CHƯƠNG 6 – MẢNG

TÌM HIỂU VỀ MẢNG – MẢNG 1 CHIỀU

CĐR buổi học

- Sau khi học xong buổi học, sinh viên có khả năng:
  - Hiểu được khái niệm cơ bản mảng, mảng một chiều và cách tổ chức lưu trữ các phần tử trong mảng.
  - Giải thích và sử dụng được một số thao tác cơ bản trên mảng một chiều.
  - Viết chương trình sử dụng mảng một chiều

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

1. Giới thiệu

- Chương trình cần lưu trữ 3 số thực

→ Khai báo 3 biến kiểu số thực : float a, b, c;

- Chương trình cần lưu trữ 10 hoặc 100 hoặc 1000 số thực

→ Khai báo 10 hoặc 100 hoặc 1000 biến kiểu số thực?

→ Không thực hiện được

1.3	9.4	2.7	6.2	4.9	7.7	3.5	8.6	0.1	5.4

→ Cần có 1 kiểu dữ liệu mới để có thể lưu trữ dãy số thực này và truy xuất dễ dàng → **MẢNG**

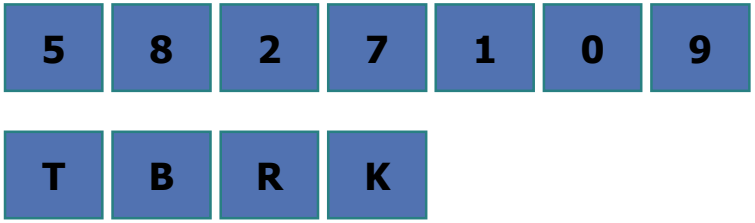
2. Khái niệm mảng

- Biểu diễn một dãy các phần tử có cùng kiểu và mỗi phần tử trong mảng biểu diễn 1 giá trị.
- Kích thước mảng được xác định ngay khi khai báo và không thay đổi.
- Một kiểu dữ liệu có cấu trúc do người lập trình định nghĩa.
- Ngôn ngữ lập trình C luôn chỉ định một khối nhớ liên tục cho một biến kiểu mảng.

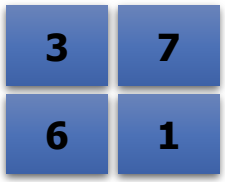
Ví dụ: dãy các số nguyên, dãy các ký tự…

2. Khái niệm mảng

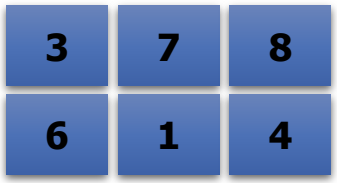
- Mảng 1 chiều gồm 1 dãy các phần tử có cùng kiểu dữ liệu (int, float, char …)



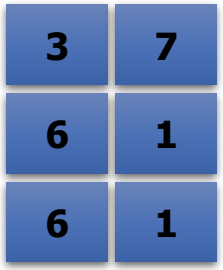
- Mảng 2 chiều (Ma trận) gồm các phần tử trên dòng và các phần tử trên cột



Ma trận dòng = cột = 2



Ma trận dòng < cột
Dòng = 2 , cột =3

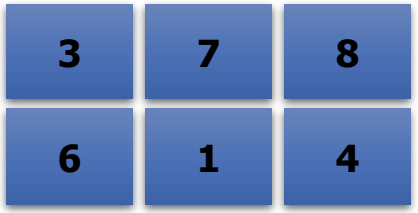


Ma trận dòng > cột
Dòng = 3 , cột =2

3. Các yếu tố xác định mảng

- Tên mảng: MangKyTu
- Kiểu mảng: char
- Số chiều: 1 chiều
- Kích thước: 4 phần tử

- Tên mảng: MangSoNguyen
- Kiểu mảng: int
- Số chiều: 2 chiều
- Kích thước: 2 cột x 3 dòng



4. Mảng 1 chiều

4.1. Khai báo và khởi tạo mảng 1 chiều

4.2. Chỉ số mảng và truy xuất phần tử mảng

4.3. Lấy địa chỉ các phần tử mảng

4.4. Truyền mảng cho hàm và lời gọi hàm

4.1. Khai báo mảng 1 chiều

- Cú pháp:

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

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

Số phần tử mảng: số lượng các phần tử của mảng 1 chiều

char A[10]

Kiểu dữ liệu: char

Tên biến mảng: A

Số phần tử mảng: 10 phần tử

int Mang1Chieu[30]

Kiểu dữ liệu: int

Tên biến mảng: Mang1Chieu

Số phần tử mảng: 30 phần tử

4.1. Khai báo mảng 1 chiều

- Phải xác định cụ thể <số phần tử mảng> ngay lúc khai báo, không được sử dụng biến hoặc hằng thường.

```
int n1 = 10; int a[n1];

const int n2 = 20; int b[n2];
```

- Nên sử dụng chỉ thị tiền xử lý #define để định nghĩa số phần tử mảng

```
#define n1 10

#define n2 20

int a[n1];          //  int a[10];

int b[n1][n2];      //  int b[10][20];
```

4.1. Khởi tạo mảng 1 chiều

- Khởi tạo giá trị cho mọi phần tử của mảng

  int A[4] = {29, 137, 50, 4};

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

  int B[4] = {91, 106};

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

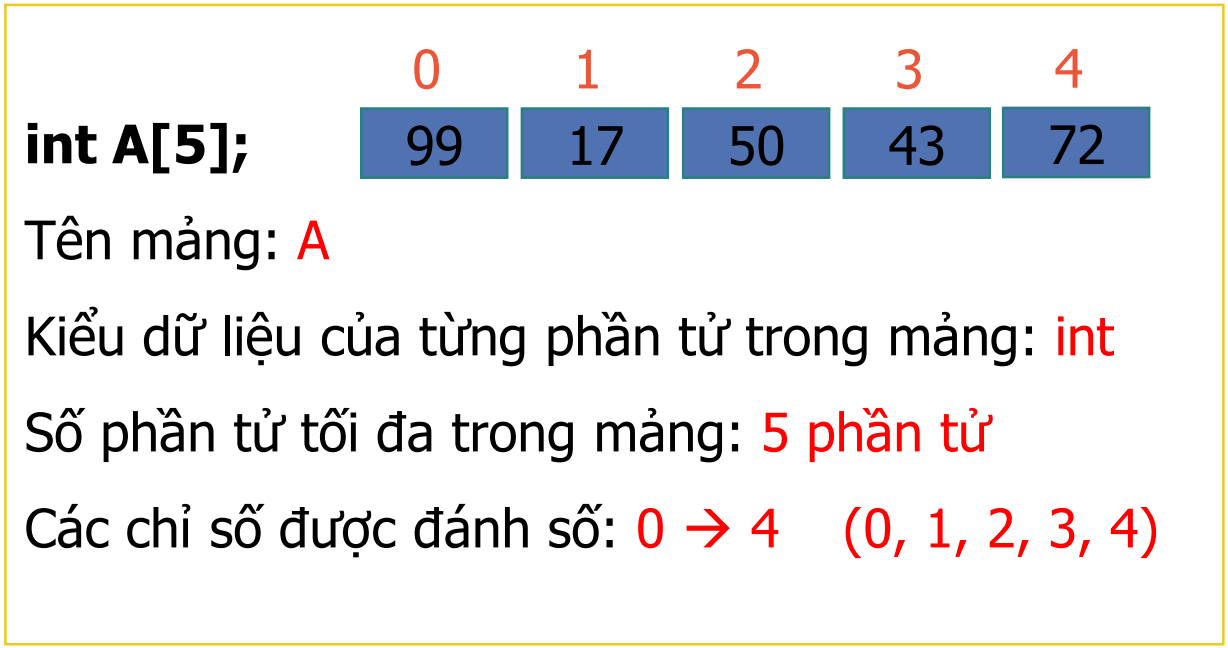
  int a[4] = {0};

- Tự động xác định số lượng phần tử

  int a[] = {22, 16, 56, 19};

4.2. Chỉ số mảng

- Chỉ số mảng (vị trí trong mảng) là một giá trị số nguyên int.
- Chỉ số bắt đầu là 0 và không vượt quá số lượng phần tử tối đa trong mảng.
- Số lượng các chỉ số mảng = số lượng phần tử tối đa trong mảng



4.2. Truy xuất phần tử mảng

- Truy xuất phần tử mảng thông qua chỉ số

<Tên biến mảng>[<chỉ số mảng>]

- Các phần tử mảng là 1 dãy liên tục có chỉ số từ 0 đến <Số phần tử mảng>-1

int A[4]

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

Các truy xuất không hợp lệ: A[-1], A[4], A[5]

Giá trị các phần tử mảng A[0]=29, A[1]=137, A[2]=50, A[3]=4

4.3. Lấy địa chỉ các phần tử mảng

- Cú pháp:

&<Tên biến mảng>[<chỉ số mảng>];

int A[4]

0	1	2	3
29	137	50	4

Địa chỉ các phần tử mạng:

Địa chỉ phần tử thứ 0: &A[0]

Địa chỉ phần tử thứ 1: &A[1]

Địa chỉ phần tử thứ 2: &A[2]

Địa chỉ phần tử thứ 3: &A[3]

4.4. Truyền mảng cho hàm và lời gọi hàm

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

```
void SapXep(int A[100], int n);
```

Tên hàm: SapXep

Tham số: kiểu mảng số nguyên A và số lượng phần tử mảng n

Giá trị trả về: không có giá trị trả về void

```
int TinhTong(int A[100], int n);
```

Tên hàm: TinhTong

Tham số: kiểu mảng số nguyên A và số lượng phần tử mảng n

Giá trị trả về: kiểu số nguyên int

4.4. 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[], int n);
void NhapMang(int *A, int n);
```

4.4. Truyền mảng cho hàm và lời gọi hàm

```
#include <stdio.h>
#include <conio.h>
void Nhap(int A[], int &N)     // nhập mảng
void Xuat(int A[], int N)      // xuất mảng
int TinhTong(int A[], int N)   // tính tổng các phần tử trong mảng
void main()
{
    int a[100], n, S;
    Nhap(a,n);
    Xuat(a,n);
    S=TinhTong (a,n);
    cout<< "Tong cac phan tu trong mang :" << S;
}
```

5. Các tác vụ trên mảng 1 chiều

5.1. Nhập mảng

5.2. Xuất mảng

5.3. Tìm kiếm một phần tử trong mảng

5.4. Kiểm tra tính chất của mảng

5.5. Đếm số lượng các phần tử có giá trị chẵn trong mảng

5.6. Tính tổng các phần tử có giá trị chẵn trong mảng

5.7. Tách mảng / Gộp mảng

5.8. Tìm giá trị nhỏ nhất/lớn nhất của mảng

5.9. Sắp xếp mảng giảm dần/tăng dần

5.10. Thêm/Xóa/Sửa một phần tử vào mảng

5.1. Nhập mảng

Yêu cầu: Cho phép nhập mảng a, số lượng phần tử n

```
void nhapmang(int A[], int N)
{
    for(int i=0; i<N; i++)
    {
        cout << "Nhap phan tu thu %d: " <<  i;
        cin >> A[i];
    }
}
```

5.2. Xuất mảng

Yêu cầu: Cho trước mảng a, số lượng phần tử n. Hãy xuất nội dung mảng a ra màn hình.

```
void xuatmang(int A[], int N)
{
    for(int i=0; i<N; i++)
        cout << A[i];
}
```

5.3. Tìm kiếm 1 phần tử trong mảng

Yêu cầu: Tìm xem phần tử x có nằm trong mảng a kích thước n hay không? Nếu có thì xuất ra màn hình vị trí đầu tiên tìm thấy được.

```
int TimKiem(int a[], int n, int x)
{
        for (int vt = 0; vt < n; vt++)
                if (a[vt] == x)
                        return vt;
        return -1;
}
```

5.3. Tìm kiếm 1 phần tử trong mảng

Yêu cầu: Tìm xem phần tử x có nằm trong mảng a kích thước n hay không? Nếu có thì xuất ra màn hình vị trí đầu tiên tìm thấy được.

```
int Max(int a[], int n)
{
        int Max=a[0];
        for (int i= 0; i< n; i++)
                if (Max < a[i])  // áp dụng kỹ thuật lính canh
                        Max=a[i];
        return Max;
}
```

5.4. Kiểm tra tính chất của mảng

- **Yêu cầu**

Cho trước mảng a, số lượng phần tử n. Mảng a có phải là mảng toàn các số nguyên tố hay không?

- **Ý tưởng**

YT 1: Đếm số lượng số ngtố của mảng. Nếu số lượng này bằng đúng n thì mảng toàn ngtố.

YT 2: Đếm số lượng số không phải ngtố của mảng. Nếu số lượng này bằng 0 thì mảng toàn ngtố.

YT 3: Tìm xem có phần tử nào không phải số ngtố không. Nếu có thì mảng không toàn số ngtố.

5.4. Kiểm tra tính chất của mảng

Hàm kiểm tra SNT

```
int LaSNT(int n)
{
    int i, flag = 0;
    for (i = 2; i < n; i++)
        if (n % i == 0)
                flag=1;  // áp dụng kỹ thuật cờ hiệu
    if (flag == 0)
        return 1; // SNT
    return 0;       // Không phải SNT
}
```

5.4. Kiểm tra tính chất của mảng

```
int KiemTra_YT1(int a[], int n)
{
	int dem = 0;
	for (int i = 0; i < n; i++)
		if (LaSNT(a[i]) == 1)
			dem++;
	if (dem == n)
		return 1;
	return 0;
}

int KiemTra_YT3(int a[], int n)
{
	for (int i = 0; i < n ; i++)
		if (LaSNT(a[i]) == 0)
			return 0;
		return 1;
}
```

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

5.5. Đếm số lượng các ptử chẵn trong mảng

```
int DemChan(int A[], int N)
{
    int DC=0;
    for(int i=0;i<n;i++)
        if(A[i]%2==0)
            DC++;
    return DC;
}
```

```
#include <stdio.h>
#include <conio.h>
void nhapmang(int A[], int &N)
void xuatmang(int A[], int N)
void DemChan(int A[], int N)
void main()
{
    int a[100],n,DC;
    nhap(a,n);
    xuat(a,n);
    DC=DemChan (a,n);
    printf("So ptu chan là %d", DC);
}
```

5.6. Tính tổng các phần tử có giá trị chẵn

```
int TongChan(int A[], int N)
{
    int TC=0;
    for(int i=0;i<n;i++)
            if(A[i]%2==0)
                TC=TC+A[i];
    return TC;
}
```

```
#include <stdio.h>
#include <conio.h>
void nhapmang(int A[], int &N)
void xuatmang(int A[], int N)
void TongChan(int A[], int N)
void main()
{
    int a[100],n,TC;
    nhap(a,n);
    xuat(a,n);
    TC=TongChan (a,n);
    printf("Tong ptu chan là %d", TC);
}
```

Bài tập bắt buộc

1. Viết chương trình nhập vào một dãy tăng dần, không cần sắp xếp. Nếu nhập sai yêu cầu sẽ phải nhập lại và xuất các số nguyên tố có trong mảng.
2. Kiểm tra mảng có đối xứng hay không?
3. Liệt kê các giá trị xuất hiện trong mảng đúng 1 lần.
4. Tìm vị trí của phần tử có giá trị âm lớn nhất trong mảng số nguyên.
5. Viết hàm xóa phần tử có chỉ số k trong mảng số nguyên a có n phần tử. Nếu giá trị của k<0 hoặc k>=n thì không xóa và hàm trả về giá trị 0. Ngược lại ta xóa giá trị phần tử a[k] và hàm trả về giá trị 1.