
HUST
ĐẠI HỌC BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY
ONE LOVE. ONE FUTURE.

File – tệp tin

ONE LOVE. ONE FUTURE.

Nội dung

- Khái niệm cơ bản
- Đóng và mở tệp
- Một số thao tác với tệp
- Xử lý file theo từng ký tự
- Xử lý file văn bản theo dòng
- Đọc/ghi dữ liệu theo định dạng

Làm việc với tập tin

- Tập tin (file) cho phép lưu trữ thông tin lâu dài trong bộ nhớ ngoài.
- Chương trình không chỉ nhập dữ liệu từ bàn phím, hiển thị dữ liệu trên màn hình mà có thể nhập dữ liệu từ tập tin, ghi dữ liệu ra tập tin.
- Ngôn ngữ C cho phép giao tiếp với các tập tin thông qua một thực thể đặc biệt gọi là con trỏ (kiểu) file.
  - Các hàm đọc, ghi tập tin sử dụng con trỏ file là đối số
  - Sau mỗi thao tác đọc/ghi vị trí của con trỏ file sẽ thay đổi.

- Khai báo: FILE *fptr;

Các thao tác cơ bản khi làm việc với tập tin

- Mở tập tin
- Đọc dữ liệu từ tập tin vào bộ nhớ chương trình (cụ thể là vào các biến)
- Ghi (xuất) dữ liệu từ bộ nhớ chương trình ra tập tin
- Đóng tập tin

File văn bản và File nhị phân

- File văn bản:
  - có nội dung là văn bản chứa các ký tự nhìn thấy được.
  - có thể được tạo ra bằng cách dùng các phần mềm thông dụng như Notepad, Notepad++, Sublime Text,…
  - thuận tiện trong việc sử dụng hàng ngày, nhưng kém bảo mật và cần nhiều bộ nhớ để lưu trữ hơn.

- File nhị phân
  - Nội dung lưu trữ dưới dạng nhị phân (chuỗi 0 và 1)
  - Bảo mật và tiết kiệm không gian lưu trữ hơn.

Mở tập tin

- Sử dụng hàm fopen()
- Nguyên mẫu: FILE *fopen(const char *filename, const char *mode);

```
FILE *fptr;
if ((fptr = fopen("test.txt", "r")) == NULL){
    printf("Cannot open test.txt file.\n");
    exit(1);
}
```

Mở tập tin

- filename: đường dẫn và tên tập tin.
  - Nhận giá trị là một giá trị xâu ký tự: **“data.txt”**
  - Có thể chứa đường dẫn đầy đủ tới tập tin: **“/root/hedspi/CProgrammingBasic/Lab1/data.txt”**
  - Có thể dùng một biến mảng ký tự để lưu trữ:

    char file_name[] = “junk.txt”;

- ***Lưu ý:*** *Nếu đường dẫn không được mô tả, tập tin mặc định nằm tại cùng thư mục chứa chương trình.*

Các tham số mode cho tập tin văn bản

mode	Ý nghĩa
"r"	Mở một tập tin văn bản đã có chỉ để đọc. Nếu tập tin không tồn tại, fopen() trả về NULL.
"w"	Mở một tập tin văn bản chỉ để ghi. Nếu file đã tồn tại, nội dung sẽ bị ghi đè. Nếu file không tồn tại, nó sẽ được tạo tự động.
"a"	Mở một tập tin văn bản đã có để ghi thêm vào cuối.
"r+"	Mở một tập tin văn bản đã có cho phép cả đọc và ghi. Nếu tập tin không tồn tại, fopen() trả về NULL.
"w+"	Mở file văn bản cho phép cả đọc và ghi.
"a+"	Mở file văn bản cho phép cả đọc và ghi « append »

Các tham số mode cho tập tin nhị phân

mode	**Ý nghĩa**
"rb"	Mở tập tin nhị phân đã có chỉ để đọc.
"wb"	Mở tập tin nhị phân chỉ để ghi.
"ab"	Mở tập tin nhị phân đã có để ghi thêm vào cuối.
"r+b"	Mở tập tin nhị phân đã có cho phép cả đọc và ghi.
"w+b"	Mở tập tin nhị phân cho phép cả đọc và ghi. Nếu file đã tồn tại, nội dung sẽ bị ghi đè. Nếu file không tồn tại, nó sẽ được tạo tự động.
"a+b"	Mở hoặc tạo tập tin nhị phân cho phép cả đọc và ghi vào cuối.

Đóng tập tin

- Hàm fclose được sử dụng để ngắt kết nối giữa một con trỏ tập tin với tập tin mà nó đang tham chiếu tới.

- int fclose(FILE *stream);

- Cần đóng tập tin khi đã hoàn tất các thao tác đọc ghi với tập tin.

Ví dụ

- Ví dụ 1. Mở và đóng tệp

```
#include <stdio.h>
enum { SUCCESS, FAIL };
main() {
   FILE* fptr;
   char filename[] = "haiku.txt";
   int reval = SUCCESS;
   if ((fptr = fopen(filename, "r")) == NULL) {
      printf("Cannot open %s.\n", filename);
      reval = FAIL;
   }
   else {
      printf("The value of fptr: 0x%p\n", fptr);
      printf("Ready to close the file.");
      fclose(fptr);
   }
   return reval;
}
```

Xử lý file theo ký tự

Đọc và ghi với tập tin (I)

- Trong ngôn ngữ C, chương trình có thể thực hiện các thao tác vào/ra (đọc/ghi) theo những cách thức khác nhau:
  - **Đọc hoặc ghi mỗi lần một ký tự**
  - Đọc hoặc ghi mỗi lần một dòng văn bản.
  - Đọc hoặc ghi một khối (block) các ký tự (byte) mỗi lần.

- **Đọc hoặc ghi mỗi lần một ký tự.**
  - Cặp đôi hàm định nghĩa trong thư viện stdio.h: fgetc() and fputc()
  - Nguyên mẫu:
    - int fgetc(FILE *stream);
    - int fputc(int c , FILE *stream);
- Ký tự EOF: khi đến cuối tập tin, các hàm trên trả về EOF.

Ví dụ

- Ví dụ 1. sao chép nội dung file
- Tạo một file văn bản với tên lab1.txt với nội dung bất kỳ, lưu trong thư mục cùng với chương trình.
- Viết chương trình đọc từ file trên mỗi lần một ký tự, sau đó ghi chúng vào một file mới với tên lab1w.txt

```
#include <stdio.h>
enum { SUCCESS, FAIL };

void CharReadWrite(FILE* fin, FILE* fout)
{
   int c;
   while ((c = fgetc(fin)) != EOF) {
      fputc(c, fout);  /* write to a file */
      putchar(c);
      /* display character on the screen */
   }
}
```

Ví dụ

```
main(void) {
   FILE* fptr1, * fptr2;
   char filename1[] = "lab1a.txt";
   char filename2[] = "lab1.txt";
   int reval = SUCCESS;

   if ((fptr1 = fopen(filename1, "w")) == NULL) {
      printf("Cannot open %s.\n", filename1);
      reval = FAIL;
   }
   else if ((fptr2 = fopen(filename2, "r")) == NULL) {
      printf("Cannot open %s.\n", filename2);
      reval = FAIL;
   }
   else {
      CharReadWrite(fptr2, fptr1);
      fclose(fptr1);
      fclose(fptr2);
   }
   return reval;
}
```

Ví dụ

- Ví dụ 2. Viết chương trình đọc nội dung từ một tập tin văn bản, mỗi lần đọc một ký tự.
- Chương trình sẽ chuyển ký tự chữ cái hoa thành ký tự chữ cái thường và ngược lại, sau đó ghi vào một tập tin khác.
- Chú ý với các ký tự khác – chương trình vẫn thực hiện sao chép một cách thông thường sang tập tin mới.

```
void CharReadWrite(FILE* fin, FILE* fout)
{
   int c;
   while ((c = fgetc(fin)) != EOF) {
      if islower(c) c = toupper(c);
      else if isupper(c) c = tolower(c);
      fputc(c, fout);  /* write to a file */
      putchar(c); /* display character on the screen */
   }
}
```

Bài tập

- Bài tập 1. Viết chương trình có tên mycp hoạt động tương tự lệnh cp trong các hệ điều hành UNIX/LINUX. Nó có thể sao chép một tập tin văn bản sang một tập tin mới theo cú pháp:
  - mycp <tập_tin_1> <tập_tin_2>

- Đường dẫn, tên các tập tin được cung cấp dưới dạng đối số dòng lệnh.
- *Chú ý: Chương trình phải kiểm tra cú pháp sử dụng (vd số đối số – thông báo lỗi và hiển thị hướng dẫn khi cần..)*

Bài tập

- Bài tập 2: Viết chương trình nhận tên hai tập tin ở đối số dòng lệnh, sau đó tiến hành ghép nội dung của tập tin thứ hai vào cuối tập tin thứ nhất. Giả sử cả hai tập tin đều tồn tại.
- Cú pháp sử dụng:
  - apd <file1> <file2>

- *Chú ý: Chương trình phải kiểm tra cú pháp sử dụng (vd số đối số – thông báo lỗi và hiển thị hướng dẫn khi cần..)*

Bài tập

- Bài tập 3. Viết chương trình có tên uconvert có chức năng chuyển đổi tất cả các chữ cái trong nội dung một tập tin cụ thể (được cung cấp trong đối số dòng lệnh) thành chữ hoa và ghi lại nội dung mới vào chính tập tin đó.
- Cú pháp: uconvert tata.txt
- Ví dụ
  - File nguồn tata.txt: helloworld
  - Nội dung tata.txt sau khi chạy chương trình : HELLOWORD.

Bài tập

- Bài tập 4. Viết một chương trình có thể sử dụng cùng một lúc hai chức năng mã hóa và giải mã một tập tin văn bản sử dụng mật mã Caesar (mã hóa cộng) như sau. Chương trình nhận ba đối số:
  - <tập tin nguồn> <độ dịch chuyển> < tập tin đích>
- Khi cần mã hóa, chạy chương trình với độ dịch chuyển (offset) n là một số nguyên dương. Chương trình sẽ thay thế mỗi ký tự trong tập tin bởi một ký tự đứng sau nó n vị trí trong bảng mã ASCII. Ví dụ với offset =3 thì A→D, B→ E
- Khi giải mã, chạy chương trình với đầu vào là tập tin mã hóa và giá trị độ dịch chuyển là số âm tương ứng (VD offset = -3)
- Chức năng nâng cao (tùy chọn): Với các ký tự là chữ cái thực hiện dịch chuyển vòng tròn: A-> D, ..., Z -> C

Xử lý file theo dòng

Thao tác với tập tin theo dòng

- Sử dụng hai hàm: fgets() and fputs()
- char *fgets(char *s, int n, FILE *stream);
  - s : tham số ứng với xâu ký tự dùng để lưu nội dùng dòng đọc từ tập tin.
  - n : độ dài xâu ký tự s – tính cả ký tự NULL.
- Hàm fgets() dừng khi thỏa mãn một trong các điều kiện sau: đọc được n-1 ký tự từ tập tin, gặp ký tự xuống dòng mới hoặc EOF. Sau đó thêm với ký tự null vào cuối xâu s.
- Hàm: int fputs(const char *s, FILE *stream);
  - s: Xâu ký tự cần ghi ra tập tin
  - stream: con trỏ file
- Kết quả trả về
  - 0 nếu thao tác thành công
  - khác 0 nếu thất bại.

Ví dụ

- Ví dụ 1. Thực hiện lại bài tập lập trình sao chép nội dung tập tin, tuy nhiên thay vì sử dụng cặp hàm fgetc và fputc – ta sử dụng cặp hàm fgets và fputs để đọc từ tập tin và ghi vào tập tin mỗi lần một dòng trong nội dung văn bản.

```
#include <stdio.h>
enum { SUCCESS, FAIL };
#define MAX_LEN 81

void LineReadWrite(FILE* fin, FILE* fout)
{
   char buff[MAX_LEN];
   while (fgets(buff, MAX_LEN, fin) != NULL) {
      fputs(buff, fout);
      printf("%s", buff);
   }
}
```

Ví dụ

```
main(void) {
   FILE* fptr1, * fptr2;
   char filename1[] = "lab1a.txt";
   char filename2[] = "lab1.txt";
   int reval = SUCCESS;

   if ((fptr1 = fopen(filename1, "w")) == NULL) {
      printf("Cannot open %s.\n", filename1);
      reval = FAIL;
   }
   else if ((fptr2 = fopen(filename2, "r")) == NULL) {
      printf("Cannot open %s.\n", filename2);
      reval = FAIL;
   }
   else {
      LineReadWrite(fptr2, fptr1);
      fclose(fptr1);
      fclose(fptr2);
   }
   return reval;
}
```

Ví dụ

- Ví dụ 2. Sửa chương trình sao chép tập tin ở slide trước để chương trình chỉ hiện nội dung tập tin ra màn hình, sau đó hiển thị số các dòng văn bản.
- Minh họa về giao diện của chương trình:

  Reading file Haiku.txt…. done!

  Haiku haiku

  Tokyo

  Hanoi

  This file has 3 lines.

- Sửa mã nguồn hàm LineReadWrite:
  - Loại bỏ lệnh sử dụng fputs
  - Tăng bộ đếm số dòng văn bản mỗi lần đọc một dòng.

Ví dụ

```
enum { SUCCESS, FAIL };
int LineReadWrite(FILE* fin)
{
   char buff[MAX_LEN]; int count = 0;
   while (fgets(buff, MAX_LEN, fin) != NULL) {
      count++; printf("%s", buff);
   }
   return count;
}
```

```
main() {
   FILE* fptr1; int c = 0;
   char filename1[] = "haiku.txt";
      int reval = SUCCESS;

   if (fptr1 = fopen(filename1, "r")) == NULL){
      printf("Cannot open %s.\n", filename1);
      reval = FAIL;
   }
   else {
      printf("Reading file % s … done!\n", filename1);
      c = LineReadWrite(fptr2, fptr1);
      printf("The file has % d lines.\n", c);
      fclose(fptr1);
   }
   return reval;
}
```

Bài tập

- Bài tập 1. Viết chương trình mycat đọc và hiển thị nội dung một tập tin văn bản trên màn hình. Chương trình hỗ trợ hai cú pháp sử dụng như sau:
- cat <filename> : Hiển thị một lần toàn bộ nội dung
- cat <filename> -p : Hiển thị theo từng trang. Người dùng chạm một phím để xem trang tiếp theo.

- Bài tập 2. Viết chương trình nhận đối số dòng lệnh là đường dẫn đến một file văn bản (nội dung dưới 80 dòng). Chương trình thêm một dòng mới vào cuối file nói trên với nội dung chứa các ký tự đầu tiên của các dòng trong file ban đầu.

Đọc ghi file văn bản có định dạng

Đọc ghi file văn bản có định dạng

- Đây là các hàm hữu ích để xử lý dữ liệu có cấu trúc (thuộc các kiểu dữ liệu khác nhau) từ văn bản.

- int fscanf(FILE *stream, const char *format, ...);
  - Hàm fscanf hoạt động tương tự hàm scanf, điểm khác biệt là nó đọc nội dung từ tập tin (đại diện bởi con trỏ file) để ghi vào các biến.

- int fprintf(FILE *stream, const char *format, ...);
  - Tương tự như hàm printf nhưng thay vì đưa nội dung ra màn hình thì nó ghi nội dung từ các đối số ra tập tin (đại diện bởi con trỏ file).

Ví dụ

- Ví dụ 1. Viết chương trình đọc từng dòng văn bản từ một tập tin, sau đó tính độ dài xâu ký tự trên mỗi dòng và ghi ra một tập tin mới theo định dạng sau: <độ dài dòng> <Nội dung dòng>
- Ví dụ, với một dòng trong tập tin đầu vào:

  The quick brown fox jumps over the lazy dog.

  trong tập tin đầu ra ở dòng tương ứng sẽ là:

  44 The quick brown fox jumps over the lazy dog.

```
void LineReadWrite(FILE* fin, FILE* fout)
{
   char buff[MAX_LEN];
   int len;
   while (fgets(buff, MAX_LEN, fin) != NULL) {
      len = strlen(buff) - 1;
      fprintf(fout, "%d %s", len, buff);
      printf("%s", buff);
   }
}
```

Ví dụ

- Ví dụ 2. Viết chương trình để đọc một dãy số nhập từ bàn phím và ghi chúng ra tệp "out.txt" theo thứ tự ngược lại. Ngoài ra, tổng các số được ghi vào cuối file.
- Cú pháp nhập liệu từ bàn phím như sau: Số đầu tiên là số lượng các số trong dãy sẽ nhập, sau đó là dãy các số nguyên. Ví dụ: khi người dùng nhập: 4 12 -45 56 3
- "4" là số các số sẽ được nhập, bao gồm "12 -45 56 3". Nội dung của tập tin "out.txt" sẽ là, với 26 là tổng của 4 số
- 3 56 -45 12 26
- Vì số lượng các số nhập thay đổi theo mỗi lần chạy, chương trình cần cấp phát động bộ nhớ cho các số này sử dụng hàm malloc( ).

Ví dụ

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

enum { SUCCESS, FAIL };

int main(void)
{
   FILE* fp;
   int* p;
   int i, n, value, sum;
   int reval = SUCCESS;

   printf("Enter a list of numbers with the first is the size of list: \n");
   scanf("%d", &n);
   p = (int*)malloc(n * sizeof(int)); i = 0; sum = 0;
   while (i < n) {
      scanf("%d", &value);
      p[i++] = value;
      sum += value;
   }
```

Ví dụ

```
    if ((fp = fopen("out.txt", "w")) == NULL) {
        printf("Can not open %s.\n", "out.txt");
        reval = FAIL;
    }
    for (i = n - 1; i >= 0; i--) {
        fprintf(fp, "%d ", p[i]);
    }
    fprintf(fp, "%d ", sum);
    fclose(fp);
    free(p);
    return reval;
}
```

Ví dụ

- Ví dụ 3. Tạo một tập tin văn bản có tên product.txt, mỗi dòng trong đó chứa thông tin về một sản phẩm: ID (kiểu int), Product Name (xâu ký tự không chứa ký tự trắng), Price (kiểu double). Các trường dữ liệu trên phân tách với nhau bởi một ký tự space hoặc tab. Ví dụ
  - 1 Samsung_Television_4K 20000000
  - 2 Apple_MacBook_2020 18560000
- Viết chương trình đọc tập tin trên vào một mảng các phần tử cấu trúc và sau đó hiện nội dung mảng trên ra màn hình dưới dạng:
  - No Product Name Price
  - 1 Samsung_Television_4K 20000000
  - …
- Gợi ý
  - Khi đọc số thực double dùng fscanf với xâu định dạng "%lf"
  - Trong trường hợp các trường dữ liệu được phân tách bởi các ký hiệu như ; hay , (delimiter), có thể kết hợp sử dụng fscanf và fgetc để đọc được từng trường
  - 1000, John_Allan, 28, NewYork

Ví dụ

```
#include <stdio.h>
enum { SUCCESS, FAIL };
#define MAX_ELEMENT 10
typedef struct {
   int no;
   char name[20];
   double price;
}product;

int main(void) {
   FILE* fp;
   product arr[MAX_ELEMENT];
   int i = 0, n;
   int reval = SUCCESS;
   printf("Loading file...\n");
   if ((fp = fopen("product.txt", "r")) == NULL) {
      printf("Can not open %s.\n", "product.txt");
      reval = FAIL;
   }
```

Ví dụ

```
  else {
     while (fscanf(fp, "%d%s%lf", &arr[i].no, arr[i].name, &arr[i].price) != EOF) {
        //printf("%-6d%-24s%-6.2f\n",arr[i].no,arr[i].name,arr[i].price);
        i++;
     }
     n = i;
     for (i = 0; i < n; i++)  printf("%-6d%-24s%-6.2f\n",
                  arr[i].no, arr[i].name, arr[i].price);
  }
  fclose(fp);
  return reval;
}
```

Bài tập

- Bài tập 1. Tạo một file văn bản nội dung là danh sách lớp gồm ít nhất 6 sinh viên. Mỗi dòng gồm 4 trường sau:
- STT(số thứ tự) Mã số sinh viên Họ và tên (không chứa ký tự trắng) Số điện thoại. Ví dụ
  - 1 20181110 Bui_Van 0903112234
  - 2 20182111 Joshua_Kim 0912123232
- Viết chương trình đọc tập tin trên vào một mảng các cấu trúc phù hợp. Chương trình yêu cầu nhập bổ sung thêm trường điểm cho mỗi sinh viên sau đó ghi lại kết quả vào tập tin bangdiem.txt (transcript.txt) gồm tất cả các trường nói trên (cùng trường điểm).