

NỘI DUNG CHÍNH BUỔI 13

MẢNG 2 CHIỀU

(1) Khái niệm

- Mảng hai chiều thực chất là mảng một chiều trong đó mỗi phần tử của mảng lại là mảng một chiều, và được truy xuất bởi hai chỉ số dòng và cột.

(2) Khai báo mảng 2 chiều

- Cú pháp: **<Kiểu> <tên mảng>[<Kích thước dòng>][<Kích thước cột>];**
- Mỗi phần tử của mảng được truy nhập thông qua tên mảng cùng với hai chỉ số: chỉ số dòng (bắt đầu từ 0 đến <Kích thước dòng> – 1) và chỉ số cột (bắt đầu từ 0 đến <Kích thước cột> – 1).

VD: `int a[2][3]; /* Khai báo mảng hai chiều gồm 6 phần tử kiểu int, bao gồm:`
`a[0][0] a[0][1] a[0][2]`
`a[1][0] a[1][1] a[1][2] */`

(3) Khởi tạo mảng 2 chiều

VD:

```
//Khởi tạo mảng 2 chiều có 2 dòng và 3 cột
int a[2][3]={11, 12, 13},{21, 22, 23}};
//Khởi tạo có thể không cần chỉ ra kích thước dòng
double b[][3]={6, 7.8, 8},{3, 12.6, 4},{6.5, 20, 7}};
```

(4) Nhập / xuất mảng 2 chiều

```
#include <stdio.h>
#include <conio.h>

#define MAXROW 10
#define MAXCOL 10

void main()
{
    int a[MAXROW][MAXCOL], d, c;

    //Nhập mảng 2 chiều
    do {
        printf("Nhập số dòng và số cột: ");
        scanf("%d%d", &d, &c);
    } while (d<1 || d>MAXROW || c<1 || c>MAXCOL);
    for(int i = 0; i < d; i++)
        for(int j = 0; j < c; j++)
        {
            printf("a[%d][%d]= ", i, j);
            scanf("%d", &a[i][j]);
        }
}
```

```

//Xuat mang 2 chieu
for(int i = 0; i < d; i++)
{
    for(int j = 0; j < c; j++)
        printf("%3d", a[i][j]);
    printf("\n");
}

_getch();
}

```

(5) Nhập / xuất mảng 2 chiều

- Tên mảng hai chiều cũng là một hằng địa chỉ và nó chính là địa chỉ phần tử đầu tiên của mảng.
- Khi dùng tên mảng làm tham số thực truyền cho hàm thì thực chất là địa chỉ phần tử đầu tiên của mảng được truyền cho hàm và như vậy tham số hình thức tương ứng trong định nghĩa hàm phải viết dưới dạng con trỏ.

```

void NhapMang2cConTro(int (*a) [MAXCOL], int *sd, int *sc)
void XuatMang2cConTro(int (*a) [MAXCOL], int sd, int sc)

```

- Tham số hình thức tương ứng với tham số thực là tên mảng hai chiều cũng có thể viết như sau:

```

void NhapMang2c(int a[] [MAXCOL], int &sd, int &sc)
void XuatMang2c(int a[] [MAXCOL], int sd, int sc)

```

```

#include <stdio.h>
#include <conio.h>

```

```

#define MAXROW 10
#define MAXCOL 10

```

```

void NhapMang2c(int a[] [MAXCOL], int &sd, int &sc)
{
    do {
        printf("Nhap so dong va so cot: ");
        scanf("%d%d", &sd, &sc);
    } while(sd<1||sd>MAXROW||sc<1||sc>MAXCOL);
    for(int i = 0; i < sd; i++)
        for(int j = 0; j < sc; j++)
        {
            printf("a[%d][%d]= ", i, j);
            scanf("%d", &a[i][j]);
        }
}

```

```
void NhapMang2cConTro(int (*a) [MAXCOL], int *sd, int *sc)
{
    do {
        printf("Nhap so dong va so cot: ");
        scanf("%d%d", &(*sd), &(*sc)); //scanf("%d%d", sd, sc);
    } while(*sd<1 || *sd>MAXROW || *sc<1 || *sc>MAXCOL);

    for(int i = 0; i < *sd; i++)
        for(int j = 0; j < *sc; j++)
        {
            printf("a[%d][%d]=", i, j);
            scanf("%d", *(a+i)+j);
        }
}

void XuatMang2c(int a[] [MAXCOL], int sd, int sc)
{
    for(int i = 0; i < sd; i++)
    {
        for(int j = 0; j < sc; j++)
            printf("%3d", a[i][j]);
        printf("\n");
    }
}

void XuatMang2cConTro(int (*a) [MAXCOL], int sd, int sc)
{
    for(int i = 0; i < sd; i++)
    {
        for(int j = 0; j < sc; j++)
            printf("%3d", (*(a+i)+j));
        printf("\n");
    }
}

void main()
{
    int a[MAXROW][MAXCOL], d, c;
    NhapMang2c(a, d, c);
    XuatMang2c(a, d, c);
    _getch();
}
```

(6) Các bài toán cơ bản trên mảng 2 chiều

- Duyệt hết các phần tử của mảng:**

a. Duyệt theo từng dòng từ trên xuống dưới, với mỗi dòng duyệt từ trái sang phải:

```
for(int i = 0; i < sd; i++)
    for(int j = 0; j < sc; j++)
    {
        //Xử lý a[i][j]
        ...
    }
```

b. Duyệt theo từng cột từ trái sang phải, với mỗi cột duyệt từ trên xuống dưới:

```
for(int j = 0; j < sc; j++)
    for(int i = 0; i < sd; i++)
    {
        //Xử lý a[i][j]
        ...
    }
```

- Duyệt các phần tử nằm trên cùng một dòng hay trên cùng một cột:**

a. Duyệt các phần tử trên dòng có chỉ số k ($0 \leq k < sd$):

```
for(int j = 0; j < sc; j++)
{
    <Xử lý a[k][j]>
}
```

b. Duyệt các phần tử trên cột có chỉ số k ($0 \leq k < sc$):

```
for(int i = 0; i < sd; i++)
{
    <Xử lý a[i][k]>
}
```

VD1: Viết hàm tính tổng giá trị các phần tử của mảng 2 chiều.

```
void TinhTongGiaTriCacPhanTu(int (*a) [MAXCOL], int sd, int sc)
{
    int tong=0;
    for(int i = 0; i < sd; i++)
        for(int j= 0; j < sc; j++)
            tong+=*(a+i)+j); //tong+=a[i][j];
    printf("\nTong gia tri cac phan tu la %d\n",tong);
}
```

```
void main()
{
    int a[MAXROW] [MAXCOL], d, c;
    NhapMang2c(a, d, c);
    XuatMang2c(a, d, c);
    TinhTongGiaTriCacPhanTu(a, d, c);
    _getch();
}
```

```
Nhap so dong va so cot: 2 3
a[0][0]=1
a[0][1]=2
a[0][2]=3
a[1][0]=4
a[1][1]=5
a[1][2]=6
1 2 3
4 5 6
Tong gia tri cac phan tu la 21
```

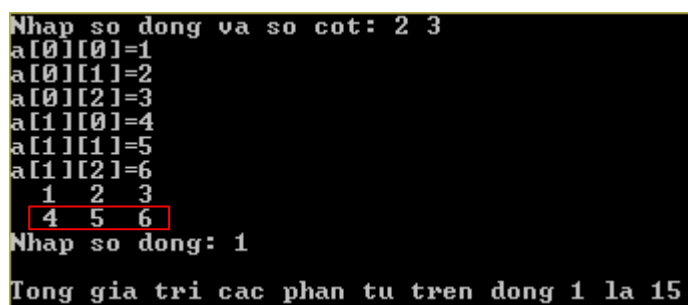
VD2: Viết hàm tính tổng giá trị các phần tử trên dòng k của mảng 2 chiều.

```

void TinhTongGiaTriCacPTuTrenDongk(int a[][MAXCOL], int sd, int sc)
{
    int dong, tong=0;
    do{
        printf("Nhap so dong: ");
        scanf("%d", &dong);
    }while(dong<0 || dong>=sd);
    for(int j=0; j<sc; j++)
        tong+=* (a+dong)+j); //tong+=a[dong][j];
    printf("\nTong gia tri cac phan tu tren dong %d la
           %d\n", dong, tong);
}

void main()
{
    int a[MAXROW][MAXCOL], d, c;
    NhapMang2c(a, d, c);
    XuatMang2c(a, d, c);
    TinhTongGiaTriCacPTuTrenDongk(a, d, c);
    _getch();
}

```

Kết quả:


```

Nhap so dong va so cot: 2 3
a[0][0]=1
a[0][1]=2
a[0][2]=3
a[1][0]=4
a[1][1]=5
a[1][2]=6
 1  2  3
 4  5  6
Nhap so dong: 1
Tong gia tri cac phan tu tren dong 1 la 15

```

VD3: Viết hàm sắp xếp tăng dần các phần tử trên cột k của mảng 2 chiều.

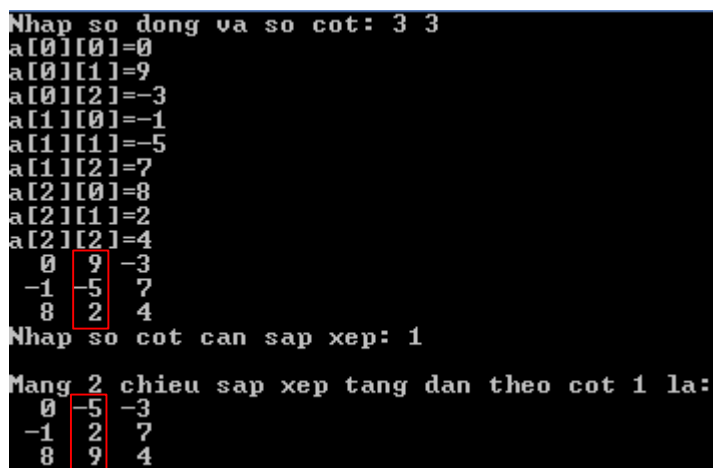
```

void SapXepTangDanTrenCotk(int a[][MAXCOL], int sd, int sc, int k)
{
    int tam;
    for (int i = 0; i < sd - 1; i++)
        for (int j = i + 1; j < sd; j++)
            if (a[i][k]>a[j][k])
            {
                tam = a[i][k];
                a[i][k] = a[j][k];
                a[j][k] = tam;
            }
}

```

```
void main()
{
    int a[MAXROW][MAXCOL], d, c;
    NhapMang2c(a, d, c);
    XuatMang2c(a, d, c);
    int cotk;
    do{
        printf("Nhap so cot can sap xep: ");
        scanf("%d", &cotk);
    } while (cotk < 0 || cotk >= c);
    SapXepTangDanTrenCotk(a,d,c,cotk);
    printf("Mang 2 chieu sap xep tang dan theo cot 2 la:\n");
    XuatMang2c(a, d, c);
    _getch();
}
```

Kết quả:



The screenshot shows the execution of a C program. It starts by asking for the number of rows and columns (3 and 3). Then, it displays the input array 'a' with values: a[0][0]=0, a[0][1]=9, a[0][2]=-3, a[1][0]=-1, a[1][1]=-5, a[1][2]=7, a[2][0]=8, a[2][1]=2, a[2][2]=4. Next, it asks for the column index to sort by (1). Finally, it displays the sorted array 'a' with values: a[0][0]=0, a[0][1]=-5, a[0][2]=-3, a[1][0]=-1, a[1][1]=2, a[1][2]=7, a[2][0]=8, a[2][1]=9, a[2][2]=4. Red boxes highlight the input column index 1 and the sorted array output.

```
Nhap so dong va so cot: 3 3
a[0][0]=0
a[0][1]=9
a[0][2]=-3
a[1][0]=-1
a[1][1]=-5
a[1][2]=7
a[2][0]=8
a[2][1]=2
a[2][2]=4
  0  9 -3
 -1 -5  7
  8  2  4
Nhap so cot can sap xep: 1
Mang 2 chieu sap xep tang dan theo cot 1 la:
  0 -5 -3
 -1  2  7
  8  9  4
```

(7) Mảng vuông (ma trận vuông)• **Tính chất:**

Mảng vuông là mảng hai chiều có số dòng = số cột = n , n gọi là cấp ma trận.

• **Đặc tính của ma trận vuông:**

- a. Các phần tử nằm trên đường chéo chính là $a[i][i]$ với $0 \leq i < n$



- b. Các phần tử nằm trên đường chéo phụ là $a[i][n-1-i]$ với $0 \leq i < n$



- c. Các phần tử nằm trong nửa mảng vuông phía trên đường chéo chính là $a[i][j]$ với $i \leq j$



- d. Các phần tử nằm trong nửa mảng vuông phía dưới đường chéo chính là $a[i][j]$ với $i \geq j$



- e. Các phần tử nằm trong nửa mảng vuông phía trên đường chéo phụ là $a[i][j]$ với $i + j \leq n - 1$



- f. Các phần tử nằm trong nửa mảng vuông phía dưới đường chéo phụ là $a[i][j]$ với $i + j \geq n - 1$

• **Duyệt mảng vuông:**

- a. Duyệt các phần tử nằm trên ĐCC



```
for(int i = 0; i < n; i++)
    <Xử lý a[i][i]>
```

- b. Duyệt các phần tử nằm trên ĐCP



```
for(int i = 0; i < n; i++)
    <Xử lý a[i][n - 1 - i]>
```

- c. Duyệt các phần tử nằm trong nửa mảng vuông phía trên ĐCC



```
for(int i = 0; i < n; i++)
    for(int j = 0; j < n; j++)
        if(i <= j)
        {
            <Xử lý a[i][j]>
        }
```

d. Duyệt các phần tử nằm trong nửa mảng vuông phía dưới ĐCC

```
for(int i = 0; i < n; i++)
    for(int j = 0; j < n; j++)
        if(i >= j)
        {
            <Xử lý a[i][j]>
        }
```

e. Duyệt các phần tử nằm trong nửa mảng vuông phía trên ĐCP

```
for(int i = 0; i < n; i++)
    for(int j = 0; j < n; j++)
        if(i + j <= n - 1)
            <Xử lý a[i][j]>
```

f. Duyệt các phần tử nằm trong nửa mảng vuông phía dưới ĐCP

```
for(int i = 0; i < n; i++)
    for(int j = 0; j < n; j++)
        if(i + j >= n - 1)
            <Xử lý a[i][j]>
```

VD: Viết CT nhập / xuất ma trận vuông, tìm phần tử lớn nhất trên đường chéo chính và kiểm tra ma trận vuông có tồn tại phần tử âm ở nửa dưới đường chéo phụ không?

```
#include <stdio.h>
#include <conio.h>

#define Cap_n 10

void NhapMangVuong(int a[][Cap_n], int &n)
{
    do {
        printf("Nhap cap ma tran: ");
        scanf("%d", &n);
    } while(n < 1 || n > Cap_n);
    for(int i = 0; i < n; i++)
        for(int j = 0; j < n; j++)
        {
            printf("a[%d][%d]=", i, j);
            scanf("%d", &a[i][j]);
        }
}
```



```
void XuatMangVuong(int a[][Cap_n],int n)
{
    for(int i = 0; i < n; i++)
    {
        for(int j = 0; j < n; j++)
            printf("%3d", a[i][j]);
        printf("\n");
    }
}

void TimMaxDCC(int a[][Cap_n],int n)
{
    int max=a[0][0], vtmax=0;
    for(int i=1;i<n;i++)
        if(a[i][i]>max)
        {
            max=a[i][i];
            vtmax=i;
        }
    printf("Phan tu lon nhat tren DCC la a[%d][%d] = %d\n",vtmax,vtmax,max);
}

int KiemTraTonTaiPhanTuAmDuoiDCP(int a[][Cap_n],int n)
{
    int am=0;
    for(int i = 0; i < n; i++)
        for(int j = 0; j < n; j++)
            if(i + j >= n - 1)
                if(a[i][j]<0)
                {
                    am=1;
                    break;
                }
    return am;
}

void main()
{
    int a[Cap_n][Cap_n],n;
    NhapMangVuong(a,n);
    XuatMangVuong(a,n);
    TimMaxDCC(a,n);
    if(KiemTraTonTaiPhanTuAmDuoiDCP(a,n)==1)
        printf("Ton tai phan tu am o nua duoi DCP\n");
    else
        printf("Khong ton tai phan tu am o nua duoi DCP\n");
    _getch();
}
```