TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

TIN HỌC ĐẠI CƯƠNG
Bài 11. Mảng và xâu kí tự

Nội dung

1. Mảng
2. Xâu kí tự

Nội dung

1. Mảng
   - 1.1. Khái niệm mảng
   - 1.2. Khai báo và sử dụng mảng
   - 1.3. Các thao tác cơ bản trên mảng
   - 1.4. Tìm kiếm trên mảng
   - 1.5. Sắp xếp trên mảng
2. Xâu kí tự

3

1.1. Khái niệm mảng

- Tập hợp hữu hạn các phần tử cùng kiểu, lưu trữ kế tiếp nhau trong bộ nhớ
- Các phần tử trong mảng có cùng tên (là tên mảng) nhưng phân biệt với nhau ở chỉ số cho biết vị trí của nó trong mảng
- Ví dụ:
  - Bảng điểm của sinh viên
  - Vector
  - Ma trận

4

1.2. Khai báo và sử dụng mảng

- Khai báo mảng (một chiều)

  KieuDuLieu tenMang [kích_thước];

- Trong đó
  - KieuDuLieu: kiểu dữ liệu của các phần tử trong mảng
  - tenMang: tên của mảng
  - kích_thước: số phần tử trong mảng
- Ví dụ

  int mangNguyen[10]; // khai báo mảng 10 phần tử có kiểu dữ liệu int

5

1.2. Khai báo và sử dụng mảng

- Cấp phát bộ nhớ
  - Các phần tử trong mảng được cấp phát các ô nhớ kế tiếp nhau trong bộ nhớ
  - Biến mảng lưu trữ địa chỉ ô nhớ đầu tiên trong vùng nhớ được cấp phát
- Ngôn ngữ C đánh chỉ số các phần tử trong mảng bắt đầu từ 0
  - Phần tử thứ i trong ***mangNguyen*** được xác định bởi ***mangNguyen [i-1]***

6

1.2. Khai báo và sử dụng mảng

- Khai báo mảng nhiều chiều

  KieuDuLieu tenMang[size$_1$][size$_2$]…[size$_k$];

  Trong đó
  - size$_i$ là kích thước chiều thứ i của mảng
- Mảng một chiều và mảng nhiều chiều
  - Mỗi phần tử của mảng cũng là một mảng

    => mảng nhiều chiều
- Ví dụ
  - int a[6][5] ; //mảng 2 chiều
  - int b[3][4][5]; // mảng 3 chiều

7

1.2. Khai báo và sử dụng mảng

- Sử dụng mảng
  - Truy cập vào phần tử thông qua tên mảng và chỉ số của phần tử trong mảng

    tenMang[chỉ_số_phần_tử]
  - Chú ý: chỉ số bắt đầu từ 0
- Ví dụ
  - int a[4];
  - phần tử đầu tiên (thứ nhất) của mảng: a[0]
  - phần tử cuối cùng (thứ tư) của mảng: a[3]
  - a[i]: là phần tử thứ i+1 của a

8

1.2. Khai báo và sử dụng mảng

- Ví dụ (tiếp)
  - int b[3][4];
  - phần tử đầu tiên của mảng: b[0] là một mảng một chiều
  - phần tử đầu tiên của mảng b[0]: b[0][0]
  - b[i][j]: là phần tử thứ j+1 của b[i], b[i] là phần tử thứ i+1 của b

9

Khai báo hằng số có kiểu mảng

- Sử dụng **`#define`**

```
#define TEN_MANG {Giá_trị_1, Giá_trị_2,... Giá_trị_n}
```

  - Lưu ý: không thể truy cập vào phần tử của mảng
- Sử dụng từ khóa **`const`**

```
const KieuDuLieu TEN_MANG[Kích_thước] = {Giá trị_1,
Giá trị_2, ..., Giá_trị_n};
```

Lưu ý:

  - Nếu không khai báo **`Kích_thước`** thì kích thước của mảng là số lượng giá trị sử dụng khi khai báo
  - Nếu số lượng giá trị nhỏ hơn **`Kích_thước`** mảng, các phần tử không được gán sẽ nhận giá trị 0

10

Khai báo hằng số có kiểu mảng – Ví dụ

const int CONST_ARR1[5] = {1,2,3,4,5}

CONST_ARR1:

1	2	3	4	5

const int CONST_ARR2[ ] = {1,2,3,4}

CONST_ARR2:

1	2	3	4

const int CONST_ARR3[5] = {1,2,3}

CONST_ARR23:

1	2	3	0	0

11

1.3. Các thao tác cơ bản trên mảng

a. Nhập dữ liệu cho mảng

- Khởi tạo giá trị cho mảng ngay khi khai báo
  - int a[4] = {1,4,6,2};
  - int b[2][3]={ {1,2,3}, {4,5,6} }:
  - Số lượng giá trị khởi tạo không được lớn hơn số lượng phần tử trong mảng
  - Nếu số lượng này nhỏ hơn, các phần tử còn lại được khởi tạo giá trị 0

12

1.3. Các thao tác cơ bản trên mảng

a. Nhập dữ liệu cho mảng

- Có thể xác định kích thước mảng thông qua số giá trị khởi tạo nếu để trống kích thước mảng
- int array1 [8] = {2, 4, 6, 8, 10, 12, 14, 16};
- int array2 [] = {2, 4, 6, 8, 10, 12, 14, 16};

13

1.3. Các thao tác cơ bản trên mảng

a. Nhập dữ liệu cho mảng

- Nhập dữ liệu từ bàn phím bằng hàm scanf
  - int a[10];
  - Nhập dữ liệu cho a[1]: scanf("%d", & a[1]);
  - Nhập dữ liệu cho toàn bộ phần tử của mảng a

    => Sử dụng vòng lặp for
- Lưu ý
  - Tên mảng là một hằng (hằng con trỏ) do đó không thể thực hiện phép toán với tên mảng như phép gán sau khi đã khai báo

14

1.3. Các thao tác cơ bản trên mảng

```
#include <stdio.h>
#define  MONTHS 12
int main(){
     int rainfall[MONTHS], i;
     for ( i=0; i < MONTHS; i++ ){
          printf("Nhap vao phan tu thu
                          %d: ", i+1);
          scanf("%d", &rainfall[i] );
     }
     return 0;
}
```

15

1.3. Các thao tác cơ bản trên mảng

a. Nhập dữ liệu cho mảng

- Lưu ý
  - Nếu số phần tử của mảng được nhập từ bàn phím và chỉ biết trước số phần tử tối đa tối đa => khai báo mảng với kích thước tối đa và sử dụng biến lưu số phần tử thực sự của mảng.
  - Ví dụ: Khai báo mảng số nguyên a có tối đa 100 phần tử. Nhập từ bàn phím số phần tử trong mảng và giá trị các phần tử đó….

16

1.3. Các thao tác cơ bản trên mảng

```
#include<stdio.h>
#include<conio.h>
int main(){
   int a[100];
   int n, i;
   do{
       printf("\n Cho biet so phan tu cua mang: ");
       scanf("%d",&n);
   }while (n>100||n<=0);
   for(i = 0; i < n; i++){
       printf("a[%d] = ", i);
       scanf("%d",&a[i]);
   }
   getch();
   return 0;
}
```

17

1.3. Các thao tác cơ bản trên mảng

b. Hiển thị dữ liệu trong mảng

- Dùng hàm printf()
- Để hiển thị tất cả các phần tử: dùng vòng for

- Ví dụ
  - Hiển thị một phần tử bất kì
  - Hiển thị tất cả các phần tử, mỗi phần tử trên một dòng
  - Hiển thị tất cả các phần tử trên một dòng, cách nhau 2 vị trí
  - Hiển thị từng k phần tử trên một dòng

18

1.3. Các thao tác cơ bản trên mảng

```
#include <stdio.h>
#define  MONTHS 12
int main(){
     int rainfall[MONTHS], i;
    for ( i=0; i < MONTHS; i++ ){
        printf("Nhap vao phan tu thu
                    %d: ", i+1);
        scanf("%d", &rainfall[i] );
     }
     for ( i=0; i < MONTHS; i++ )
        printf( "%2d " , rainfall[i]);
     printf("\n");
     return 0;
}
```

19

1.3. Các thao tác cơ bản trên mảng

c. Tìm giá trị lớn nhất, nhỏ nhất

- Tìm giá trị lớn nhất
  - Giả sử phần tử đó là phần tử đầu tiên
  - Lần lượt so sánh với các phần tử còn lại
  - Nếu lớn hơn hoặc bằng => so sánh tiếp
  - Nếu nhỏ hơn => coi phần tử này là phần tử lớn nhất và tiếp tục so sánh
  - Cách làm?
- Tìm giá trị nhỏ nhất: tương tự

20

1.4. Tìm kiếm trên mảng

- Bài toán
  - Cho mảng dữ liệu a và một giá trị k
  - Tìm các phần tử trong mảng a có giá trị bằng (giống) với k. Nếu có in ra vị trí (chỉ số) các phần tử này. Ngược lại thông báo không tìm thấy
- Cách làm
  - Duyệt toàn bộ các phần tử trong mảng
  - Nếu a[i] bằng (giống) k thì lưu lại chỉ số i
  - Sử dụng một biến để xác định tìm thấy hay không tìm thấy

21

1.4. Tìm kiếm trên mảng

- Phân tích
  - Duyệt toàn bộ các phần tử
    - Vòng lặp for (while, do while)
  - Lưu lại i nếu a[i] bằng (giống) k
    - Sử dụng mảng lưu chỉ số
  - Biến xác định tìm thấy hay không tìm thấy
    - Biến nhận giá trị 0 hoặc 1
    - Biến nhận giá trị 0 hoặc >=1 (tìm thấy thì tăng giá trị)

22

1.4. Tìm kiếm trên mảng

```
#include <stdio.h>
#include <conio.h>
int main(){
  int a[100], chi_so[100];
  int n;//n la số phần tử trong mảng
  int i, k, kiem_tra;
  printf("Nhap vao so phan tu cua mang:");
  scanf("%d",&n);
  printf("Nhap vao giá trị tim kiem:");
  scanf("%d",&k);
```

23

1.4. Tìm kiếm trên mảng

```
  kiem_tra = 0;
  // Duyệt qua tất cả các phần tử
  for(i = 0;i<n;i++)
     if(a[i] == k)
     {
          chi_so[kiem_tra] = i;
          kiem_tra ++;
     }
```

24

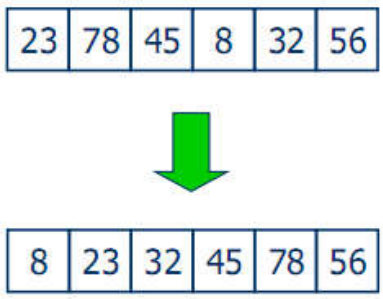
1.4. Tìm kiếm trên mảng

```
if(kiem_tra > 0){
  printf("Trong mang co %d phan tu co
  gia tri bang %d",kiem_tra,k);
  printf("\nChi so cua cac phan tula:");
  for(i = 0;i < kiem_tra;i++)
     printf("%3d",chi_so[i]);
} else
     printf("\n Trong mang khong co phan
          tu nao co gia tri bang %d",k);
  getch();
  return 0;
}
```

25

1.5. Sắp xếp mảng

- Bài toán
  - Cho mảng a gồm n phần tử. Sắp xếp các phần tử của mảng a theo thứ tự tăng dần/giảm dần



26

1.5. Sắp xếp mảng

- Giải thuật sắp xếp
  - Sắp xếp thêm dần (insertion sort)
  - Sắp xếp lựa chọn (selection sort)
  - Sắp xếp nổi bọt (bubble sort)
  - Sắp xếp vun đống (heap sort)
  - Sắp xếp nhanh (quick sort)
  - Sắp xếp trộn (merge sort)
  - ….

27

Giải thuật sắp xếp lựa chọn

Ý tưởng

- Lần sắp xếp thứ 1:
  - Đoạn đã được sắp xếp: chưa có phần tử nào
  - Đoạn chưa được sắp xếp: có n phần tử tử a_0, a_1,…, a_{n-1}
  - So sánh a_0 với a_j ($1 \le j \le n-1$). Nếu $a_0 > a_j$ thì đổi chỗ
  - Sau lượt sắp xếp này a_0 đã đúng thứ tự
- Lần sắp xếp thứ 2:
  - Đoạn đã được sắp xếp: a_0
  - Đoạn chưa được sắp xếp: a_1, a_2, …, a_{n-1}
  - So sánh a_1 với a_j ($2 \le j \le n-1$). Nếu $a_1 > a_j$ thì đổi chỗ
  - Sau lượt sắp xếp này a_1 đã đúng thứ tự

28

Giải thuật sắp xếp lựa chọn

- Lần sắp xếp thứ i:
  - Đoạn đã được sắp xếp: $a_0, a_1,\ldots,a_{i-2}$
  - Đoạn chưa được sắp xếp: $a_{i-1}, a_i, \ldots, a_{n-1}$
  - So sánh a_{i-1} với a_j ($i \le j \le n-1$). Nếu $a_i > a_j$ thì đổi chỗ
  - Sau lượt sắp xếp này a_{i-1} đã đúng thứ tự
- Thuật toán dừng khi dãy chưa được sắp xếp chỉ có 1 phần tử

29

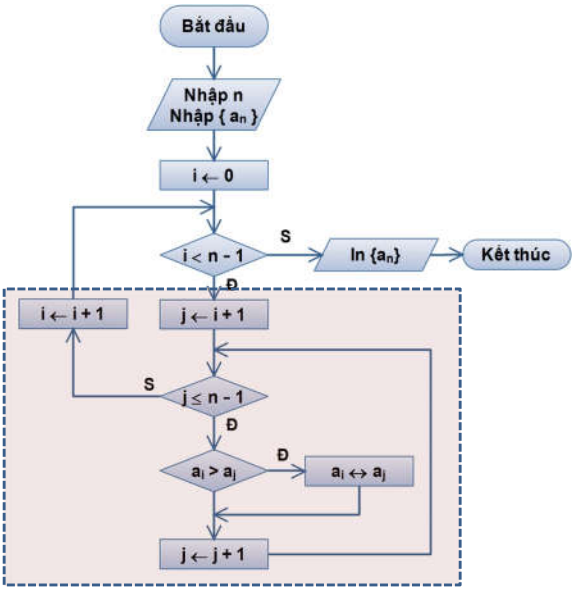
1.5. Sắp xếp mảng

- A = { 12, 5, 3, 4 };

	Lượt 1	Lượt 2	Lượt 3
12	3	3	3
5	12	4	4
3	5	12	5
4	4	5	12

30

Lưu đồ thuật toán sắp xếp lựa chọn



31

1.5. Sắp xếp mảng

```
//Khai bao cac bien
int a[100];
int i, j, tmp;

//Sap xep
for (i = 0; i < n-1; i++)
    for (j = i+1; j <n ; j++)
        if ( a[i] > a[j]){
            tmp = a[i];
            a[i]= a[j];
            a[j]= tmp;
       }
```

32

Nội dung

2.1. Khái niệm xâu kí tự

- Xâu kí tự (string) là một dãy các kí tự viết liên tiếp nhau
  - Ký tự kết thúc xâu: ‘0’ (mã ASCII là 0)
  - Độ dài xâu là số kí tự có trong xâu không gồm ký tự kết thúc xâu
- Ví dụ: “Tin hoc”, “String”
- Lưu trữ: kết thúc xâu bằng kí tự ‘\0’ hay NUL (mã ASCII là 0)

‘T’	‘i’	‘ n ‘	‘ ‘	‘h’	‘o’	‘c’	‘\0’

2.1. Khái niệm xâu kí tự

- So sánh
  - Xâu kí tự và mảng kí tự?
    - Tập hợp các kí tự viết liên tiếp nhau
    - Sự khác biệt: xâu kí tự có kí tự kết thúc xâu, mảng kí tự không có kí tự kết thúc xâu
  - Xâu kí tự "A" và kí tự 'A'?
    - 'A' là 1 kí tự
    - "A" là 1 xâu kí tự, ngoài kí tự 'A' còn có kí tự '\0' => gồm 2 kí tự

35

2.2. Khai báo và sử dụng xâu

a. Khai báo xâu

- Cú pháp

  char tenXau [kích_thước];

- Lưu ý:
  - Để lưu trữ một xâu có n kí tự chúng ta cần một mảng có kích thước n+1
- Ví dụ
  - Để lưu trữ xâu "Tin hoc" chúng ta phải khai báo xâu có số phần tử tối đa ít nhất là 8

    char str [8];

36

2.2. Khai báo và sử dụng xâu

a. Khai báo xâu

- Cú pháp

  char tenXau [kích_thước] = Giá_trị;

- Lưu ý:
  - *kích_thước* ≥ độ dài (***Giá_trị***) + 1
  - Có thể không cần khai báo *kích_thước*: khi đó kích thước biến ***tenXau*** là độ dài (***Giá_trị***) + 1
- Ví dụ

  char str1[10] = “BKHN”;

  char str2[5] = “SoICT”; //Lỗi

  char str3[ ] = “SoICT”; //Đúng

37

2.2. Khai báo và sử dụng xâu

b. Truy cập vào một phần tử của xâu

- Cú pháp:

  tenXau [chỉ_số_của_kí_tự]

- Ví dụ

  char quequan[10];

  Giả sử xâu này có nội dung là “Ha noi”

  ⇒ quequan[0] lưu trữ ‘H’

  quequan[1] ‘a’

  quequan[6] ‘i’

  quequan[7] ‘\0’

38

Khai báo hằng số có kiểu xâu ký tự

- Sử dụng **#define**

#define TEN_XAU Giá_trị

Ví dụ: `#define DAI_HOC "BKHN"`

- Sử dụng từ khóa const

const char TEN_XAU[Kích_thước] = Giá_trị;

Ví dụ: `const char DAI_HOC[5] = "BKHN";`

  - Khi khai báo với từ khóa **const**, kích thước phải đủ để chứa ký tự '\0'
    - Tốt hơn: không khai báo kích thước hằng xâu ký tự

39

2.3. Các hàm xử lý kí tự

- Tệp tiêu đề sử dụng: ctype.h
- char toupper(char ch): chuyển kí tự thường thành kí tự hoa

```
char ch = 'a';
ch = toupper(ch);// ch = 'A';
ch = toupper('B');//ch = 'B';
```

- char tolower(char ch): chuyển kí tự hoa thành kí tự thường

```
char ch = 'A';
ch = tolower(ch);// ch = 'a';
ch = tolower('b');//ch = 'b';
```

40

2.3. Các hàm xử lý kí tự

- int isalpha(char ch): kiểm tra xem kí tự có phải chữ cái hay không (‘a’…’z’,’A’,..’Z’)
- int isdigit(char ch): kiểm tra chữ số (‘0‘,‘1‘,..‘9‘)
- int islower(char ch): kiểm tra chữ thường
- int isupper(char ch): kiểm tra chữ hoa
- int iscntrl(char ch): kiểm tra kí tự điều khiển (0-31)
- int isspace(char ch): kiểm tra kí tự dấu cách (mã 32), xuống dòng (‘\n’ 10), đầu dòng (‘\r’ 13), tab ngang (‘\t’ 9), tab dọc (‘\v’ 11)

- trả về khác 0 nếu đúng, ngược lại trả về 0

41

2.3. Các hàm xử lý kí tự

```
#include<stdio.h>
#include<conio.h>
#include<ctype.h>
int main(){
    char ch;
    printf("Nhap vao mot ki tu: ");
    fflush(stdin);
    scanf("%c", &ch);
```

42

2.3. Các hàm xử lý kí tự

```
  if(isupper(ch)){
     printf("Ki tu nay la chu hoa\n");
     printf("Ki tu chu thuong tuong
          ung %c\n",tolower(ch));
  }else if(islower(ch)){
  printf("Ki tu nay la chu thuong\n");
  printf("Ki tu chu hoa tuong ung
  %c\n",toupper(ch));
  }
  getch();
  return 0;
}
```

43

2.3. Các hàm xử lý kí tự

Vào ra xâu kí tự

- Tệp tiêu đề: stdio.h
- Nhập xâu kí tự
  - ***gets(tenXâu);***
  - ***scanf("%s",&tenXau);***
- Hiển thị xâu kí tự
  - ***puts(tenXau);***
  - ***printf("%s", tenXau);***
- Sự khác nhau giữa gets và scanf?

44

2.4. Các hàm xử lý xâu kí tự

Tệp tiêu đề: string.h

- int strlen(char[] ten_xau): trả về độ dài xâu tính đến trước ký tự '\0' đầu tiên trong xâu

```
char s[] = "Tin hoc dai cuong";
int n;
n = strlen(s);// n = ?
s[7] = 0;
n = strlen(s);// n = ?
printf("Xau: %s",s)//Hiển thị?
```

45

2.4. Các hàm xử lý xâu kí tự

- strcpy(char[] xauDich, char[] xauNguon): sao chép nội dung xauNguon và xauDich
  - Lưu ý: không dùng phép gán giá trị cho các biến xâu

```
char s1[] = "Tin hoc", s2[10];

strcpy(s2, s1);// s2 = "Tin hoc"
strcpy(s1, "Tin hoc dai cuong");
```

46

2.4. Các hàm xử lý xâu kí tự

- int strcmp(char[] xâu_thứ_nhất, char[] xâu_thứ_hai): so sánh hai xâu
  - giá trị 0 : hai xâu giống nhau
  - giá trị<0: xâu thứ nhất nhỏ hơn xâu thứ hai
  - giá trị >0: xâu thứ nhất lớn hơn xâu thứ hai
- Quy tắc: so sánh lần lượt các ký tự từ đầu mỗi xâu.
  - Xâu nào xuất hiện ký tự có mã ASCII lớn hơn trước thì lớn hơn
  - Tất cả các ký tự giống nhau thì hai xâu bằng nhau
- Ví dụ: “Tin hoc” > “TIN hoc dai cuong”

  “Tin hoc” = “Tin hoc”

47

Hàm int strcmp( ) – Ví dụ

```
char s1[30], s2[20];
printf("Nhap xau thu nhat:");
fflush(stdin); gets(s1);
printf("Nhap xau thu hai:");
fflush(stdin); gets(s2);
if (strcmp(s1,s2)>0)
     printf ("Xau thu nhat lon hon");
else if(strcmp(s1,s2)<0)
          printf ("Xau thu hai lon hon");
     else printf ("Hai xau giong nhau");
```

Không so sánh trực tiếp 2 xâu

48

2.4. Các hàm xử lý xâu kí tự

- char[] strcat(char[ ] xauDich, char[ ] xauNguon): ghép nội dung xâu nguồn vào sau nội dung xâu đích

```
char s1[] = "Hello,", s2[20];
printf("Nhap ten:");
fflush(stdin); gets(s2);
strcat(s1, s2);//s1 = ?
strcat(s1,".Start game!"); //s1 = ?
```

49

Thảo luận

50