Nội dung

1. Khái niệm
2. File nhị phân
3. File văn bản

1. Khái niệm tệp tin

- Tệp tin là một tài nguyên dùng để lưu trữ thông tin lâu dài, sử dụng cho các chương trình máy tính

2. Khai báo tệp tin

- FILE *tên_con_trỏ_tệp;
- Ví dụ: FILE *fp;
- Khai báo một con trỏ file tên fp
- Chú ý: tệp tin nên tạo trong thư mục chương trình của mình luôn

3

3. Các loại tệp tin

- Có 2 loại file mà ta sẽ sử dụng đó là:
- File văn bản (text) :
  - Dữ liệu là một chuỗi kí tự liên tục.
  - Chúng ta sẽ thao tác trên file text (*.txt).

    Ví dụ: Các tập tin: *.cpp, *.txt, ... là các tập tin văn bản.

- File nhị phân (binary)
  - Dữ liệu được xem như là một dãy byte liên tục.
  - Chuyển đổi dữ liệu tùy thuộc vào biến lưu trữ khi đọc/ghi.

Khái niệm file nhị phân

- File nhị phân (hay binary file) là một tập tin trên máy tính, file có thể lưu bất kỳ dạng dữ liệu nào và được mã hóa dưới dạng nhị phân để lưu trữ trong máy tính.
- Tập tin nhị phân thường được coi là một chuỗi các byte (các số nhị phân được nhóm theo nhóm cứ 8 bit thành 1 byte).
- File nhị phân có thể lưu trữ được nhiều dạng dữ liệu như file txt, file nhạc, file ảnh…

4. Các bước xử lý tập tin

- Bước 1: Mở tập tin để chuẩn bị làm việc.
- Bước 2: Thực hiện thao tác đọc/ghi tập tin: Đọc dữ liệu từ tập tin vào biến nhớ hoặc ghi dữ liệu từ biến nhớ vào tập tin.
- Bước 3: Đóng tập tin.

5. Thao tác trên tệp tin

- Mở file
- Đọc file
- Ghi dữ liệu vào file
- Đóng file

❖Thư viện hỗ trợ tệp tin: #include<stdio.h>

a. Mở file văn bản

Tên_con trỏ_tệp=fopen(“filePath”, “mode”);

- filePath: đường dẫn file
- mode: chế độ mở file.

❖Các mod trong C:

- “r”: Mở file để đọc (read).
- “w”: Tạo file mới để ghi (write). Nếu file này đã tồn tại trên đĩa thì bị ghi đè.
- “a”: Mở để ghi vào cuối file nếu file này đã tồn tại, nếu file này chưa có thì sẽ được tạo mới để ghi (append).
- “r+”: Mở file đã có để cập nhật (cả đọc lẫn ghi).
- “w+”:Mở file mới để được cập nhật (cả đọc lẫn ghi). Nếu file này đã có sẽ bị ghi đè
- “a+”: Mở để đọc và cập nhật (ghi) vào cuối file. Sẽ tạo mới nếu file này chưa có.

8

- “rt”: Mở một tập tin văn bản để đọc. Tập tin cần tồn tại nếu không sẽ có lỗi.
- “wt”: Mở một tập tin văn bản mới để ghi. Nếu tập tin đã tồn tại nó sẽ bị xóa.
- “at”: Mở một tập tin văn bản để ghi thêm. Nếu tập tin chưa tồn tại thì sẽ tạo tập tin mới.
- “r+t” : Mở một tập tin văn bản để đọc/ghi. Tập tin cần tồn tại nếu không sẽ có lỗi.

9

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
        FILE *f;
        char ht[100]=“Ninh Trang";
        f=fopen("data.txt","w");
        fprintf(f,"%s",ht);
    return 0;
}
```


10

b. Đóng file

- Sau khi thao tác với file xong, chúng ta sẽ đóng file bằng lệnh:
  - fclose(tên_con trỏ_tệp);
  - Việc đóng file sẽ giúp chúng ta bảo toàn được dữ liệu đang tiến hành lưu trữ và tránh khỏi một số lỗi không đáng có

5. Các hàm đọc /ghi tệp văn bản

A. Đọc file:

- Dữ liệu của chương trình được đọc ra từ tệp tin.
- Hàm fscanf() – Đọc file
- Hàm fscanf () được sử dụng để đọc tập hợp các ký tự từ file. Nó đọc một từ trong file và trả về EOF ở vị trí kết thúc file.
- Cú pháp: fscanf(<tên tệp>, “chuỗi định dạng”, &tên_biến)
- Chú ý: khi đọc dữ liệu từ file chương trình hàm thì các biến để là tham biến con trỏ (tham biến chiếu)

12

b. Ghi tệp tin

B. Ghi file:

- Dữ liệu của chương trình sẽ ghi lại xuống tệp tin và lưu lại đó và sau này cần thì mở ra
- Hàm fprintf() – Ghi file
- Hàm fprintf() trong C được sử dụng để ghi các ký tự vào file. Nó gửi dữ liệu được định dạng tới một stream.
- Cú pháp: fprintf(<tên_tệp>, “chuỗi định dạng”, tên_biến)

13

- **int fputs(char *s, FILE *fp);**

  Ghi chuỗi s lên tập tin fp (ký tự ‘\0’ không ghi lên tập tin.
- *Hàm fputc();*

  Chức năng**:** Ghi một ký tự vào tệp

- ***char *fgets(char *s, int n, FILE *fp);***

  Đọc một dãy ký tự từ tập tin fp và lưu vào chuỗi s.

- hàm: ***int feof(FILE *fp);*** với fp là con trỏ tệp

  Công dụng: Dùng để kiểm tra cuối tệp. Hàm cho giá trị khác không nếu chưa đến cuối tệp, và cho giá trị bằng 0 nếu ở vị trí cuối tệp

- void ***rewind(FILE *fp);*** với đối fp là con trỏ tệp

  Công dụng: Chuyển con trỏ tệp về vị trí đầu tệp.

VÍ dụ 3

- Tạo file input.txt có chứa 2 số nguyên
- Hãy đọc dữ liệu ra hai số a, b và tính tổng , hiệu và ghi tổng, hiệu vào cuối file input.txt

16

Đọc vào tệp

```
#include<stdio.h>
int main(){
    FILE *f;
    int a, b;
    f=fopen("input.txt","w");
    printf("\nNhap a=
");scanf("%d",&a);
    printf("\nNhap b=
");scanf("%d",&b);
   fprintf(f,"%d",a);
   fprintf(f,"%d",b);
    fclose(f);
```

```
FILE *f1;
    f1=fopen("input.txt","a");
    int tong=a+b;
    int hieu=a-b;
    int tich=a*b;
    fprintf(f1,"\n%d+%d=
%d",a,b,tong);
    fprintf(f1,"\n%d-%d=
%d",a,b,hieu);
    fclose(f1);

    return 0;
}
```


17

Ví dụ 4: đọc tệp mảng 2 chiều

- Cho tệp matran.txt có dạng sau:

3

4

1 2 3 4

2 3 4 5

4 5 6 7

Hãy đọc số dòng cột, và mà trận in ra màn hình

18

Ví dụ 4

```
#include <stdio.h>
main() {
   FILE *fp;
   int row, col, i, j;
   int matrix[10][10];
   fp = fopen("matran.txt", "r");
   // doc so hang cua ma tran
   fscanf(fp, "%d", &row);
   // doc so cot cua ma tran
   fscanf(fp, "%d", &col);
 printf("So hang cua ma tran: %d\n", row);
   printf("So cot cua ma tran: %d\n", col);
   // doc noi dung ma tran
    for (i = 0; i < row; i++) {
          for (j = 0; j < col; j++) {
               fscanf(fp, "%d", &matrix[i][j]);
                    }
   }
   printf("\nMa tran: \n");
   for (i = 0; i < row; i++) {
   for (j = 0; j < col; j++) {
   printf("%d ", matrix[i][j]);
                    }
             printf("\n");   }

   fclose(fp);
}
```

A. ghi file nhị phân

- ***int fwrite(void *ptr, int size, int n, FILE *fp);***
- Ghi n phần tử, mỗi phần tử có kích thước là size byte, từ vùng nhớ được trỏ bởi con trỏ ptr lên tập tin fp.
- Hàm trả về một giá trị bằng số phần tử thực sự ghi được.

- ***int fread(void *ptr, int size, int n, FILE *fp);***
  - Đọc n phần tử, mỗi phần tử có kích thước là size byte, từ tập tin fp
- ***void rewind(FILE *fp);***
  - Di chuyển con trỏ tập tin về đầu tập tin fp. Khi đó việc truy xuất trên tập tin được thực hiện từ đầu tập tin.

- ***long ftell(FILE *fp);***
  - Nếu thành công hàm trả về vị trí hiện tại của con trỏ tệp tin. Có thể sử dụng hàm ftell() để lấy tổng kích thước của một file sau khi di chuyển con trỏ tệp ở cuối tệp.

int fseek(FILE *fp, long sb, int xp);

Trong đó:

fp: là con trỏ tệp

sb là số byte cần di chuyển

xp: cho biết vị trí xuất phát mà việc dịch chuyển được bắt đầu từ đó

xp có thể nhận:

xp=SEEK_SET hay 0: xuất phát từ đầu tệp

xp=SEEK_CUR hay 1: xuất phát từ vị trí hiện tại của con trỏ

xp=SEEK_END hay 2: xuất phát từ cuối tệp

Công dụng: chuyển vị trí con trỏ chỉ vị của tệp fp về vị trí xác định bởi xp qua một số byte xác định. Chiều di chuyển về cuối tệp nếu sb>0

23

Ví dụ 5

```
#include <stdio.h>
  main() {
 FILE *fp;
 fp = fopen("myfile.txt","w+");
 fputs("Hello C", fp);
 fseek( fp, 6, SEEK_SET );
 fputs("Java", fp);
 fclose(fp);
}
```

- Kết quả: Hello Java

24

Mở file nhị phân:

- ***FILE *fopen(char *tentaptin, char *mod)***
- Mod gồm:
  - "rb": Mở một tập tin nhị phân để đọc. Tập tin cần tồn tại nếu không sẽ có lỗi.
  - "wb": Mở một tập tin nhị phân mới để ghi. Nếu tập tin đã tồn tại nó sẽ bị xóa.
  - "ab": Mở một tập tin nhị phân để ghi thêm. Nếu tập tin chưa tồn tại thì sẽ tạo tập tin mới.
  - "r+b" : Mở một tập tin nhị phân để đọc/ghi. Tập tin cần tồn tại nếu không sẽ có lỗi.
  - "w+b": Mở một tập tin nhị phân mới để đọc/ghi. Nếu tập tin đã tồn tại nó sẽ bị xóa.
  - "a+b": Mở một tập tin nhị phân để đọc/ghi thêm. Nếu tập tin chưa tồn tại thì sẽ tạo tập tin mới

25

Bài tập

- Viết chương trình đọc file matrix.txt với nội dung như sau:

```
3
4   2   3
4   7   6
3   8   2
```

3 là kích thước ma trận vuông

- Xuất ma trận vừa đọc từ file ra màn hình
- Sắp xếp ma trận giảm dần
- Ghi ma trận vừa sắp xếp vào file matrix1.txt

26

Hàm sắp xếp

```
#include <stdio.h>

void Sapxep(int a[],int n){
        int i,j,tam;
        for(i=0;i<n*n-1;i++)
                for(j=i+1;j<n*n;j++)
                        if (a[i]<a[j]){
                                tam=a[i];
                                a[i]=a[j];
                                a[j]=tam;
                        }
}
```

Đọc ma trận vào mảng a

```
main() {
   FILE *fi, *fo;
   int n, i, j;
   int a[1000];
   fi = fopen("matrix.txt", "r");
   fscanf(fi, "%d\n", &n);
            printf("n=%d",n);
   // doc noi dung ma tran
    for (i=0;i<n;i++)
            {
                     for (j=0;j<n;j++)
                              fscanf(fi,"%d",&a[i*n+j]);
            }
```

```
printf("\nMa tran: \n");
  for (i=0;i<n;i++)          {
                    for (j=0;j<n;j++)
                              printf(" %d",a[i*n+j]);
                    printf("\n");
          }
          Sapxep(a,n);
          fo = fopen("matrix1.txt", "w+");
          for (i=0;i<n;i++) {
                    for (j=0;j<n;j++)
                              fprintf(fo," %d",a[i*n+j]);
                    fprintf(fo,"\n");
          }
          fclose(fi);          fclose(fo);
}
```