NMLT- Hướng dẫn thực hành tuần 9

CODE SAMPLES

1. MẢNG 1 CHIỀU

Sắp xếp các phần tử trên mảng tăng dần:

```
void SapXep(int a[], int n, bool bSapTang)
{
   for (int i = 0; i<n - 1; i++)
         for (int j = i + 1; j<n; j++)
         {
               // bSapTang == true -> sap tang
               // bSapTang == false -> sap giam
               if (bSapTang == true)
               {
                     if (a[j] < a[i])
                           HoanVi(a[i], a[j]);
               }
               else
               {
                     if (a[j] > a[i])
                           HoanVi(a[i], a[j]);
               }
         }
}
```

Trộn 2 mảng một chiều a, b các phần tử xen kẽ nhau thành một mảng một chiều:

```
void Tron2Mang(int a[], int n, int b[], int m, int c[])
{
   int min = (n > m ? m : n);
   int i = 0, j = 0;
   for (i = 0; i < min; i++, j += 2)
   {
```

```
            c[j] = a[i];
            c[j + 1] = b[i];
      }
      while (i < n)
            c[j++] = a[i++];
      while (i < m)
            c[j++] = b[i++];
}
```

Xóa một phần tử bất kỳ trên mảng:

```
void Xoa1PhanTu(int a[], int &n, int x)
{
      int b[100];
      for (int i = 0; i<n; i++)
            b[i] = a[i];
      int m = 0;
      for (i = 0; i<n; i++)
            if (b[i] != x)
                  a[m++] = b[i];
}
```

2. CHUỖI

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <string.h>
#include <ctype.h>
#define NUM_ALPHABET 26
//Viết hàm in mỗi từ của chuỗi trên một dòng.
//Lập trinh C:
//      Lập
//      trinh
//      C
```

```
void InTu(char *s)
{
   int i = 0, j;
   char tu[20];
   int len = strlen(s);
   while (i < len)
   {
         //Đi qua các khoảng trắng
         while (s[i] == ' ' && i < len)
               i++;
         if (i < len)
         {
               //Đi qua các ký tự khác khoảng trắng
               j = 0;
               while (s[i] != ' ' && i < len)
               {
                     tu[j] = s[i];
                     j++; i++;
               }
               tu[j] = 0;
               printf("%s\n", tu);
         }
   }
}
//Viết hàm thống kê số lần xuất hiện của mỗi ký tự từ A đến Z trong một chuỗi
(không phân biệt chữ hoa hay chữ thường).
//Hello World
//      D : 1 lần
//      E : 1 lần
//      H : 1 lần
//      L : 3 lần
//      ...
```

```
void ThongKe(char *s)
{
   int dem[NUM_ALPHABET]; //Mảng lưu trữ số lần xuất hiện ký tự
   //Khởi tạo mảng đếm
   for (int i = 0; i < NUM_ALPHABET; i++)
        dem[i] = 0;
   //Thống kê
   int len = strlen(s);
   for (int i = 0; i < len; i++)
        if (toupper(s[i]) >= 'A' && toupper(s[i]) <= 'Z')
             dem[toupper(s[i]) - 'A']++;
   //In kết quả
   for (int i = 0; i < NUM_ALPHABET; i++)
        printf("ky tu %c: %d lan\n", i + 'A', dem[i]);
}

void main()
{
   char buffer[81];
   int i, ch;
   for (i = 0; (i < 80) && ((ch = getchar()) != EOF) && (ch != '\n'); i++)
        buffer[i] = (char)ch;
   // Terminate string with a null character (them phan tu ket thuc chuoi)
   buffer[i] = '\0';
   printf("Input was: %s\n", buffer);
   InTu(buffer);
   ThongKe(buffer);
   system("pause");
}
```

BÀI TẬP THỰC HÀNH

//Array

1. Sắp xếp các phần tử là số nguyên tố.
2. Trộn hai mảng tăng dần lại thành một mảng có thứ tự tăng dần.
3. Tìm vị trí của phần tử
   a. lớn nhất
   b. nhỏ nhất
4. Thêm một phần tử có giá trị x vào mảng tại vị trí k.

//String

5. Viết các hàm thực hiện:
   a. Chuyển đổi một chuỗi sang dạng thường
   b. Chuyển đổi một chuỗi sang dạng hoa
   c. Chuyển đổi một chuỗi sang dạng Title case
   (ký tự đầu của mỗi từ là chữ hoa, các ký tự còn lại là chữ thường)
6. Viết hàm kiểm tra một chuỗi có đối xứng hay không?

   Ví dụ: Các chuỗi level, radar, dad… là các chuỗi đối xứng
7. Viết hàm tìm chuỗi đảo ngược của một chuỗi.

   Ví dụ: Chuỗi "Lap trinh C" có chuỗi đảo ngược là "C hnirt paL"

BÀI TẬP VỀ NHÀ

//Array

1. Liệt kê tần suất xuất hiện của các phần tử. Ví dụ với mảng 12 34 12 34 43 12 5, thì tần suất xuất hiện của các phần tử như trong bảng sau:

Phần tử	Tần suất
12	3
34	2
43	1
5	1

2. Xóa phần tử tại vị trí k trong mảng.
3. Trộn 2 mảng một chiều có cùng độ dài thành một mảng một chiều với mỗi phần tử của mảng mới là tổng của 2 phần tử tương ứng từ 2 mảng cho trước.
4. Xóa n phần tử liên tục trên mảng bắt đầu từ một vị trí x cho trước.
5. Nhập vào 2 mảng một chiều, xóa trên 2 mảng này tất cả các phần tử trùng nhau của 2 mảng.

6. Kiểm tra xem có tồn tại mảng con tăng dần hay giảm dần không. Nếu có, in mảng con tăng dần dài nhất xuất hiện trong mảng. Nếu có nhiều mảng cùng dài nhất thì chỉ cần in ra một.

//String

7. Nhập một chuỗi S từ bàn phím. Tìm ký tự xuất hiện nhiều nhất trong chuỗi đó và số lần xuất hiện.
8. Nhập họ tên của một người từ bàn phím. Hãy chuẩn hóa chuỗi họ tên này. (Xóa các khoảng trắng thừa và ký tự đầu tiên của họ, chữ lót và tên phải viết hoa, các ký tự còn lại viết thường).

   VD: " NgUyen VaN A " => "Nguyen Van A"
9. Không sử dụng các hàm có sẵn. Viết chương trình xóa N ký tự từ vị trí i trong chuỗi S.

   VD: "Nguyen Van A" i = 2 N = 3 (Xóa 3 ký tự từ vị trí 2) → "Nen Van A"
10. Viết chương trình nhập một số nguyên, xuất lại số đó ở dạng chuỗi nhưng có dấu "," ngăn cách hàng triệu, ngàn…

    VD: 123456789 → "123,456,789"
11. Viết chương trình nhập từ bàn phím 2 xâu ký tự S1 và S2. Hãy xét xem S1 có xuất hiện bao nhiêu lần trong S2 (hoặc ngược lại S2 xuất hiện bao nhiêu lần trong S1) và tại những vị trí nào?
12. Viết chương trình nhập một xâu S chỉ gồm các chữ cái thường. Hãy lập xâu S1 nhận được từ xâu S bằng cách sắp xếp lại các ký tự theo thứ tự abc.