CHƯƠNG 4 MẢNG MỘT CHIỀU

Cách khai báo dữ liệu kiểu mảng, các thao tác nhập xuất, các kỹ thuật thao tác trên mảng. Ứng dụng các kỹ thuật này trong việc cài đặt các hàm tìm kiếm, kiểm tra, xây dựng mảng, tách và ghép mảng.

I. TÓM TẮT LÝ THUYẾT

I.1. Khái niệm

Mảng thực chất là một biến được cấp phát bộ nhớ liên tục và bao gồm nhiều biến thành phần.

Các thành phần của mảng là tập hợp các biến có cùng kiểu dữ liệu và cùng tên. Do đó để truy xuất các biến thành phần, ta dùng cơ chế chỉ mục.

I.2. Khai báo mảng

Để khai báo một mảng, ta có 2 cách khai báo sau :

❖ Cách 1: Con trỏ hằng

< Kiểu dữ liệu > < Tên mảng > [< Số phần tử tối đa của mảng>] ;

Ví dụ:

int a[100]; // Khai bao mang so nguyen a gom 100 phan tu

float b[50]; // Khai bao mang so thuc b gom 50 phan tu

❖ Cách 2: Con trỏ

Ý nghĩa: Khi ta khai báo một mảng với kiểu dữ liệu bất kì (int, float, char,…) thì tên của mảng thực chất là một **hằng địa chỉ của phần tử đầu tiên**.

< Kiểu dữ liệu > *< Tên mảng >;

Ví dụ :

```
int *p;          // khai bao con tro p
int b[100];
p = b;           // p tro vao phan tu 0 cua mang b
```

Với cách viết như trên thì ta có thể hiểu các cách viết sau là tương đương

p[i] ⇔ *(p + i) ⇔ b[i] ⇔ *(b+i)

> ✍ ***Lưu ý:*** *Khi sử dụng biến con trỏ để truy xuất mảng, theo cách như trên thì thực chất con trỏ p chỉ chiếm 2 byte bộ nhớ để chứa địa chỉ mà thôi. Để tạo mảng chứa dữ liệu thành phần thì ta phải cấp phát vùng nhớ cho con trỏ p.*
> *Dùng hàm :* ***malloc, calloc*** *trong thư viện* ***<stdlib.h>*** *để cấp phát vùng nhớ.*

Ví dụ:

+ Cách 1: dùng malloc

```
int *px;                        //Khai báo con trỏ  px
px = (int *) malloc (100);  //Cấp phát 100 ô nhớ kiểu int cho con trỏ  px
```

+ Cách 2: dùng calloc

```
int *p;                                 //khai báo con trỏ p
p=(int *) calloc (100,sizeof (int)); //cấp phát 10 ô nhớ mỗi ô chiếm 2bytes
```

Sau khi sử dụng xong thì nên giải phóng vùng nhớ bằng hàm free

Ví dụ : ***free (p) ; // giải phóng vùng nhớ cho con trỏ p.***

I.3. Truy xuất phần tử của mảng

Với khái niệm và cách khai báo như trên ta có hình dạng của mảng một chiều như sau:

Ví dụ : int A[5] // Khai báo mảng A gồm tối đa 5 phần tử nguyên.

Chỉ số	0	1	2	3	4
	A[0]	A[1]	A[2]	A[3]	A[4]

Ví dụ minh hoạ:

Khai báo và gán giá trị cho mảng

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

void main ( )
{
    clrscr ( );
    int a[4] = {5,9,3,8};
    for (int i = 0; i < 4 ; i++)
            printf (" a [ %d ] = %d \t", i , a[i] );
    getch ( );
}
```

Đối với con trỏ: Lấy địa chỉ của phần tử trong mảng ta dùng dấu "**&**"

Ví dụ:

```
int a[7];
```

int *p = a[3]; ***//Lấy địa chỉ phần tử thứ 3***

Ví dụ :

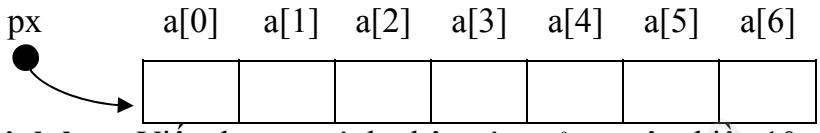
int a[7];

int *px;

px = a; ***//px trỏ tới phần tử thứ 0***

px = px + 4; ***//px trỏ tới phần tử thứ 4***

Từ ví dụ trên ta có thể mô hình hoá mảng như sau:



Ví dụ minh hoạ: Viết chương trình nhập vào mảng một chiều 10 phần tử kiểu số nguyên

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

void main ( )
{
        int a[10], i;
        int *p;

        for (i = 0 ; i < 10 ; i ++)
        {
                printf (" a [ %d ] = ", i );
                scanf (" %d", &a[i] );
        }
        p = a;
        printf (" \n Noi dung mang vua nhap: ");
        for (i = 0; i < 10 ; i ++)
                printf (" %d \t ", *(p + i));
        getch ( );
}
```

II. BÀI TẬP

II.1. Một số kĩ thuật cơ bản

a. Kĩ thuật đặt cờ hiệu

Kĩ thuật này thường được áp dụng cho những bài toán ***"kiểm tra"*** *hay* ***"đánh dấu"****.*

Viết hàm kiểm tra xem mảng các số nguyên có thứ tự tăng dần không? (Trả về 1: Nếu mảng tăng dần, ngược lại trả về 0).

```
int KiemTraTang (int a[ ], int n)
{
	int flag = 1;
	for (int i = 0; i < n-1; i ++ )
		if ( a[i] > a[i+1] ) // Vi phạm điều kiện tăng dần
		{
			flag = 0;
			break;
		}
	return flag;
}
```

Viết hàm kiểm tra xem trong mảng các số nguyên có tồn tại số nguyên lẻ lớn hơn 100 hay không?

(Trả về 1: Nếu có tồn tại số lẻ và lớn hơn 100, ngược lại trả về 0).

```
int KiemTraLe (int a[ ], int n)
{
	int flag = 0;
	for (int i = 0; i < n; i ++ )
		if ( a[i] % 2 != 0 && a[i][j] > 100 )  //Gặp phần tử thoả
		{
			flag = 1;
			break;
		}
	return flag;
}
```

b. Kĩ thuật đặt lính canh

*Kĩ thuật này thường được áp dụng cho những bài tập về **"tìm kiếm"**, **"liệt kê"** theo một điều kiện nhất định nào đó.*

Viết hàm tìm và trả về giá trị lớn nhất trong mảng một chiều các số nguyên.

```
int TimMax (int a[], int n)
{
	int max, i = 1;

	max = a[0];
	while ( i < n )
	{
		if ( a[i] > max )
			max = a[i] ;
		i++;
	}
	return max;
}
```

II.2. Bài tập cơ bản

a. Nhập xuất mảng một chiều

Phương pháp cơ bản

Viết chương trình nhập xuất mảng một chiều các số nguyên.

```
#include <conio.h>
#include <stdio.h>
#define MAX 100

void NhapMang (int a[], int &n)
{
	printf ("Nhap so phan tu: ");
	scanf (" %d ", &n);
	for (int i = 0; i < n; i ++)
	{
		printf (" a [%d] = ", i);
		scanf (" %d ", &a[i]);
	}
}

void XuatMang (int a[], int n)
{
	printf ("\nNoi dung mang: ");
	for (int i = 0; i < n; i ++)
		printf (" %d \t ", a[i]);
}

void main ( )
{
	clrscr ( );
	int a[MAX] , n;
	NhapMang (a,n);
	XuatMang (a,n);
	getch ( );
}
```

Bài tập

1. Viết chương trình nhập xuất mảng một chiều các số thực.
2. Viết chương trình khởi tạo giá trị các phần tử là 0 cho mảng một chiều các số nguyên gồm n phần tử.
3. Viết chương trình phát sinh ngẫu nhiên mảng một chiều các số nguyên âm.
4. Viết chương trình phát sinh ngẫu nhiên mảng một chiều các số nguyên sao cho mảng có thứ tự tăng dần (Không sắp xếp).

5. Viết chương trình nhập mảng các số thực và xuất các phần tử âm trong mảng.
6. Viết chương trình nhập mảng các số nguyên và xuất các phần tử lẻ có trong mảng.
7. Viết chương trình nhập vào mảng một chiều các số nguyên và xuất ra các phần tử chẵn nhỏ hơn 20.
8. Viết chương trình nhập vào mảng một chiều các số nguyên và xuất ra màn hình các phần tử là số nguyên tố.
9. Viết chương trình nhập vào số nguyên n và liệt kê các số nguyên tố nhỏ hơn n, nếu mảng không tồn tại số nguyên tố nào nhỏ hơn n thì phải xuất ra một câu thông báo.
10. Viết chương trình nhập vào mảng một chiều các số nguyên và xuất ra màn hình các phần tử là số chính phương nằm tại những vị trí lẻ trong mảng.

b. Tìm kiếm trên mảng một chiều

Phương pháp cơ bản

Viết hàm tìm phần tử có giá trị x xuất hiện đầu tiên trong mảng một chiều.

(Nếu tìm thấy trả về vị trí xuất hiện x, ngược lại trả về -1)

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

Bài tập

11. Viết hàm tìm vị trí phần tử có giá trị x xuất hiện cuối cùng trong mảng.
12. Viết hàm tìm vị trí của phần tử nhỏ nhất trong mảng các số nguyên.
13. Viết hàm tìm vị trí của phần tử lớn nhất trong mảng các số nguyên.
14. Viết hàm in vị trí các phần tử nguyên tố trong mảng các số nguyên.
15. Viết hàm in vị trí các phần tử nguyên tố lớn hơn 23.
16. Viết hàm tìm vị trí phần tử âm đầu tiên trong mảng. Nếu không có phần tử âm trả về –1.
17. Viết hàm tìm vị trí phần tử âm lớn nhất trong mảng.

18. Viết hàm tìm vị trí phần tử dương đầu tiên trong mảng. Nếu không có phần tử âm trả về –1.
19. Viết hàm tìm vị trí phần tử dương bé nhất trong mảng.
20. Viết hàm in các phần tử là bội của 3 và 5.
21. Viết hàm tìm số chẵn cuối cùng có trong mảng, nếu không tồn tại số chẵn hàm trả về -1 .
22. Viết hàm tìm số lẻ lớn nhất có trong mảng, nếu không tồn tại số lẻ hàm trả về -1.
23. Viết hàm tìm và đổi chỗ phần tử lớn nhất với phần tử nhỏ nhất trong mảng.
24. Nhập vào X. Viết hàm in ra màn hình những phần tử có giá trị từ 1 đến X có trong mảng.
25. Viết chương trình nhập vào một dãy số a gồm n số thực ($n \le 100$), nhập vào dãy số b gồm m số thực ($m \le 100$).
    - In ra những phần tử chỉ xuất hiện trong dãy a mà không xuất hiện trong dãy b.
    - In ra những phần tử xuất hiện ở cả hai dãy.

c. Đếm – Tần suất

Phương pháp cơ bản

Viết hàm đếm các phần tử chia hết cho 5 trong mảng các số nguyên.

```
int Dem (int a[], int n )
{
        int dem = 0;
        for (int i = 0; i < n ; i++ )
                if ( a[i] % 5 == 0 )
                        dem++;
        return dem;
}
```

Bài tập

26. Viết hàm đếm các phần tử âm, dương trong mảng.
27. Viết hàm đếm các phần tử chẵn, lẻ trong mảng.
28. Viết hàm đếm số lần xuất hiện của phần tử x trong mảng.
29. Viết hàm đếm các phần tử nhỏ hơn x trong mảng.
30. Viết hàm đếm các phần tử là số nguyên tố trong mảng.

31. Viết hàm đếm các phần tử là số hoàn thiện trong mảng.
32. Viết hàm đếm các phần tử là bội của 3 và 5 trong mảng các số nguyên.

d. Tính tổng – Trung bình có điều kiện

Phương pháp cơ bản

Viết hàm tính tổng các phần tử trong mảng.

```
long  TinhTong (int a[], int n )
{
        long  tong = 0;
        for (int i = 0; i < n; i++ )
                tong = tong + a[i] ;
        return tong;
}
```

Viết hàm tính giá trị trung bình các phần tử có giá trị âm trong mảng.

Đối với hàm tính trung bình có điều kiện phải lưu ý khi chia giá trị (Có thể mảng không có phần tử nào thoả điều kiện, nếu ta chia tức là chia cho 0).

```
float  TrungBinhAm (int a[], int n )
{
        long  tong = 0;
        int spt=0;
        for (int i = 0; i < n; i++ )
                if( a[i]<0 )
                {
                    tong = tong + a[i] ;
                   spt++;
                }
        if(spt==0)
                return 0;
        return 1.0*tong/spt;
}
```

Bài tập

33. Viết hàm tính tổng các phần tử chẵn trong mảng.
34. Viết hàm tính tổng các phần tử lẻ trong mảng các số nguyên.
35. Viết hàm tính tổng các phần tử nguyên tố trong mảng.
36. Viết hàm tính tổng các phần tử nằm ở vị trí chẵn trong mảng các số nguyên.
37. Viết hàm tính tổng các phần tử nằm ở vị trí nguyên tố trong mảng.
38. Viết hàm tính tổng các phần tử chia hết cho 5 có trong mảng.
39. Viết hàm tính tổng các phần tử cực đại trong mảng các số nguyên *(phần tử cực đại là phần tử lớn hơn các phần tử xung quanh nó)*.

Ví dụ : *1* ***5*** *2* ***6*** *3* ***5*** *1* ***8*** *6*

40. Viết hàm tính tổng các phần tử cực tiểu trong mảng các số nguyên *(phần tử cực tiểu là phần tử nhỏ hơn các phần tử xung quanh nó).*

    Ví dụ : 6 4 **2** 9 5 **3** 7 **1** 5 8

41. Viết hàm tính tổng các phần tử là bội của 3 và 5 trong mảng các số nguyên.
42. Viết hàm tính tổng các phần tử là số hoàn thiện trong mảng các số nguyên.
43. Viết hàm tính giá trị trung bình của các số hoàn thiện trong mảng các số nguyên.

e. Sắp xếp

Kĩ thuật cơ bản

Viết hàm sắp xếp mảng theo thứ tự tăng dần.

```
void HoanVi (int &a, int &b)
{
        int tam = a;
        a = b;
        b = tam;
}

void SapTang (int a[], int n)
{
        for (int i = 0; i < n-1 ; i++)
                for (int j = i+1; j < n; j++)
                        if (a[i] > a [j])
                                HoanVi (a[i], a[j]);
}
```

Bài tập

44. Viết hàm sắp xếp mảng theo thứ tự giảm dần.
45. Viết hàm sắp xếp mảng theo thứ tự tăng dần của các phần tử là số nguyên tố.
46. Viết hàm sắp xếp các phần tử lẻ tăng dần.
47. Viết hàm sắp xếp các phần tử chẵn giảm dần.
48. Viết hàm sắp xếp các phần tử chẵn nằm bên trái theo thứ tự tăng dần còn các phần tử lẻ bên phải theo thứ tự giảm dần.
49. Viết hàm sắp xếp các phần tử âm giảm dần từ trái sang phải, phần tử dương tăng dần từ phải sang trái.

f. Xoá

Kĩ thuật cơ bản

Duyệt mảng từ trái sang phải . Xuất phát từ vị trí cần xoá tiến hành dời lần lượt các phần tử về phía trước cho đến khi kết thúc mảng, sau đó giảm kích thước mảng.

Vấn đề đặt ra là tìm vị trí cần xóa theo điều kiện bài toán rồi thực hiện xóa.

Viết hàm xoá phần tử đầu tiên của mảng.

```
void XoaDau (int a[], int &n)
{
        for (int i = 0; i < n-1 ; i++)
                a[i] = a[i+1];
      n--;
}
```

Viết hàm xoá phần tử tại vị trí (vitri) cho trước trong mảng.

```
void XoaTaiViTri (int a[], int &n, int vitri)
{
        for (int i = vitri; i < n-1 ; i++)
                a[i] = a[i+1];
      n--;
}
```

Bài tập

50. Viết hàm xoá phần tử tại vị trí lẻ trong mảng.
51. Viết hàm xoá phần tử có giá trị lớn nhất trong mảng.
52. Nhập vào giá trị X. Viết hàm xoá tất cả các phần tử có giá trị nhỏ hơn X.
53. Nhập vào giá trị X. Viết hàm xoá phần tử có giá trị gần X nhất.

g. Chèn

Kĩ thuật cơ bản

Duyệt mảng từ phải sang trái. Xuất phát từ cuối mảng tiến hành đẩy lần lượt các phần tử về phía sau cho đến vị trí cần chèn, chèn phần tử cần chèn vào vị trí chèn và tăng kích thước mảng.

Trước khi chèn ta phải xác định vị trí cần chèn theo điều kiện bài toán.

Thêm phần tử có giá trị X vào cuối mảng.

```
void ThemCuoi (int a[], int &n, int X)
{
        a[n]=X;
        n++;
```

```
}
```

Chèn phần tử có giá trị X vào mảng tại vị trí cho trước

```
void ChenX  (int a[], int &n, int X, int vitri)
{
        for (int i = n; i >vitri ; i--)
                a[i] = a[i-1] ;
        a[vitri] = X;
        n++;
}
```

Bài tập

54. Viết hàm chèn phần tử có giá trị X vào vị trí đầu tiên của mảng.
55. Viết hàm chèn phần tử có giá trị X vào phía sau phần tử có giá trị lớn nhất trong mảng.
56. Viết hàm chèn phần tử có giá trị X vào trước phần tử có giá trị là số nguyên tố đầu tiên trong mảng.
57. Viết hàm chèn phần tử có giá trị X vào phía sau tất cả các phần tử có giá trị chẵn trong mảng.

h. Tách / ghép mảng

Kĩ thuật tách cơ bản

Cho mảng a kích thước n (n chẵn). Tách mảng a thành 2 mảng b và c sao cho: b có ½ phần tử đầu của mảng a, ½ phần tử còn lại đưa vào mảng c.

```
void TachMang(int a[], int n, int b[], int &m, int c[], int &l)
{
        int k=n/2;

        m=l=0;
        for(int i=0; i<k; i++)
        {
                b[m++]=a[i];
                c[l++]=a[k+i]
        }
}
```

Kĩ thuật ghép cơ bản

Cho 2 mảng số nguyên a và b kích thước lần lượt là n và m. Viết chương trình nối mảng b vào cuối mảng a.

```
void NoiMang(int a[], int &n, int b[], int m)
{
	for(int i=0; i<m; i++)
		a[n+i]=b[i];
	n=n+m;
}
```

Cho 2 mảng số nguyên a và b kích thước lần lượt là n và m. Viết chương trình nối xen kẻ ***(đan xen)*** lần lượt các phần tử mảng a và b vào mảng c.

Cách thực hiện: Đưa lần lượt từng phần tử của mảng a và mảng b vào mảng c, tăng chỉ số tương ứng. Nếu một trong hai mảng hết trước thì chép tất cả các phần tử còn lại của mảng chưa hết vào mảng c.

*Đặt **i** là chỉ số của mảng a; **j**: chỉ số của mảng b và **k** là chỉ số của mảng c.*

```
void NoiMang(int a[], int &n, int b[], int m, int c[], int &k)
{
	int i=0, j=0;
	k=0;
	while(i<n&&j<m)
	{
		c[k++]=a[i++];
		c[k++]=b[j++];
	}
	while(i<n)
		c[k++]=a[i++];
	while(j<m)
		c[k++]=b[j++];
}
```

Bài tập

58. Viết chương trình tách 1 mảng các số nguyên thành 2 mảng a và b, sao cho mảng a chứa toàn số lẻ và mảng b chứa toàn số chẵn.

    Ví dụ: *Mảng ban đầu:* *1 3 **8 2** 7 5 9 0 **10***

    Mảng a: *1 3 7 5 9*

    Mảng b: *8 2 10*

59. Cho 2 mảng số nguyên a và b kích thước lần lượt là n và m. Viết chương trình nối 2 mảng trên thành mảng c theo nguyên tắc chẵn ở đầu mảng và lẻ ở cuối mảng.

    Ví dụ: *Mảng a:* *3 2 7 5 9*

Mảng b: *1 8 10 4 12 6*

Mảng c: ***6 12 4 10 2 8*** *3 1 7 5 9*

II.3. Bài tập luyện tập và nâng cao

60. Viết chương trình nhập vào mảng A gồm n phần tử, trong quá trình nhập kiểm tra các phần tử nhập vào không được trùng, nếu trùng thông báo và yêu cầu nhập lại.
61. Viết hàm tính tổng của từng dãy con giảm có trong mảng.
62. **(*)** Cho mảng các số nguyên a gồm n phần tử ($n \le 30000$) và số dương k ($k \le n$). Hãy chỉ ra số hạng lớn thứ k của mảng.

    Ví dụ: *Mảng a:* *6 3 1 10 11 18*

    k = 2

    Kết quả: 10
63. **(*)** Cho 2 dãy A, B các số nguyên *(kích thước dãy A nhỏ hơn dãy B)*. Hãy kiểm tra xem A có phải là con của B hay không?
64. Viết hàm liệt kê các bộ 4 số a, b, c, d trong mảng các số nguyên *(có ít nhất 4 phần tử và đôi một khác nhau)* sao cho a + b = c + d.
65. **(*)** Viết chương trình tính trung bình cộng của các tổng các dãy tăng dần có trong mảng các số nguyên.

    Ví dụ: *1 2 3 4 2 3 4 5 6 4 5 6 => TB = 15.*
66. Viết chương trình tính tổng tất cả các phần tử xung quanh trên mảng các số nguyên. (Phần tử xung quanh là hai phần tử bên cạnh cộng lại bằng chính nó (Ví dụ: 1 3 2 ➔ 1,2 là hai phần tử xung quanh của 3).

    Ví dụ : 1 3 2 5 3 9 6 ➔ tổng 17
67. **(**)** Viết chương trình nhập vào hai số lớn a, b nguyên (a, b có từ 20 chữ số trở lên). Tính tổng, hiệu, tích, thương của hai số trên.
68. Viết hàm tính tổng các phần tử là số Amstrong (số Amstrong là số có đặc điểm như sau: số có k ký số, tổng của các luỹ thừa bậc k của các ký số bằng chính số đó.

    Ví dụ: 153 là số có các ký số $1^3+5^3+3^3$= 153 là một số Amstrong).
69. Viết hàm tìm và xóa tất cả các phần tử trùng với x trong mảng một chiều các số nguyên, nếu không tồn tại phần tử x trong mảng thì trả về -1.

70. Viết hàm xoá tất cả những phần tử trùng nhau trong dãy chỉ giữ lại một phần tử trong đó.

    Ví dụ: *1 6 **2** 3 **2** 4 **2** 6 5* ➔ *1 6 2 3 4 5*

71. **(**)** Viết hàm xoá những phần tử sao cho mảng kết quả có thứ tự tăng dần và số lần xoá là ít nhất.

72. Cho dãy a gồm n số nguyên có thứ tự tăng dần. Nhập vào một phần tử nguyên X, viết hàm chèn X vào dãy sao cho dãy vẫn có thứ tự tăng dần (*không sắp xếp*).

73. Viết chương trình tìm số lẻ nhỏ nhất lớn hơn mọi số chẵn có trong mảng.

74. Viết hàm tìm giá trị chẵn nhỏ nhất nhỏ hơn mọi giá trị lẻ trong mảng các số nguyên.

75. Viết hàm tìm phần tử xuất hiện nhiều nhất trong mảng các số nguyên.

76. Viết chương trình đếm và liệt kê các mảng con tăng dần trong mảng một chiều các số nguyên.

    Ví dụ: *6 5 3 **2 3 4** 2 7 các dãy con tăng dần là 2 3 4 và 2 7*

77. Viết chương trình tìm mảng con tăng dần có tổng lớn nhất trong mảng một chiều.

78. **(*)** Viết chương trình nhập vào một dãy số a gồm n số nguyên (n <= 100). Tìm và in ra dãy con tăng dài nhất

    Ví dụ : *Nhập dãy a : **1 2 3 6** 4 7 8 **3 4 5 6 7 8 9** 4 5*
    Dãy con tăng dài nhất : 3 4 5 6 7 8 9

79. **(**)** Viết chương trình tách 1 mảng các số nguyên thành 2 mảng a và b, sao cho kết quả thu được là:

    - Mảng a chứa toàn số lẻ tăng dần.
    - Mảng b chứa toàn số chẵn giảm dần.

    (Không dùng sắp xếp)

    Hướng dẫn: Tìm vị trí chèn thích hợp khi trích phần tử từ mảng ban đầu.

    Ví dụ: *Mảng ban đầu:* *9 3 **8 2** 7 5 1 0 **10***
    Mảng a: *1 3 5 7 9*
    Mảng b: *10 8 2*

80. **(**) Viết** chương trình in ra tam giác Pascal (dùng mảng một chiều).

81. Viết chương trình nhập vào dãy số a gồm n số thực (n <= 100), nhập vào dãy số b gồm m số thực (m <= 100).

    - Hãy sắp xếp hai dãy theo thứ tự tăng dần.

- **(*)** Trộn 2 dãy trên thành dãy c sao cho dãy c vẫn có thứ tự tăng.
- Xuất dãy a, b, c ra màn hình.

82. **(*)** Cho mảng C có n phần tử (n < 200), các phần tử là các chữ số trong hệ đếm cơ số 16 (Hexa) (điều kiện mỗi phần tử <= n). Hãy tách mảng C ra các mảng con theo điều kiện sau: các mảng con được giới hạn bởi hai lần xuất hiện thứ hai của con số trong dãy.

   *Ví dụ: **123**A45**1**8B**23*** ➔ *có các dãy con là123A451, 23A4518B2, 23A4518B23*

83. **(**)** Cho hai số nguyên dương A, B. Hãy xác định hai số C, D tạo thành từ hai số A, B sao cho C là số lớn nhất, D là số nhỏ nhất. Khi gạch đi một số chữ số trong C (D), thì các số còn lại giữ nguyên tạo thành A, các chữ số bỏ đi giữ nguyên tạo thành B.

   Ví dụ: A = 52568, B = 462384 -> C = 54625682384, D = 45256236884.

84. Viết chương trình nhập vào dãy số a gồm n số nguyên (n <= 100).
    - Hãy đảo ngược dãy đó.

      Ví dụ: Nhập a: 3 4 5 2 0 4 1
      Dãy sau khi đảo: 1 4 0 2 5 4 3
    - **(*)** Hãy kiểm tra xem dãy đã cho có thứ tự chưa (dãy được gọi là thứ tự khi là dãy tăng hoặc dãy giảm).

85. Cho mảng A có n phần tử hãy cho biết mảng này có đối xứng hay không.

86. **(**)** Hãy viết chương trình phát sinh ngẫu nhiên mảng các số nguyên gồm 10.000 phần tử, mỗi phần tử có giá trị từ 0 đến 32.000 và xây dựng hàm thống kê số lần xuất hiện các phần tử trong mảng, sau đó cho biết phần tử nào xuất hiện nhiều lần nhất.

    Ví dụ: Mảng: 5 6 11 4 4 5 4
    5 xuat hien 2 lan
    6 xuat hien 1 lan
    11 xuat hien 1 lan
    4 xuat hien 3 lan

    4 xuat hien nhieu lan nhat

87. Cho mảng A có n phần tử. Nhập vào số nguyên k ($k \geq 0$), dịch phải xoay vòng mảng A k lần.

    Ví dụ: *Mảng A: 5 7 2 3 1 9*
    Nhập k = 2
    Dịch phải xoay vòng mảng A: 1 9 5 7 2 3

III. KẾT LUẬN

- Dữ liệu kiểu mảng dùng cho việc biểu diễn những thông tin có **cùng kiểu** dữ liệu liên tiếp nhau.
- Khi cài đặt bài tập mảng một chiều nên **xây dựng thành những hàm chuẩn** để dùng lại cho các bài tập khác.
- Các thao tác trên mảng đều theo quy tắc nhất định, chúng ta có thể ứng dụng mảng trong việc biểu diễn số lớn, dùng bảng tra, khử đệ qui, …