2 chiều và thao tác.

# Khái niệm mảng 1 chiều



## ■ Xét chương trình sau:

- Nhập 5 số nguyên, sau đó xuất 5 số vừa nhập.

- Khai báo 5 biến int a1, a2, a3, a4, a5.

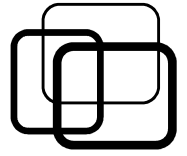
- Nhập 50 số nguyên, sau đó xuất 50 số vừa nhập.

- Khai báo 50 biến int!!

➔ Làm sao khai báo nhiều biến cùng lúc?

➔ Mảng 1 chiều.

# Khái niệm mảng 1 chiều



## ■ Mảng 1 chiều trong C:

- Một dãy biến liên tục có cùng kiểu.
- Các biến trong dãy là phần tử mảng.
- Khai báo:

<Kiểu dữ liệu> **<Tên mảng>**[ <Số phần tử> ];

<Số phần tử>: phải là một hằng số.

```
int    m1[ 10 ];    // Dãy 10 số nguyên.
```

```
float  m2[ 50 ];    // Dãy 50 số thực.
```

```
int    N;
```

```
float  m3[ N ];      // Sai
```

```
const int K = 100;
```

```
float  m4[ K ];      // Đúng
```



# Khái niệm mảng 1 chiều



## ■ Mảng 1 chiều trong C:

### ■ Truy xuất phần tử mảng:

<Tên mảng> [ <**Chỉ số mảng**> ]

<Chỉ số mảng>: một số nguyên từ **0** đến **<Số phần tử> - 1**.

```
int a[ 10 ] = { 0 };    a
```

0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	0	0	0	0	0

a[ 0 ] = 5;

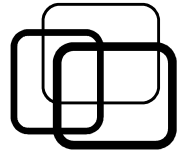
a[ 1 ] = 6;

a[ 2 ] = a[ 0 ] + a[ 1 ];

a[ -1 ] = 7;           // Sai

a[ 10 ] = 8;           // Sai

# Khái niệm mảng 1 chiều



## ■ Mảng 1 chiều trong C:

### ■ Truyền tham số mảng:

- Cách 1: `void foo( int a[ 100 ], int size );`  
➔ Chấp nhận mảng 100 phần tử.
- Cách 2: `void foo( int a[ ], int size );`  
➔ Chấp nhận mảng số phần tử bất kỳ.
- Phần tử mảng CÓ THỂ THAY ĐỔI sau khi ra khỏi hàm.

```
void foo( int a[ ], int size )  
{  
    a[ 2 ] = 9;  
    a[ 5 ] = 8;  
}
```

```
int main()  
{  
    int a[ 100 ] = { 0 };  
    int b[ 200 ] = { 0 };  
    foo( a, 50 );  
    foo( b, 70 );  
    // a[2], a[5] bị thay đổi.  
    // b[2], b[5] bị thay đổi.  
}
```





- Khái niệm mảng 1 chiều.
- **Thao tác mảng 1 chiều.**
- Mảng 2 chiều và thao tác.

# Thao tác mảng 1 chiều



## ■ Cách thức chung:

### ■ B1: Duyệt mảng.

- Dùng **vòng lặp + biến đếm**.

### ■ B2: Thao tác trên từng phần tử.

- Dùng biến đếm truy xuất phần tử.

// Duyệt mảng A có kích thước N.

```
for ( int i = 0; i < N; i++ )  
{  
    <Xử lý phần tử A[ i ]>;  
}
```

# Thao tác mảng 1 chiều



## ■ Nhập xuất mảng:

```
void nhapMang( int a[ ], int &n )
{
    printf("Nhap kích thước = ");
    scanf("%d", &n);

    for (int i = 0; i < n; i++)
    {
        printf("Nhap phần tử %d = ", i);
        scanf("%d", &a[ i ]);
    }
}

void xuatMang( int a[ ], int n )
{
    for ( int i = 0; i < n; i++ )
        printf("%d ", a[ i ]);
}
```

```
#define MAX 100

int main()
{
    int m[ MAX ], size;

    nhapMang(m, size);
    xuatMang(m, size);
}
```

# Thao tác mảng 1 chiều



## ■ Tính tổng phần tử mảng:

```
// Tính tổng mảng a, kích thước n
long  tinhTong( int a[ ], int n )
{
    long tong = 0;

    for ( int i = 0; i < n; i++ )
        tong += a[ i ];

    return tong;
}
```

```
#define MAX 100

int main()
{
    int m[ MAX ], size;

    nhapMang(m, size);
    long tong =  tinhTong(m, size);
}
```

# Thao tác mảng 1 chiều



## ■ Các thao tác thông dụng:

### ■ Sao chép mảng:

```
int a[ ] = { 1, 2, 3, 4, 5 };  
int b[ 5 ];  
b = a; // SAI, không sao chép bằng phép gán!  
copyArray( b, a, n ); // Viết hàm sao chép.
```

### ■ Chèn phần tử:

```
int a[ ] = { 1, 2, 4, 5 };  
int x = 3;  
int pos = 2;  
insertArray( a, x, pos ); // a = { 1, 2, 3, 4, 5 }
```

### ■ Xóa phần tử:

```
int a[ ] = { 1, 2, 3, 4, 5 };  
int pos = 2;  
deleteArray( a, pos ); // a = { 1, 2, 4, 5 }
```



- Khái niệm mảng 1 chiều.
- Thao tác mảng 1 chiều.
- **Mảng 2 chiều và thao tác.**

# Mảng 2 chiều và thao tác



## ■ Xét chương trình sau:

### ■ Nhập và xuất ma trận $5 \times 10$ .

➤ Khai báo 5 mảng: `int a1[10], a2[10], a3[10], a4[10], a5[10]`.

### ■ Nhập và xuất ma trận $50 \times 10$ .

➤ Khai báo 50 mảng!!

➔ Làm sao biểu diễn ma trận  $M \times N$ ?

# Mảng 2 chiều và thao tác



## ■ Giải pháp 1:

- Dùng mảng một chiều!!
- Biểu diễn ma trận  $M \times N$ :
  - Khai báo mảng một chiều  $M \times N$  phần tử.
  - Để truy xuất dòng  $i$  cột  $j$ 
    - ➔ Truy xuất phần tử  $[i * N + j]$ .



# Mảng 2 chiều và thao tác



## ■ Giải pháp 2:

- Dùng mảng 2 chiều.

- Khai báo:

<Kiểu dữ liệu> **<Tên mảng>**[<Số dòng>] [<Số cột>];

<Số dòng>, <Số cột> phải là một hằng số.

```
int    m1[ 5 ][ 10 ]; // Ma trận 5 x 10 số nguyên.
```

```
int    m2[ M ][ N ]; // Sai.
```

- Truy xuất phần tử:

**<Tên mảng> [ <Chỉ số dòng> ] [ <Chỉ số cột> ]**

<Chỉ số dòng>: một số nguyên từ **0** đến **<Số dòng> - 1**.

<Chỉ số cột>: một số nguyên từ **0** đến **<Số cột> - 1**.

```
m1[ 0 ][ 2 ] = 5;
```

```
m1[ 1 ][ 3 ] = 6;
```

```
m1[ -1 ][ 10 ] = 7; // Sai.
```

# Mảng 2 chiều và thao tác



## ■ Giải pháp 2:

### ■ Khởi tạo:

```
<Kiểu dữ liệu> <Tên mảng>[<Số dòng>][<Số cột>] =  
{  
    <Khởi tạo dòng 0>,  
    <Khởi tạo dòng 1>,  
    ...  
};
```

// Khởi tạo tất cả phần tử.

```
int m1[ 3 ][ 5 ] =
```

```
{  
    { 1, 1, 1, 1, 1 },  
    { 1, 2, 3, 4, 5 },  
    { 5, 4, 3, 2, 1 }  
};
```

// Khởi tạo vài phần tử.

```
int m1[ 3 ][ 5 ] =
```

```
{  
    { 1, 1 },  
    { 1, 2, 3 },  
    { 0 }  
};
```

// Tự động biết số dòng.

```
int m1[ ][ 5 ] =
```

```
{  
    { 1, 1 },  
    { 1, 2, 3 },  
    { 0 }  
};
```

# Mảng 2 chiều và thao tác



## ■ Cách thức chung:

### ■ B1: Duyệt mảng.

- Dùng **2 vòng lặp lồng nhau**.
- Vòng lặp ngoài: duyệt từng dòng.
- Vòng lặp trong: duyệt từng phần tử.

### ■ B2: Thao tác trên từng phần tử.

- Dùng biến đếm truy xuất phần tử.

// Duyệt ma trận A có kích thước M dòng N cột.

```
for ( int i = 0; i < M; i++ )  
    for ( int j = 0; j < N; j++ )  
    {  
        <Lệnh truy xuất phần tử A[ i ][ j ]>;  
    }
```

# Mảng 2 chiều và thao tác



## ■ Nhập ma trận:

```
// Nhập ma trận số nguyên A, M dòng N cột
void nhapMaTran( int A[ ][ MAX ], int &M, int &N )
{
    printf("Nhap dong cot = ");
    scanf("%d %d", &M, &N);

    for (int i = 0; i < M; i++)
        for (int j = 0; j < N; j++)
        {
            printf("Nhap phan tu %d, %d = ", i, j);
            scanf("%d", &A[ i ][ j ]);
        }
}

#define MAX 100

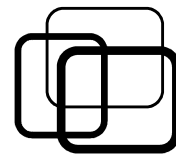
int main()
{
    int a[ MAX ][ MAX ];
    int M, N;

    nhapMang(a, M, N);
}
```



- Khái niệm mảng 1 chiều:
  - Dãy biến cùng kiểu.
  - Các biến trong dãy là phần tử mảng.
- Thao tác mảng 1 chiều:
  - Thao tác chung: duyệt + thao tác từng phần tử.
- Mảng 2 chiều và thao tác:
  - Giải pháp 1: mảng một chiều.
  - Giải pháp 2: mảng 2 chiều.
  - Ma trận các biến cùng kiểu.





## ■ Bài tập 7.1:

Viết chương trình C (tổ chức theo dạng hàm và chia làm nhiều file):

- Nhập vào mảng N số nguyên.
- Hãy cho biết:
  - a) Có bao nhiêu số âm trong mảng.
  - b) Có bao nhiêu số nguyên tố trong mảng.

*Định dạng nhập:*

*Nhap N = 3*

*Phan tu 0 = 2*

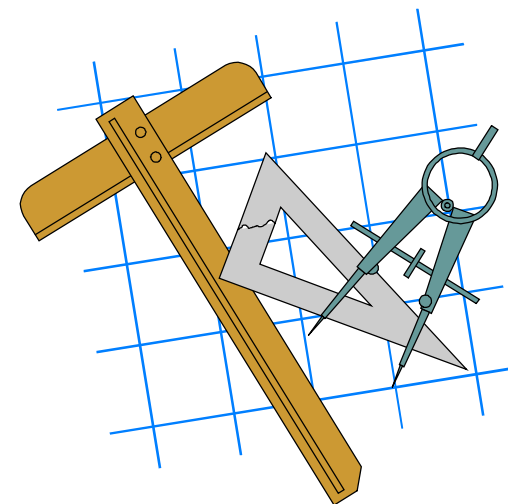
*Phan tu 1 = 3*

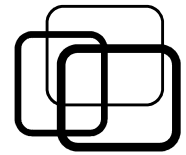
*Phan tu 2 = -6*

*Định dạng xuất:*

*Co 1 so am.*

*Co 2 so nguyen to.*





## ■ Bài tập 7.2:

Viết chương trình C kiểm tra mảng như sau:  
(tổ chức theo dạng hàm và chia làm nhiều file)

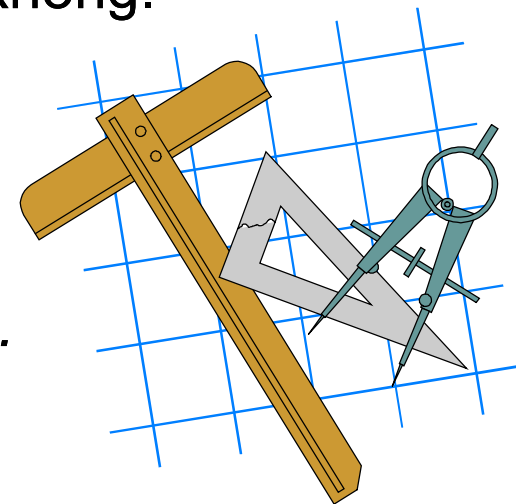
- Nhập vào mảng N số nguyên.
- Hãy cho biết:
  - a) Mảng có tăng dần không.
  - b) Mảng có đối xứng không.
  - c) Mảng có lập thành một cấp số cộng không.

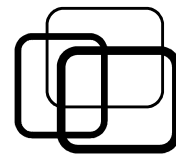
*Định dạng xuất:*

*Mang <tang/khong tang> dan.*

*Mang <doi xung/khong doi xung>.*

*Mang <lap thanh/khong lap thanh> cap so cong.*





## ■ Bài tập 7.3:

Viết chương trình C thao tác ma trận như sau:  
(tổ chức theo dạng hàm và chia làm nhiều file)

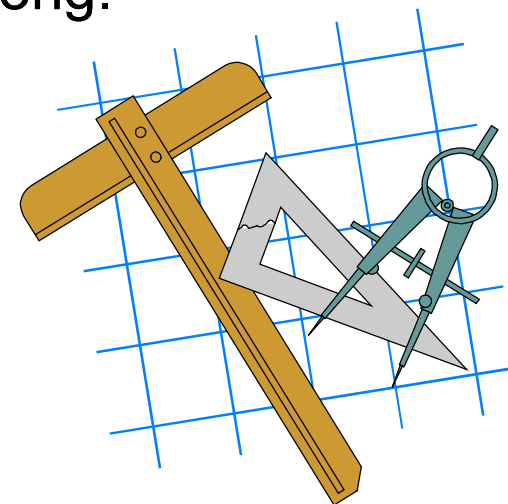
- Nhập vào ma trận vuông  $N \times N$  chứa số nguyên.
- Hãy cho biết:
  - a) Tổng phần tử nằm trên đường chéo chính/phụ.
  - b) Dòng có tổng lớn nhất.
  - c) Ma trận có là một ma phương hay không.

*Định dạng xuất:*

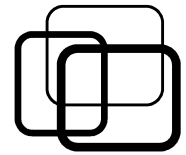
*Tong cheo chinh = <tong cheo chinh>.*

*Tong cheo phu = <tong cheo phu>.*

*Ma tran <la/khong la> ma phuong.*



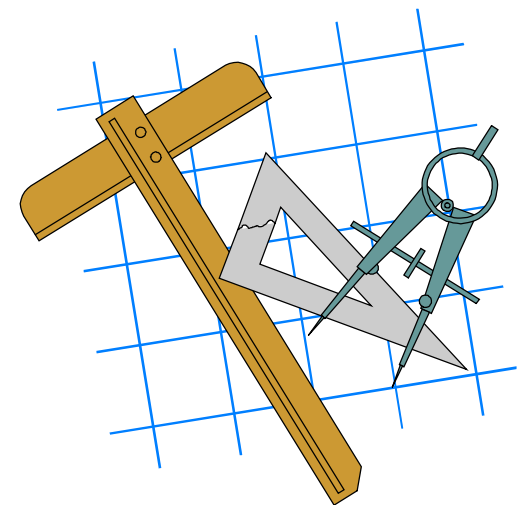


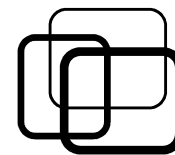


## ■ Bài tập 7.4:

Viết chương trình C thao tác ma trận như sau:  
(tổ chức theo dạng hàm và chia làm nhiều file)

- Nhập vào ma trận  $M \times N$  chứa số nguyên.
- Xuất ra màn hình các phần tử có giá trị bằng tổng các phần tử còn lại trên dòng và cột của nó.





## ■ Bài tập 7.5:

Viết chương trình C xoay ma trận như sau:

(tổ chức theo dạng hàm và chia làm nhiều file)

- Nhập vào ma trận nguyên  $M \times N$ .
- Xoay trái 90 độ ma trận và xuất ra màn hình kết quả.
- Xoay phải 90 độ ma trận và xuất ra màn hình kết quả.

