# CHƯƠNG 6 – MẢNG

BUỔI 10: TÌM HIỂU MẢNG 2 CHIỀU VÀ CHUỖI KÝ TỰ

### Nội dung



- 1. Giới thiệu về mảng
- 2. Khái niệm mảng
- 3. Các yếu tố xác định mảng
- 4. Mảng 1 chiều
- 5. Các tác vụ trên mảng 1 chiều
- 6. Mảng 2 chiều
- 7. Các tác vụ trên mảng 2 chiều
- 8. Chuỗi ký tự
- 9. Các tác vụ trên chuỗi ký tự

## 6. Mảng 2 chiều



- 6.1. Khai báo mảng 2 chiều
- 6.2. Chỉ số mảng và truy xuất phần tử mảng
- 6.3. Lấy địa chỉ các phần tử mảng
- 6.4. Một số khái niệm liên quan: đường chéo chính, đường chéo phu, nửa trên/nửa dưới đường chéo chính, ...
- 6.5. Truyền mảng cho hàm và lời gọi hàm

### 6.1. Khai báo mảng 2 chiều



#### Cú pháp:

<Kiểu dữ liệu> <Tên biến mảng>[<Số Dòng>][<Số Cột>];

#### Trong đó:

Kiểu dữ liệu: int, float, char

Tên biến mảng: 1 ký tự hoặc 1 dãy ký tự viết liền nhau và không có khoảng trắng

Dòng, Cột: số lượng các phần tử mỗi chiều của mảng

#### char A[10][20]

Kiểu dữ liệu: char

Tên biến mảng: A

Mảng có 10 dòng và 20 cột

#### int Mang2Chieu[3][5]

Kiểu dữ liệu: int

Tên biến mảng: Mang2Chieu

Mảng có 3 dòng và 5 cột

## 6.1. Khai báo mảng 2 chiều





## 6.2. Chỉ số mảng 2 chiều



- · Chỉ số mảng là một giá trị số nguyên int.
- · Chỉ số trong mảng 2 chiều gồm chỉ số dòng và chỉ số cột.
  - 0 ≤ chỉ số dòng ≤ số dòng của mảng 1
  - 0 ≤ chỉ số cột ≤ số cột của mảng 1

		0	1	2
int A[2][3];	0	2	45	7
Tên mảng: A	1	73	11	187

Kiểu dữ liệu của từng phần tử trong mảng: int

Số phần tử tối đa trong mảng: 2\*3=6 phần tử

Các chỉ số được đánh số: Chỉ số dòng: 0, 1

Chỉ số Cột: 0, 1, 2

## 6.2. Truy xuất phần tử mảng



• Truy xuất phần tử mảng thông qua chỉ số

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

0 1 2 29 137 50 0 3 78 943 1

Các truy xuất hợp lệ: A[0][0], A[0][1],..., A[1][2], A[1][3]

Các truy xuất không hợp lệ: A[-1][0], A[1][4], A[2][0]

Giá trị các phần tử mảng:

A[0][0]=29, A[0][1]=137, A[0][2]=50

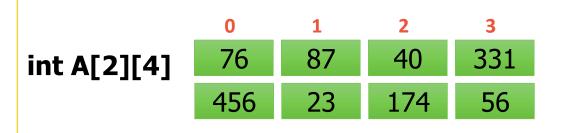
A[1][0]=3, A[1][1]=78, A[1][2]=943

## 6.3. Lấy địa chỉ các phần tử mảng



· Cú pháp:

&<Tên biến mảng>[<Chỉ số dòng>][<Chỉ số cột>];



#### Địa chỉ các phần tử mãng 2 chiều:

Địa chỉ các phần tử trên dòng thứ 0:

&A[0][0], &A[0][1], &A[0][2], &A[0][3]

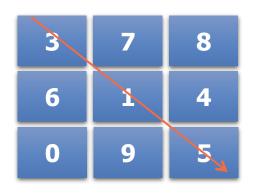
Địa chỉ các phần tử trên dòng thứ 1:

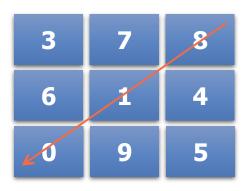
&A[1][0], &A[1][1], &A[1][2], &A[1][3]

## 6.4. Một số khái niệm liên quan



Cho ma trận A gồm 3 dòng x 3 cột như hình dưới đây:





- · Các phần tử nằm trên đường chéo chính là {3,1,5}
- · Các phần tử nằm trên đường chéo phụ là {8,1,0}
- Các phần tử nằm nửa trên đường chéo chính là {3,7,8,1,4,5}
- Các phần tử nằm nửa dưới đường chéo chính là {3,6,1,0,9,5}

## 6.5. Truyền mảng cho hàm và lời gọi hàm



 Tham số kiểu mảng trong khai báo hàm giống như khai báo biến mảng.

```
int TinhDCheo(int A[50][50], int n, int m);
Tên hàm: TinhDCheo
Tham số: kiểu mảng số nguyên A và số lượng dòng n, số lượng cột
m
```

```
void XuatMang(int A[50][50], int n, int m);
Tên hàm: XuatMang
Tham số: kiểu mảng số nguyên A và số lượng dòng n, số lượng cột
m
```

Giá trị trả về: Không có kiểu trả về void

## 6.5. Truyền mảng cho hàm và lời gọi hàm



- · Mảng có thể thay đổi nội dung sau khi thực hiện hàm.
- · Có thể bỏ số lượng phần tử hoặc sử dụng con trỏ.

```
void NhapMang(int A[][50] , int n, int m);
void NhapMang(int (*A)[50], int n, int m);
```

## 6.5. Truyền mảng cho hàm và lời gọi hàm



```
#include <stdio.h>
#include <conio.h>
void nhap(int A[][100], int &N, int &M)
void xuat(int A[][100], int N , int M)
void SapXep(int A[][100], int N , int M)
void main()
   int a[100],n,m;
   nhap(a,n,m);
   xuat(a,n,m);
   SapXep (a,n,m);
```

## 7. Các tác vụ trên mảng 1 chiều



- 7.1. Nhập mảng
- 7.2. Xuất mảng
- 7.3. Tìm kiếm một phần tử trong mảng
- 7.4. Kiểm tra tính chất của mảng
- 7.5. Đếm số lượng các phần tử trong mảng
- 7.6. Tính tổng các phần tử có giá trị chẵn trong mảng
- 7.7. Tính Tổng giá trị các phần tử trên đường chéo chính

### 7.1. Nhập mảng



### Yêu cầu: nhập mảng A gồm m dòng và n cột

```
void NhapMaTran(int A[][MAXC], int &m, int &n)
{
    cout << "Nhap so dong, so cot cua ma tran: ";
    cin >> m;
    cin >> n;
    int i, j;
    for (i = 0; i < m; i++)
        for (j = 0; j < n; j++)
            cout << "Nhap A[" << i<< "]" << "["<< j;
            cin >> A[i][j];
```

## 7.2. Xuất mảng



### Yêu cầu: xuất mảng A gồm m dòng và n cột

```
void XuatMaTran(int A[][MAXC], int m, int n)
    int i, j;
    for (i = 0; i < m; i++)
        for (j = 0; j < n; j++)
            cout << A[i][j];
        cout << "\n";
```

## 7.3. Tìm kiếm 1 phần tử trong mảng



**Yêu cầu:** Tìm xem phần tử x có nằm trong ma trận a kích thước mxn hay không?

```
int TimKiem(int a[][MAXC], int m, int n, int x)
{
    int i, j;
    for (i = 0; i < m; i++)
        for (j = 0; j < n; j++)
                 if (a[i][j] == x)
                          return 1;
    return 0;
```



#### ਲ Yêu cầu

Cho trước ma trận a kích thước mxn. Ma trận a có phải là ma trậntoàn các số chẵn hay không?

#### **ω Ý tưởng**

YT 1: Đếm số lượng số chẵn của ma trận. Nếu số lượng này bằng đúng mxn thì ma trận toàn chẵn.

YT 2: Đếm số lượng số không phải chẵn của ma trận.

Nếu số lượng này bằng 0 thì ma trận toàn chẵn.

YT 3: Tìm xem có phần tử nào không phải số chẵn không.

Nếu có thì ma trận không toàn số chẵn.



```
int KiemTra_YT1(int a[][MAXC], int m, int n)
   int i, j, dem = 0;
   for (i = 0; i < m; i++)
      for (j = 0; j < n; j++)
             if (LaSNT(a[i][j] == 1)
              dem++;
   if (dem == m * n)
       return 1;
   return 0;
```



```
int KiemTra_YT2(int a[][MAXC], int m, int n)
   int i, j, dem = 0;
   for (i = 0; i < m; i++)
      for (j = 0; j < n; j++)
            if (LaSNT(a[i][j] == 0)
              dem++;
   if (dem == 0)
       return 1;
   return 0;
```



```
int KiemTra_YT3(int a[][MAXC], int m, int n)
   int i, j, dem = 0;
   for (i = 0; i < m; i++)
       for (j = 0; j < n; j++)
          if (LaSNT(a[i][j] == 0)
              return 0;
   return 1;
```

## 7.5. Đếm số lượng các phần tử trong mảng



```
int Dem(int A[][MAXC], int N, int M)
   int Dem=0;
   for (i = 0; i < m; i++)
      for (j = 0; j < n; j++)
          Dem++;
   return Dem;
```





```
int TongChan(int A[][MAXC], int N, int M)
   int TC=0;
   for (i = 0; i < m; i++)
      for (j = 0; j < n; j++)
          if(A[i][j]\%2==0)
             TC=TC+A[i][j];
   return TC;
```

## 7.7. Tính Tổng gtrị các ptử trên đchéo chính (



```
int TongDCChinh(int a[][MAXC], int m, int n)
       int i, tong;
       tong = 0;
       for (i = 0; i < n; i++)
              tong = tong + a[i][i];
       return tong;
```

## **BÀI TẬP**



- Nhập mảng / Xuất mảng
- Tìm kiếm một phần tử trong mảng
- · Kiểm tra mảng có đối xứng qua đường chéo chính hay không?
- Tính tổng các phần tử trên dòng/cột/toàn mảng/đường chéo chính/nửa trên/nửa dưới
- · Tìm giá trị nhỏ nhất/lớn nhất của mảng
- Tính tống 2 ma trận (mảng được xem là ma trận)
- Tính tích 2 ma trận (mảng được xem là ma trận)
- Kiểm tra ma trận có phải là ma trận đơn vị không? (mảng được xem là ma trận)

## 8. Chuỗi ký tự



- 8.1. Khái niệm
- 8.2. Khai báo, khởi tạo
- 8.3. Nhập xuất chuỗi
- 8.4. Một số hàm thông dụng trong thư viện

## 8.1. Khái niệm chuỗi ký tự



- Kiểu char chỉ chứa được một ký tự. Để lưu trữ một chuỗi (nhiều ký tự) ta sử dụng mảng (một chiều) các ký tự.
- Chuỗi ký tự kết thúc bằng ký tự "\0" (null)
- Độ dài chuỗi = kích thước mảng 1

char Hoten[30]; // Dài 29 ký tự char NgaySinh[9]; // Dài 8 ký tự

## 8.2. Khai báo chuỗi ký tự



#### Các kiểu khai báo chuỗi

- char sName[100];
- char sName[];
- char \*sName;

### 8.2. Khởi tạo chuỗi ký tự



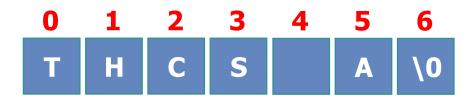
Khởi tạo như mảng thông thường

• Độ dài cụ thể

```
char s[10] = {\T', \H', \C', \S', \', \A', \0'};
char s[10] = \THCS A''; // Tự động thêm \0'
```



· Tự xác định độ dài



## 8.3. Nhập xuất chuỗi



Hàm nhập chuỗi: gets
 Ví dụ: gets(hoten);
 Hàm tự động thêm ký tự NULL (`\0') vào cuối biến chuỗi.

```
void nhapchuoi(char s[100])
{
    cout << "Nhap chuoi";
    gets(s); // hàm nhập chuỗi
}</pre>
```

## 8.3. Nhập xuất chuỗi



```
    Hàm xuất chuỗi: puts

Ví du: puts(hoten);
 void Xuatchuoi(char s[100])
      cout << "Xuat chuoi";</pre>
      puts(s); // hàm xuất chuỗi
```

## 8.4. Một số hàm thông dụng trong thư viện



Một số hàm thuộc thư viện <string.h>

- strlen: hàm tính độ dài chuỗi ký tự
- strcpy: hàm sao chép chuỗi ký tự
- strdup: hàm tạo bản sao
- strlwr/strupr: hàm chuyển chuỗi thành chuỗi viết thường / hoa
- strrev: hàm đảo ngược
- strcmp: hàm so sánh 2 chuỗi có phân biệt hoa thường
- stricmp: hàm so sánh 2 chuỗi không phân biệt hoa thường
- strcat : hàm nối 2 chuỗi
- strstr: hàm tìm chuỗi trong chuỗi

### 9. Các thao tác trên chuỗi ký tự



- 9.1. Đếm các ký tự khoảng trắng trong chuỗi ký tự
- 9.2. Đếm các ký tự hoa / thường trong chuỗi ký tự
- 9.3. Đổi các từ ở đầu câu sang chữ hoa và những từ không phải đầu câu sang chữ thường.
- 9.4. Chuyển các ký tự viết hoa thành viết thường
- 9.5. Chuyển các ký tự viết thường thành viết hoa
- 9.6. Liệt kê các từ trong chuỗi
- 9.7. Xóa các khoảng trắng đầu chuỗi / cuối chuỗi

## 9.1. Đếm các ký tự khoảng trắng



```
void DemKT(char chuoi[100])
   int i;
   int dem=0;
   for (i=0; i<strlen(chuoi); i++)
          if (chuoi[i]==' ')
               dem++;
   return dem;
```





```
void DemKTThuong(char chuoi[])
   int i, dt=0, dh=0;
   for(i=0; i<strlen(chuoi); i++)</pre>
          if((chuoi[i]>='a')&&(chuoi[i]<='z'))
               dt++;
          else if ((chuoi[i]>='A')&&(chuoi[i]<='Z'))
                dh++;
     cout << "So ky tu thuong:" << dt;
     cout << "So ky tu hoa:" << dh;
```

## 9.3. Đổi hoa – thường



```
void DoiHoaThuong(char chuoi[100])
   chuoi[0]=toupper(chuoi[0]);
   for(int i=1; i < strlen(chuoi); i++)
          chuoi[i]=tolower(chuoi[i]);
   cout << "Xuat chuoi";
   puts(chuoi);
```

# 9.4. Chuyển các ký tự viết hoa thành viết thg

```
void ChuyenHoaSangThuong(char chuoi[100])
   char kq[100];
   strcpy(kq, chuoi);
   for(int i=0; kq[i]!='\0'; i++)
      if ((kq[i] > = 'A') \&\& (kq[i] < = 'Z'))
          kq[i]=tolower(kq[i]);
   cout << "Xuat chuoi";
    puts(kq);
```

# 9.5. Chuyển các ký tự viết thg thành viết hoa

```
void ChuyenThuongSangHoa(char chuoi[100])
   char kq[100];
   strcpy(kq, chuoi);
   for(int i=0; kq[i]!='\0'; i++)
       if ((kq[i] > = 'a') \&\& (kq[i] < = 'z'))
          kq[i]=toupper(kq[i]);
   cout << "Xuat chuoi";
   puts(kq);
```





```
void LietKe (char chuoi[100])
     int d=0;
     for(i=0; i<strlen(chuoi); i++)</pre>
           if(chuoi[i]==' ')
                for(j=d; j<i; j++)
                      cout << chuoi[j];</pre>
                d=i+1;
                cout << "\n";
```





```
void xoadau (char chuoi[100])
     int i=0
     while (chuoi[0]==')
         for(int i = 0; i < strlen(chuoi); ++i)
               str[i] = str[i + 1];
void xoacuoi (char chuoi[100])
     while (chuoi[strlen(chuoi)]==' ')
          chuoi[strlen(chuoi)]='\0';
```

## **BÀI TẬP**



- · Nhập / xuất chuỗi
- Xuất các ký tự in hoa trong chuỗi
- · Đảo ngược các kí tự trong chuỗi.
- · Đổi chữ xen kẻ 1 chữ hoa và 1 chữ thường.
- · Đếm một ký tự xuất hiện bao nhiều lần trong chuỗi.
- · Tìm kiếm xem ký tự nào xuất nhiện nhiều nhất trong chuỗi.
- Kiểm tra xem chuỗi có đối xứng hay không?
- Nhập vào một từ và xoá từ đó trong chuỗi đã cho.