TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN TP.HCM
KHOA CÔNG NGHỆ THÔNG TIN
MÔN: **NHẬP MÔN LẬP TRÌNH**

HƯỚNG DẪN THỰC HÀNH

TUẦN 06

MẢNG MỘT CHIỀU

TP.HCM, ngày 09 tháng 09 năm 2017

MỤC LỤC

1 Khái niệm

Trong phần này ta sẽ tìm hiểu các vấn đề liên quan tới mảng một chiều. Như đã đề cập trong các phần trước, ngôn ngữ lập trình thường cài đặt sẵn cho ta một vài kiểu dữ liệu cơ bản để hỗ trợ trong quá trình chuyển một bài toán thực tế vào máy tính. Ví dụ để biểu diễn số nguyên trong thực tế, ta có kiểu int, hay chuyển số thực thì có kiểu float…Hoặc để phân rã công việc lớn ra thành các tác vụ nhỏ hơn thì có khái niệm hàm. Mảng một chiều cũng ra đời để hỗ trợ trong công việc thực tế. Ví dụ trong lớp học có 100 học sinh, ta cần chương trình để quản lí 100 học sinh này. Chưa kể tới việc xử lý công việc liên quan, việc nhập thông tin của 100 học sinh này vào máy tính đã là một công việc mệt mỏi. Nếu không có khái niệm mảng một chiều thì ví dụ để lưu phẩy trung bình của 100 học sinh này ta cũng cần khai báo 100 biến số thực rời rạc với 100 tên khác nhau (Chưa kể tới việc lưu các thông tin khác như ngày sinh, các điểm thành phần…). Khái niệm mảng giúp ta xử lý các tình huống mang tính chất xử lý tập hợp dữ liệu lớn như thế.

2 Các kĩ thuật cơ bản trên mảng một chiều

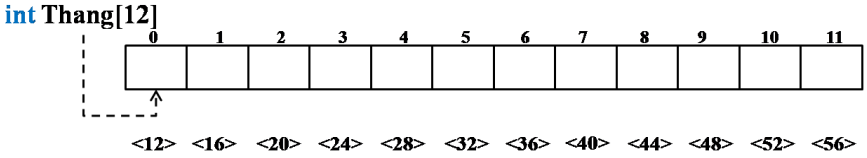
Trong phần này ta sẽ lần lượt giới thiệu các kĩ thuật cần biết khi sử dụng mảng một chiều trong ngôn ngữ C.

2.1 Khai báo và khởi gán mảng một chiều

Để khai báo mảng một chiều ta dùng cú pháp: <Kiểu dữ liệu> Tên_mảng[<kích thước>]. Ví dụ giả sử khai báo một mảng số nguyên có tên là Thang gồm 12 phần tử là

```
int Thang[12];
```

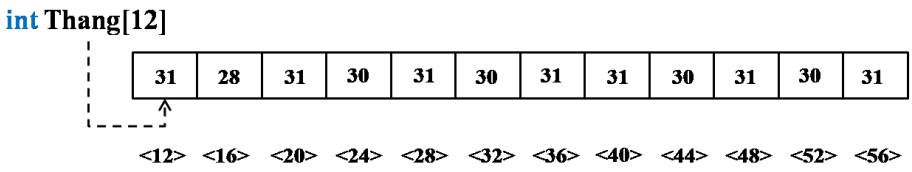
Với khai báo như trên thì ta có thể hiểu trong RAM có một vùng nhớ **liên tục** gồm 12 phần tử, mỗi phần tử có kích thước là 4 byte (kích thước tùy ngôn ngữ lập trình quyết định).



Cũng như khai báo biến, thực sự chưa có giá trị nào. Ta cần khởi gán (định nghĩa) các giá trị cho mảng. Ví dụ ta gán như sau:

```
int Thang[12] = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
```

Như vậy ta vừa định nghĩa giá trị cho mảng Thang (giá trị này do ta gán, không phải do người dùng nhập thông qua scanf).



2.2 Nhập xuất mảng một chiều

Tiếp theo ta sẽ học cách nhập xuất mảng một chiều. Kĩ thuật khởi gán giá trị như trên cũng có thể xem là nhập dữ liệu cho mảng một chiều, tuy nhiên đó là ta khởi gán cứng cho mảng. Trong phần này ta sẽ học cách cho người dùng nhập giá trị thông qua bàn phím bằng hàm scanf cũng như có thể xuất dữ liệu mảng ra màn hình console (màn hình đen). Xét đoạn mã sau minh họa về cách nhập xuất mảng số nguyên từ bàn phím.

Dòng	
1	#include <stdio.h>
2	#define N 100
3	void NhapMangSoNguyen(int[], int&);
4	void XuatMangSoNguyen(int[], int);
5	
6	void NhapMangSoNguyen(int a[], int &n){
7	printf(“Nhap so luong phan tu: ”);
8	scanf(“%d”, &n);
9	for(int i = 0; i < n; i++){
10	printf(“Nhap a[%d]: ”, i);
11	scanf(“%d”, &a[i]);
12	}
13	}
14	void XuatMangSoNguyen(int a[], int n){
15	printf(“Mang co %d phan tu: ”, n);
16	for(int i = 0; i < n; i++){
17	printf(“a[%d] = %d\n”, i, a[i]);
18	}
19	}
20	
21	void main(){
22	int b[N], m;
23	NhapMangSoNguyen(b, m);
24	XuatMangSoNguyen(b, m);
25	}

Trong ví dụ này ta thấy có các vấn đề cần lưu ý. Thứ nhất chỉ số mảng bắt đầu từ zero, như vậy ví dụ mảng 12 phần tử thì phần tử đầu có chỉ số là 0 và phần tử cuối có chỉ số là 11. Thứ hai trong hàm main ta thấy hằng số N là 100, như vậy ta đã khai báo mảng có tên là b chứa 100 phần tử rỗng chưa có giá trị, sau đó khi đi vào hàm 'NhapMangSoNguyen' thì ta mới quyết định sử dụng m phần tử (dĩ nhiên m < 100). Vậy nếu ta nhập m = 10 thì có nghĩa 90 phần tử còn lại sẽ bị dư ra và đây cũng chính là hạn chế của **mảng một chiều tĩnh**. Điều lưu ý cuối cùng là cách truyền tham số kiểu mảng, ta thấy trong hàm main khi gọi hàm 'NhapMangSoNguyen' thì vị trí tham số mảng ta chỉ truyền tên biến là 'b' và không ghi thêm kí hiệu '&' để thể hiện mong muốn thay đổi giá trị cho b. Lí do là trong C thì biến kiểu mảng đã là một tham chiếu nên không cần dùng kí hiệu '&'. Còn vị trí biến 'm' thì ta vẫn dùng bình thường để thể hiện đây là tham chiếu nhằm muốn hàm 'NhapMangSoNguyen'thay đổi giá trị cho m khi truyền vào nó.

3 Một số kĩ thuật khác

Ngoài việc khai báo, định nghĩa và nhập/xuất mảng, ta có rất nhiều thao tác liên quan tới mảng một chiều. Phần này ta sẽ tìm hiểu hai thao tác cơ bản là tìm kiếm và sắp xếp mảng.

3.1 Thao tác tìm kiếm trên mảng

Trong ví dụ này ta sẽ lần lượt in ra các phần tử nào trong mảng số nguyên dương là số chẵn. Mục tiêu của ví dụ này là minh họa quá trình xử lý tìm kiếm các phần tử thỏa mãn yêu cầu nào đó trên mảng một chiều.

Dòng	
1	#include <stdio.h>
2	#include <math.h>
3	
4	void InSoChanTrongMang(int[], int);
5	
6	void InSoChanTrongMang(int a[], int n){
7	for(int i = 0; i < n; i++){
8	if(a[i] % 2 == 0)
9	printf("%d ", a[i]);
10	}
11	}
12	void main(){
13	int a[5] = {2, 4, 7, 9, 1, 6};
14	InSoChanTrongMang(a, 5);
15	}

3.2 Thao tác sắp xếp

Một kĩ thuật khác cũng khá phổ biến trong mảng một chiều đó là thao tác sắp xếp. Giả sử đoạn mã sau sẽ sắp xếp tăng mảng số nguyên một chiều.

Dòng	
1	#include <stdio.h>
2	
3	void SapXepTang(int[], int);
4	
5	void SapXepTang(int a[], int n){
6	for(int i = 0; i < n – 1; i++){
7	for(int j = i + 1; j < n; j++){
8	if(a[i] > a[j]){
9	int temp = a[i];
10	a[i] = a[j];
11	a[j] = temp;
12	}
13	}
14	}
15	}
16	
17	void main(){
18	int a[5] = {2, 6, 4, 9, 1, 7};
19	SapXepTang(a, 5);
20	XuatMangSoNguyen(a, 5);
21	}

Xem bảng bên dưới để hiểu rõ từng bước trong thuật toán sắp xếp trên

i = 0	a_0	a_1	a_2	a_3	a_4	a_5
	2	6	4	9	1	7
j = 1	2	6	4	9	1	7
j = 2	2	6	4	9	1	7
j = 3	2	6	4	9	1	7
j = 4	1	6	4	9	2	7
j = 5	1	6	4	9	2	7
i = 1	a_0	a_1	a_2	a_3	a_4	a_5
	1	6	4	9	2	7
j = 2	1	4	6	9	2	7
j = 3	1	4	6	9	2	7
j = 4	1	2	6	9	4	7
j = 5	1	2	6	9	4	7

i = 2	a_0	a_1	a_2	a_3	a_4	a_5
	1	2	6	9	4	7
j = 3	1	2	6	9	4	7
j = 4	1	2	4	9	6	7
j = 5	1	2	4	9	6	7
i = 3	a_0	a_1	a_2	a_3	a_4	a_5
	1	2	4	9	6	7
j = 4	1	2	4	6	9	7
j = 5	1	2	4	6	9	7
i = 4	a_0	a_1	a_2	a_3	a_4	a_5
	1	2	4	6	9	7
j = 5	1	2	4	6	7	9

Lưu ý: các ô được tô màu đỏ là những vị trí có xảy ra sự hoán chuyển (điều kiện if thỏa).

4 Các kĩ thuật nâng cao trong mảng một chiều

Tiếp theo ta sẽ tìm hiểu hai kĩ thuật cuối cùng đó là chèn một phần tử vào một vị trí nào đó trong mảng một chiều cũng như xóa một phần tử nào đó ở một vị trí trong mảng.

4.1 Thêm một phần tử vào vị trí nào đó trong mảng một chiều

Hàm chèn phần tử sẽ có kiểu trả về là luận lý (bool), đại diện cho kết quả chèn là thành công (true) hay thất bại (false).

Dòng	
1	`#include <stdio.h>`
2	
3	`bool Chen(int a[], int n, int x, int k){`
4	`  if(k < 0 || k > n)`
5	`  return -1;`
6	`  for(int i = n – 1; i >= k; i--){`
7	`  a[i + 1] = a[i];`
8	`  }`
9	`  a[k] = x;`
10	`  n++;`
11	`  return true;`
12	`}`

4.2 Xóa một phần tử tại vị trí nào đó trong mảng một chiều

Tương tự ta có hàm loại bỏ một phần tử tại một vị trí nào đó trong mảng một chiều

Dòng	
1	`#include <stdio.h>`

```
2
3  bool XoaKhongGiuThuTu(int a[], int &n, int k){
4    if(k < 0 || k >= n) return false;
5    a[k] = a[n – 1];
6    n--;
7   return true;
8  }
9  bool XoaGiuThuTu(int a[], int &n, int k){
10   if(k < 0 || k >= n) return false;
11   for(int i = k; i < n – 1; i++)
12     a[i] = a[i + 1];
13   n--;
14  }
```

Ta có hai phiên bản xóa phần tử, một phiên bản nếu không cần duy trì thứ tự các phần tử thì ta chỉ cần sao chép giá trị phần tử cuối mảng vào vị trí phần tử bị xóa, sau đó giảm kích thước mảng xuống một đơn vị. Với phiên bản cần giữ thứ tự thì ta cần tốn chi phí di dời toàn bộ giá trị bên phải của vị trí cần xóa sang trái một đơn vị, sau đó giảm kích thước mảng xuống.

5 Bài tập

Sinh viên thực hiện các bài tập sau đây:

5.1 Viết chương trình nhập vào mảng số nguyên có n phần tử. Hãy tìm số chẵn lớn nhất và số lẻ nhỏ nhất và xuất ra màn hình

5.2 Hãy nhập dãy n số nguyên dương có giá trị trong khoảng từ 1->100 (nếu nhập sai yêu cầu nhập lại). In ra giá trị trung bình cộng của các số chẵn xuất hiện trong dãy.

5.3 Viết chương trình nhập vào một mảng n số nguyên (n<=100) và n được nhập từ bàn phím, thực hiện các công việc sau:

5.3.1 In ra trung bình cộng của các số dương và số âm có trong mảng.

5.3.2 In ra tất cả các số nguyên tố có trong mảng.

5.3.3 Sắp xếp các số chẵn trong mảng theo thứ tự tăng dần.

5.4 Nhập vào một mảng A, cho biết mảng đó tăng, giảm hay không tăng không giảm.

5.5 Nhập vào một mảng số nguyên A và một số nguyên x, thực hiện các công việc sau:

5.5.1Thông báo x có trong A hay không.

5.5.2 Nếu có, in số lần x xuất hiện trong a và xóa toàn bộ các phần tử x khỏi A.

5.6 Viết chương trình nhập vào một mảng số nguyên A thực hiện các công việc sau:

5.6.1 Sắp xếp A theo thứ tự giảm dần (hoặc tăng dần).

5.6.2 Nhập vào một số nguyên x, chèn x vào A sao cho mảng vẫn giữa nguyên tính thứ tự.

5.7 Viết chương trình thực hiện các công việc sau:

5.7.1 Nhập vào số n nguyên dương từ bàn phím

5.7.2 Sắp xếp mảng theo thứ tự tăng dần.

5.7.3 Hãy loại bỏ các phần tử trùng nhau trong mảng, chỉ giữ lại một giá trị duy nhất trong các số trùng nhau đó.

5.8 Viết chương trình nhập một mảng có n số nguyên dương. Nhập vào một số nguyên dương k. Hãy tính trung bình cộng của các phần tử trong mảng có giá trị lớn hơn hay bằng k.

5.9 Viết chương trình nhập vào mảng nguyên dương A[N], sau đó hãy sắp xếp các phần tử có giá trị lẻ ở đầu mảng và theo thứ tự tăng dần, các phần tử chẵn ở cuối mảng và theo thứ tự giảm dần.

5.10 Viết chương trình nhập vào mảng nguyên dương A[N] và 2 số nguyên dương p, n. Hãy hủy n phần tử trong mảng A bắt đầu từ vị trí p. Sau đó xuất mảng A ra màn hình.

5.11 Viết chương trình nhập vào 2 mảng A[N], B[N] và số nguyên p. Hãy chèn mảng B vào mảng A tại vị trí p. Sau đó xuất mảng A ra màn hình.

5.12 Cho dãy a(a_1, a_2, a_3, …, a_n) và b(b_1, b_2, …, b_n). Viết chương trình thực hiện các công việc sau:

5.12.1 Nhập vào 2 dãy trên, sau đó gộp 2 dãy lại theo thứ tự xen kẽ nhau.

5.12.2 Thực hiện việc xóa các phần tử giống nhau trên hai dãy vừa gộp và in ra màn hình.

5.13 Viết chương trình nhập vào một mảng a, có n phần tử. Ta định nghĩa một mảng con tăng dần trong a là một dãy các phần tử liên tiếp gần nhau và có thứ tự tăng dần trong a.

5.13.1 Xác định số mảng con tăng có trong a.

5.13.2 In ra mảng con tăng dài nhất trong a.

5.14 Viết chương trình nhập vào mảng a

5.14.1 Viết hàm kiểm tra mảng đối xứng không? Nếu có trả về 1 ngược lại trả về 0.

5.14.2 Nhập mảng b, kiểm tra mảng b có phải là mảng con của mảng a không? Nếu có trả về số lần mảng b xuất hiện trong mảng a.

5.15. Viết chương trình thực hiện các bước sau:

5.15.1 Nhập mảng thực.

5.15.2 Sắp xếp mảng thực theo thứ tự tăng dần.

5.15.3 In phần tử có số lần xuất hiện nhiều nhất trong mảng.

5.15.4 Nhập một số thực x, kiểm tra x có xuất hiện trong mảng a hay không. Nếu có in ra vị trí xuất hiện của x trong mảng, ngược lại chèn x vào mảng sao cho mảng vẫn tăng.

5.16. Nhập vào một mảng a. Thực hiện sắp xếp sau:

5.16.1 Tất cả các số lẻ nằm phía trước dãy số, các số chẵn nằm phía sau dãy số, các số 0 nằm giữa.

5.16.2 Nhập vào một số x, hãy tìm số nguyên tố trong a bé hơn và gần với x nhất.

5.17 Hãy nhập mảng 1 chiều có n phần tử là những số nguyên dương. Hãy cho biết mảng đó chứa bao nhiêu số cùng thỏa mãn hai điều kiện: có 3 chữ số và các chữ số đều được sắp tăng. Ví dụ: 122, 457, 889…