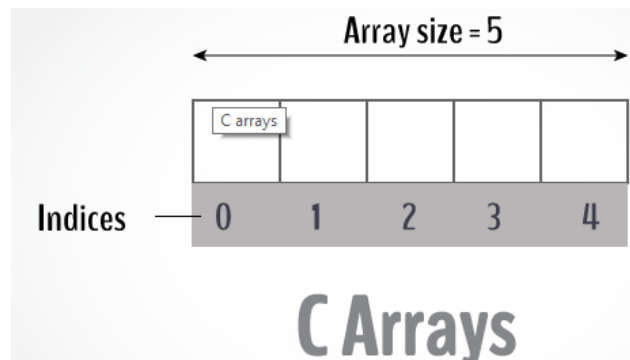
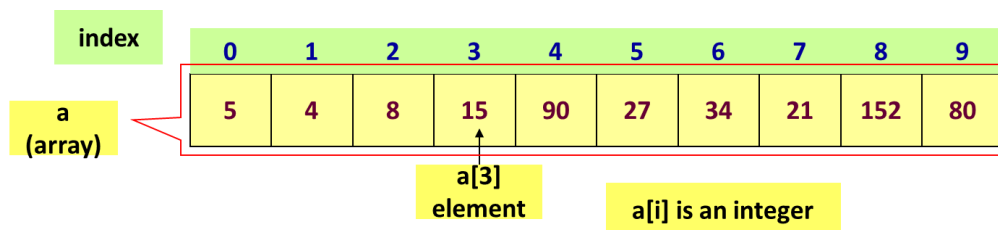


BÀI THỰC HÀNH SỐ 10: MẢNG MỘT CHIỀU (One-dimensional Arrays)



1-D array: tập các phần tử cùng kiểu dữ liệu và được lưu trữ liên tiếp trong bộ nhớ.



I. Khai báo

```
DataType ArrayName[NumberOfElements];
```

Examples:

```
int a1[5];
```

```
char s[12];
```

```
double a3[100];
```

II. Duyệt mảng

A way to visit each element of an array

Suppose that the 1-D array, named *a*, containing *n* elements.

Forward traversal:

```
int i;
```

```
for (i=0; i<n; i++)
```

```
{ [if (condition)] Access a[i];
```

```
}
```

Backward traversal:

```
int i;
```

```
for (i=n-1; i>=0; i--)
```

```
{ [if (condition)] Access a[i];
```

```
}
```

III. Một số ví dụ

Example 1:

```

1. // Program to find the average of n (n < 10) numbers using arrays
2.
3. #include <stdio.h>
4. int main()
5. {
6.     int marks[10], i, n, sum = 0, average;
7.     printf("Enter n: ");
8.     scanf("%d", &n);
9.     for(i=0; i<n; ++i)
10.    {
11.        printf("Enter number%d: ",i+1);
12.        scanf("%d", &marks[i]);
13.        sum += marks[i];
14.    }
15.    average = sum/n;
16.
17.    printf("Average = %d", average);
18.
19.    return 0;
20. }
```

Example 2: Chương trình sau thực hiện các yêu cầu:

- Nhập các giá trị cho một mảng nguyên có thể chứa tối đa 100 phần tử
- In ra giá trị lớn nhất
- In ra các giá trị của mảng
- In ra các giá trị chẵn.

```

5 #include <stdio.h>
6 #define MAXN 100
7
8 /*Prototypes*/
9 void input(int a[], int n);
10 int print(int a[], int n);
11 int max(int a[], int n);
12 void printEven(int a[], int n);
13 int main()
14 {
15     int a[MAXN]; //Mang lưu trữ tối đa 100 giá trị nguyên
16     int n; // số phần tử thực sự sử dụng
17     int maxVal;
18     do
19     {
20         printf("Số phần tử sẽ sử dụng (1..%d):", MAXN);
21         scanf("%d", &n);
22     }
23     while (n<1 || n>MAXN);
24
25     printf("Nhập %d giá trị cho mảng:\n", n);
26     input(a,n);
```

```

27     maxVal=max(a,n);
28     printf("Gia tri lon nhat:%d\n", maxVal);
29     printf("\nMang da nhap:");
30     print(a,n);
31     printf("\nCac gia tri chan trong mang:");
32     printEven(a,n);
33     printf("\nHave a Nice Day!\n\n");
34     getchar();
35     return 0;
36 }
37 //-----
38 void input(int a[], int n)
39 {
40     int i;
41     for(i=0; i<n; i++)
42         scanf("%d", &a[i]);
43 }
44 //-----
45 int print(int a[], int n)
46 {
47     int i;
48     for(i=0; i<n; i++)
49         printf("%d ", a[i]);
50 }
51 //-----
52 int max(int a[], int n)
53 {
54     int result = a[0];
55     int i;
56     for(i=1; i<n; i++)
57         if(result<a[i])
58             result=a[i];
59     return result;
60 }
61 //-----
62 void printEven(int a[], int n)
63 {
64     int i;
65     for(i=0; i<n; i++)
66         if(a[i]%2==0)
67             printf("%d ", a[i]);
68 }

```

Example 3: Tìm kiếm trên mảng một chiều

```

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

```

```
int lastLinearSearch ( double x, double *a, int n)
{ int i;
  for ( i=n-1; i>=0; i--)
    if ( x == a[i] ) return i;
  return -1;
}
```

```
/* Linear search Demo. */
#include <stdio.h>
int firstLinearSearch ( int x, int a[], int n)
{
    /* Your code */
}
int lastLinearSearch ( int x, int a[], int n)
{
    /* Your code */
}

int main()
{ int a[] = { 3,34,5,1,2,8,9,2,9 }, x=2;
  int pos1= firstLinearSearch(x,a,9);
  if (pos1>=0)
  { int pos2= lastLinearSearch(x,a,9);
    printf("First existence:%d, last existence:%d\n", pos1, pos2);
  }
  else printf("%d does not exist!\n", x);
  getchar();
  return 0;
}
```




Example 4: Sắp xếp mảng một chiều

```

2 #include <stdio.h>
3 void ascSelectionSort( int* a, int n)
4 { int minIndex; /* index of min. value in a group */
5   int i,j ; /* vars for looping */
6   /* Group begins at position i to n-1*/
7   for (i=0; i< n-1; i++)
8   { minIndex = i; /* init minimum position */
9     /* update minIndex of the group at i, i+1,..., n-1*/
10    for (j=i+1; j<n; j++)if (a[minIndex]> a[j]) minIndex= j;
11    /* Move minimum value to the begin of the group */
12    if (minIndex > i)
13    { int t = a[minIndex];
14      a[minIndex] = a[i];
15      a[i] = t;
16    }
17  }
18 }
19
20 void print (int*a, int n)
21 {
22     /* Your code */
23 }
24 int main()
25 { int a[] = { 1,3,5,7,9,2,4,6,8, 0 };
26   ascSelectionSort(a, 10);
27   print(a,10);
28   getchar();
29   return 0;
30 }

```



IV. Bài tập thực hành

Bài 1

Viết chương trình để nhập n giá trị trong một mảng và hiển thị theo thứ tự ngược lại.

Test Data :

Input the number of elements to store in the array :3

Input 3 number of elements in the array :

element - 0 : 2

element - 1 : 5

element - 2 : 7

Expected Output :

The values store into the array are :

2 5 7

The values store into the array in reverse are :

7 5 2

Bài 2

Viết chương trình để tìm phần tử lớn nhất và nhỏ nhất trong một mảng.

Test Data :

Input the number of elements to be stored in the array :3

Input 3 elements in the array :

element - 0 : 45

element - 1 : 25

element - 2 : 21

Expected Output :

Maximum element is : 45

Minimum element is : 21

Bài 3

Viết chương trình để trộn hai mảng có cùng kích thước được sắp xếp theo thứ tự giảm dần.

Test Data :

Input the number of elements to be stored in the first array :3

Input 3 elements in the array :

element - 0 : 1

element - 1 : 2

element - 2 : 3

Input the number of elements to be stored in the second array :3

Input 3 elements in the array :

element - 0 : 1

element - 1 : 2

element - 2 : 3

Expected Output :

The merged array in decending order is :

3 3 2 2 1 1

Bài 4

Viết chương trình sử dụng mảng 1-D để quản lý một dãy số nguyên (tối đa 100 phần tử), thực hiện các công việc sau:

1. Nhập N giá trị cho mảng.
2. Xuất các phần tử trong mảng.
3. Xuất các số nguyên tố có trong mảng.

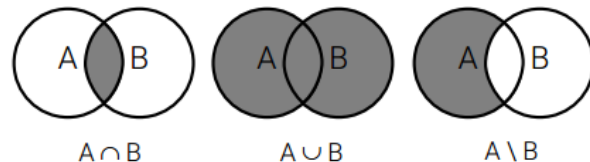
4. Xuất các phần tử có giá trị nằm trong khoảng $[a, b]$. (a, b nhập từ bàn phím)
5. Xuất các phần tử từ vị trí x đến y (x, y nhập từ bàn phím)
6. Tính tổng các giá trị trong mảng.
7. Tổng các số chẵn.
8. Thêm giá trị X vào vị trí P (X, P nhập từ bàn phím)
9. Xóa phần tử tại vị trí P (P nhập từ bàn phím)
10. Tìm kiếm giá trị X trong mảng (X nhập từ bàn phím)

Bài 5*****

Cho hai mảng A, B là hai tập hợp, khởi tạo trước hoặc nhập từ bàn phím. Tạo

mảng C là một tập hợp gồm các phần tử:

- a. Xuất hiện trong cả A và B (giao).
- b. Không xuất hiện trong B (hiệu).
- c. Xuất hiện trong A hoặc B (hợp).



```

Tập hợp A: {1, 2, 3, 5}
Tập hợp B: {1, 3, 6, 7}
C = A * B: {1, 3}
C = A + B: {1, 2, 3, 5, 6, 7}
C = A \ B: {2, 5}
    
```

Example 5: Phát sinh ngẫu nhiên Randomize (stdlib.h)

```

5 #include <stdio.h>
6 #include <stdlib.h>
7 #include <time.h>
8 int main()
9 {
10     int i, a=5, b=50;
11     double x=3.5, y= 20.8;
12     srand(time(NULL));
13     printf("\nPhat sinh 10 so nguyen ngau nhien:\n");
14     for (i=0;i<10; i++)
15         printf("%d ", rand());
16     printf("\n\nPhat sinh 10 so nguyen ngau nhien between:%d...%d\n", a, b);
17     for (i=0;i<10; i++)
18         printf("%d ", a + rand()% (b-a));
19     printf("\n\nPhat sinh 5 so thuc ngau nhien between:%lf...%lf\n", x, y);
20     for (i=0;i<5; i++)
21         printf("%lf ", x + (double)rand()/RAND_MAX*(y-x));
22     printf("\n\n");
23     return 0;
24 }
    
```

Bài 6

Viết chương trình thực hiện những yêu cầu sau:

1. Tạo ngẫu nhiên mảng một chiều n phần tử nguyên có giá trị chứa trong đoạn $[-100, 100]$
2. Xuất mảng
3. Xuất các số nguyên tố có trong mảng.
4. Xuất các phần tử có giá trị nằm trong khoảng $[a, b]$. (a, b nhập từ bàn phím)
5. Xuất các phần tử từ vị trí x đến y (x, y nhập từ bàn phím)
6. Tính trung bình các giá trị của mảng.
7. Đếm số phần tử chia hết cho 4 và có chữ số tận cùng là 6.
8. Thay các phần tử lẻ bằng 2 lần giá trị của nó.
9. Thêm giá trị X vào vị trí P (X, P nhập từ bàn phím)
10. Xóa phần tử tại vị trí P (P nhập từ bàn phím)
11. Tìm kiếm giá trị X trong mảng (X nhập từ bàn phím)
12. Sắp xếp mảng tăng/giảm