



TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN TP.HCM  
KHOA CÔNG NGHỆ THÔNG TIN  
BỘ MÔN CÔNG NGHỆ PHẦN MỀM  
HỆ CHÍNH QUY  
MÔN: **NHẬP MÔN LẬP TRÌNH**

## **HƯỚNG DẪN THỰC HÀNH TUẦN 11**

### **KIỂU TẬP TIN**

 NGUYỄN LÊ HOÀNG DŨNG

TP.HCM, ngày 30 tháng 12 năm 2020

## MỤC LỤC

1	Tập tin .....	4
1.1	Nhu cầu.....	4
1.2	Khái niệm .....	4
1.3	Định vị tập tin.....	4
1.4	Quy trình thao tác với tập tin.....	4
2	Một số hàm thao tác trên tập tin.....	5
2.1	Hàm mở tập tin .....	5
2.2	Đọc và ghi dữ liệu .....	6
2.2.1	Hàm xuất theo định dạng.....	6
2.2.2	Hàm nhập theo định dạng .....	7
2.2.3	Hàm nhập kí tự .....	8
2.2.4	Hàm nhập chuỗi.....	9
2.2.5	Hàm xuất ký tự .....	9
2.2.6	Hàm xuất chuỗi.....	9
2.2.7	Hàm xuất trực tiếp .....	10
2.2.8	Hàm nhập trực tiếp .....	10
2.3	Đóng tập tin .....	11
2.3.1	Hàm đóng tập tin xác định.....	11
2.3.2	Hàm đóng tất cả tập tin.....	11
2.4	Con trỏ chỉ vị (position indicator) .....	12
2.4.1	Khái niệm.....	12
2.4.2	Vị trí con trỏ chỉ vị.....	12
2.4.3	Các hàm cơ bản trên con trỏ chỉ vị .....	12
2.4.4	Hàm đặt lại vị trí con trỏ.....	12
2.4.5	Hàm tái định vị con trỏ .....	13
2.4.6	Hàm xác định vị trí con trỏ .....	13

nlhdung@fit.hcmus.edu.vn

2.5	Dấu hiệu kết thúc tập tin.....	13
3	Bài tập .....	14

## 1 Tập tin

### 1.1 Nhu cầu

Dữ liệu giới hạn và được lưu trữ tạm thời nên mất thời gian, không giải quyết được bài toán với số dữ liệu lớn:

- Nhập: gõ từ bàn phím.
- Xuất: hiển thị trên màn hình.
- Lưu trữ dữ liệu: trong bộ nhớ RAM.

Vì vậy, cần một thiết bị lưu trữ sao cho dữ liệu vẫn còn khi kết thúc chương trình, có thể sử dụng nhiều lần và kích thước không hạn chế.

### 1.2 Khái niệm

- Tập hợp thông tin (dữ liệu) được tổ chức theo một dạng nào đó với một tên xác định.
- Một dãy byte liên tục (ở góc độ lưu trữ).
- Được lưu trữ trong các thiết bị lưu trữ ngoài như đĩa mềm, đĩa cứng, USB...
- Vẫn tồn tại khi chương trình kết thúc.
- Kích thước không hạn chế (tùy vào thiết bị lưu trữ).
- Cho phép đọc dữ liệu (thiết bị nhập) và ghi dữ liệu (thiết bị xuất).

### 1.3 Định vị tập tin

Đường dẫn chỉ đến một tập tin không nằm trong thư mục hiện hành.

Ví dụ: c:\Data\list.txt chỉ tập tin list.txt nằm trong thư mục Data của ổ đĩa C.

Trong chương trình, đường dẫn này được ghi trong chuỗi như sau:  
"C:\\data\\list.txt"

Dấu "\\" biểu thị ký tự điều khiển nên để thể hiện nó ta phải thêm một dấu "\" ở trước. Nhưng nếu chương trình yêu cầu nhập đường dẫn từ bàn phím thì chỉ nhập một dấu "\\".

### 1.4 Quy trình thao tác với tập tin

**Bước 1.** Mở tập tin: tạo một stream (dòng dữ liệu nhị phân/văn bản) nối kết với tập tin cần mở, stream được quản lý bởi biến con trỏ đến cấu trúc FILE:

- Cấu trúc được định sẵn trong STDIO.H
- Các thành phần của cấu trúc này được dùng trong các thao tác xử lý tập tin.

**Bước 2.** Sử dụng tập tin (sau khi đã mở được tập tin)

- Đọc dữ liệu từ tập tin đưa vào chương trình.
- Ghi dữ liệu từ chương trình lên tập tin.

**Bước 3.** Đóng tập tin (sau khi sử dụng xong).

## 2 Một số hàm thao tác trên tập tin

### 2.1 Hàm mở tập tin

Nguyên mẫu hàm	FILE *fopen(const char *filename, const char *mode);	
Ý nghĩa	Mở tập tin có tên (đường dẫn) là filename với kiểu mở mode (bảng bên dưới)	
Giá trị trả về	Thất bại	trả về NULL (sai quy tắc đặt tên tập tin, không tìm thấy ổ đĩa, không tìm thấy thư mục, mở tập tin chưa có để đọc, ...)
	Thành công	trả về biến con trỏ tập tin filename đã được mở theo kiểm mở mode

Bảng các đối số mode:

Đối số	Ý nghĩa
b	Mở tập tin kiểu nhị phân.
t	Mở tập tin kiểu văn bản (text) (kiểu mở mặc định)
r	Mở tập tin chỉ để đọc dữ liệu từ tập tin. Trả về null nếu không tìm thấy tập tin.
w	Mở tập tin chỉ để ghi dữ liệu vào tập tin. Tập tin sẽ được tạo nếu chưa có, ngược lại dữ liệu trước đó sẽ bị xóa hết.
a	Mở tập tin chỉ để thêm (append) dữ liệu vào cuối tập tin. Tập tin sẽ

	được tạo nếu chưa có.
r+	Giống mode r và bổ sung thêm tính năng ghi dữ liệu và tập tin sẽ được tạo nếu chưa có.
w+	Giống mode w và bổ sung thêm tính năng đọc.
a+	Giống mode a và bổ sung thêm tính năng đọc.

Ví dụ:

```
FILE* fp = fopen("taptin.txt", "rt");
if (fp == NULL)
    printf("Khong mo duoc tap tin!");
else
{
    //Xu ly doc du lieu va cac xu ly khac
}
```

## 2.2 Đọc và ghi dữ liệu

Việc đọc/ghi dữ liệu có thể thực hiện theo các cách sau:

- Nhập/xuất theo định dạng
  - Hàm: fscanf, fprintf
  - Chỉ dùng với tập tin kiểu văn bản.
- Nhập/xuất từng ký tự hay dòng lên tập tin
  - Hàm: getc, fgetc, fgets, putc, fput.
  - Chỉ nên dùng với kiểu văn bản.
- Đọc/ghi trực tiếp dữ liệu từ bộ nhớ lên tập tin
  - Hàm: fread, fwrite
  - Chỉ dùng với tập tin kiểu nhị phân.

### 2.2.1 Hàm xuất theo định dạng

Nguyên mẫu hàm	int fprintf(FILE *fp, char *fmt, ...);	
Ý nghĩa	Ghi dữ liệu có chuỗi định dạng fmt (giống hàm printf) vào stream fp. Nếu fp là stdout thì hàm giống printf.	
Giá trị trả về	Thất bại	trả về EOF (có giá trị là -1, được định nghĩa trong STDIO.H, sử dụng trong tập tin có kiểu

		văn bản)
	Thành công	trả về số byte ghi được

Ví dụ:

```
int i = 2912;
int c = 'P';
float f = 17.06;
FILE* fp = fopen("taptin.txt", "wt");
if (fp != NULL)
    fprintf(fp, "%d %c %.2f\n", i, c, f);
```

### 2.2.2 Hàm nhập theo định dạng

Nguyên mẫu hàm	int fscanf(FILE *fp, char *fmt, ...);	
Ý nghĩa	Đọc dữ liệu có chuỗi định dạng fmt (giống hàm scanf) từ stream fp. Nếu fp là stdin thì hàm giống printf.	
Giá trị trả về	Thất bại	trả về EOF.
	Thành công	trả về số thành phần đọc và lưu trữ được.

Ví dụ:

```
int i;
FILE* fp = fopen("taptin.txt", "rt");
if (fp != NULL)
    fscanf(fp, "%d", &i);
```

Ví dụ:

Một tập tin chứa nhiều dòng, mỗi dòng là thông tin mỗi sinh viên theo định dạng sau:

<MSSV>-<Tên>(<Phái>)tab<NTNS>tab<ĐTB>

Ví dụ: 1012078-Nguyen.Thi.Tham(Nu) 17/06/92 8.5

Có thể thực hiện theo một trong 2 cách sau:

Cách 1: Đọc toàn bộ chuỗi và tách từng thành phần theo yêu cầu.

Cách 2: Đọc chuỗi có định dạng theo quy tắc sau:

%[chuỗi]: đọc cho đến khi không gặp ký tự nào trong chuỗi thì dừng.

%[^chuỗi]: đọc cho đến khi gặp một trong những ký tự trong chuỗi thì dừng.

```
struct SINHVIEN {
    char MSSV[8];    // 0312078
    char HoTen[30];  // H. P. Trang
    char GioiTinh[4]; // Nu
    char NTNS[9];    // 17/06/85
    float DiemTB;    // 8.5
};
void main()
{
    SINHVIEN sv;
    FILE *fp = fopen("DanhSachSinhVien.txt", "rt");
    if (fp != NULL)
    {
        fscanf(fp, "%[^-]-%[^](%[^)])\t%[^t]\t%f",
            &sv.MSSV, &sv.HoTen, &sv.GioiTinh,
            &sv.NTNS, &sv.DiemTB);
        fclose(fp);
    }
}
```

### 2.2.3 Hàm nhập kí tự

Nguyên mẫu hàm	int getc(FILE *fp) và int fgetc(FILE *fp);	
Ý nghĩa	Đọc một ký tự từ stream fp. getc là macro còn fgetc là phiên bản hàm của macro getc.	
Giá trị trả về	Thất bại	trả về EOF khi kết thúc stream fp hoặc gặp lỗi.
	Thành công	trả về ký tự đọc được sau khi chuyển sang số nguyên không dấu.

Ví dụ:

```
char ch;
FILE* fp = fopen("taptin.txt", "rt");
if (fp != NULL)
    ch = getc(fp); // <=> ch = fgetc(fp);
```



### 2.2.4 Hàm nhập chuỗi

Nguyên mẫu hàm	int fgets(char *str, int n, FILE *fp);	
Ý nghĩa	Đọc một dãy ký tự từ stream fp vào vùng nhớ str, kết thúc khi đủ n-1 ký tự hoặc gặp ký tự xuống dòng.	
Giá trị trả về	Thất bại	trả về NULL khi gặp lỗi hoặc gặp ký tự EOF.
	Thành công	trả về str.

Ví dụ:

```
char s[20];
FILE* fp = fopen("taptin.txt", "rt");
if (fp != NULL)
    fgets(s, 20, fp);
```

### 2.2.5 Hàm xuất ký tự

Nguyên mẫu hàm	int putc(int ch, FILE *fp); int fputc(in ch, FILE *fp);	
Ý nghĩa	Ghi ký tự ch vào stream fp. putc là macro còn fputc là phiên bản hàm của macro putc.	
Giá trị trả về	Thất bại	trả về EOF.
	Thành công	trả về ký tự ch.

Ví dụ:

```
FILE* fp = fopen("taptin.txt", "rt");
if (fp != NULL)
    putc('a', fp); // hoặc fputc('a', fp);
```

### 2.2.6 Hàm xuất chuỗi

Nguyên mẫu hàm	int fputs(const char *str, FILE *fp);	
Ý nghĩa	Ghi chuỗi ký tự str vào stream fp. Nếu fp là stdout thì fputs giống hàm puts, nhưng puts ghi ký tự xuống dòng.	
Giá trị trả về	Thất bại	trả về EOF.

	Thành công	trả về ký tự cuối cùng đã ghi.
--	------------	--------------------------------

Ví dụ:

```
char s[] = "Ky thuat lap trinh";
FILE* fp = fopen("taptin.txt", "wt");
if (fp != NULL)
    fputs(s, fp);
```

### 2.2.7 Hàm xuất trực tiếp

Nguyên mẫu hàm	int fwrite(void *buf, int size, int count, FILE *fp);	
Ý nghĩa	Ghi count mẫu tin có kích thước mỗi mẫu tin là size (byte) từ vùng nhớ buf vào stream fp (theo kiểu nhị phân)	
Giá trị trả về	Thất bại	trả về số lượng nhỏ hơn count.
	Thành công	trả về số lượng mẫu tin (không phải số lượng byte) đã ghi.

Ví dụ:

```
int a[] = {1, 2, 3};
FILE* fp = fopen("taptin.dat", "wb");
if (fp != NULL)
    fwrite(a, sizeof(int), 3, fp);
```

### 2.2.8 Hàm nhập trực tiếp

Nguyên mẫu hàm	int fread(void *buf, int size, int count, FILE *fp)	
Ý nghĩa	Đọc count mẫu tin có kích thước mỗi mẫu tin là size (byte) vào vùng nhớ buf từ stream fp (theo kiểu nhị phân).	
Giá trị trả về	Thất bại	số lượng nhỏ hơn count khi kết thúc stream fp hoặc gặp lỗi.
	Thành công	trả về số lượng mẫu tin (không phải số lượng byte) thật sự đã đọc.

Ví dụ:

```
int a[5];
FILE* fp = fopen("taptin.dat", "wb");
if (fp != NULL)
    fread(a, sizeof(int), 3, fp);
```

## 2.3 Đóng tập tin

### 2.3.1 Hàm đóng tập tin xác định

Nguyên mẫu hàm	int fclose(FILE *fp);	
Ý nghĩa	Đóng stream fp. Dữ liệu trong stream fp sẽ được "vét" (ghi hết lên đĩa) trước khi đóng.	
Giá trị trả về	Thất bại	trả về EOF.
	Thành công	trả về 0.

Ví dụ:

```
FILE* fp = fopen("taptin.dat", "wb");

// các xử lý.....

fclose(fp);
```

### 2.3.2 Hàm đóng tất cả tập tin

Nguyên mẫu hàm	int fcloseall();	
Ý nghĩa	Đóng tất cả stream đang được mở ngoại trừ các stream chuẩn stdin, stdout, stderr, stderr, stderr. Nên đóng từng stream thay vì đóng tất cả.	
Giá trị trả về	Thất bại	trả về EOF.
	Thành công	trả về số lượng stream được đóng.

Ví dụ:

```
FILE* fp1 = fopen("taptin1.txt", "rt");
FILE* fp2 = fopen("taptin2.txt", "wt");
//...
```

```
fcloseall();
```

## 2.4 Con trỏ chỉ vị (position indicator)

### 2.4.1 Khái niệm

Con trỏ chỉ vị được tạo tự động khi mở tập tin và dùng để xác định nơi diễn ra việc đọc/ghi trong tập tin đó.

### 2.4.2 Vị trí con trỏ chỉ vị

Khi tập tin chưa mở, con trỏ chỉ vị ở đầu tập tin (giá trị 0).

Khi mở tập tin để chèn (mode a hay a+), con trỏ chỉ vị sẽ ở cuối tập tin. Ngược lại con trỏ chỉ vị sẽ ở đầu tập tin khi mở với các mode khác (w, w+, r, r+).

### 2.4.3 Các hàm cơ bản trên con trỏ chỉ vị

Có 2 cách truy xuất nội dung tập tin như sau:

- Truy xuất tuần tự (sequentially access): phải đọc/ghi dữ liệu từ vị trí con trỏ chỉ vị đến vị trí n-1 trước khi đọc dữ liệu tại vị trí n. Do đó, không cần quan tâm đến con trỏ chỉ vị vì con trỏ chỉ vị tự động chuyển sang vị trí kế tiếp sau thao tác đọc/ghi dữ liệu.
- Truy xuất ngẫu nhiên (random access): có thể đọc/ghi tại vị trí bất kỳ trong tập tin mà không cần phải đọc/ghi toàn bộ dữ liệu trước đó. Do đó, cần quan tâm đến con trỏ chỉ vị nếu sử dụng cách truy xuất này.

### 2.4.4 Hàm đặt lại vị trí con trỏ

Nguyên mẫu hàm	<code>void rewind(FILE *fp);</code>
Ý nghĩa	Đặt lại vị trí con trỏ chỉ vị về đầu (byte 0) tập tin fp.
Giá trị trả về	

Ví dụ:

```
FILE* fp = fopen("taptin.txt", "w+");
fprintf(fp, "0123456789");
rewind(fp);
fprintf(fp, "*****");
```

### 2.4.5 Hàm tái định vị con trỏ

Nguyên mẫu hàm	int fseek(FILE *fp, long offset, int origin);	
Ý nghĩa	Đặt vị trí con trỏ chỉ vị trong stream fp với vị trí offset so với cột mốc origin: <input type="checkbox"/> SEEK_SET (hay 0): đầu tập tin. <input type="checkbox"/> SEEK_CUR (hay 1): vị trí hiện tại. <input type="checkbox"/> SEEK_END (hay 2): cuối tập tin.	
Giá trị trả về	Thất bại	trả về giá trị khác 0.
	Thành công	trả về 0.

Ví dụ:

```
FILE* fp = fopen("taptin.txt", "w+");
fseek(fp, 0L, SEEK_SET); // <=> rewind(fp);
fseek(fp, 0L, SEEK_END); // cuối tập tin
fseek(fp, -2L, SEEK_CUR); // lùi lại 2 vị trí
```

### 2.4.6 Hàm xác định vị trí con trỏ

Nguyên mẫu hàm	long ftell(FILE *fp);	
Ý nghĩa	Hàm trả về vị trí hiện tại của con trỏ chỉ vị (tính từ vị trí đầu tiên của tập tin, tức là 0) của stream fp.	
Giá trị trả về	Thất bại	trả về -1L.
	Thành công	trả về vị trí hiện tại của con trỏ chỉ vị.

Ví dụ:

```
FILE* fp = fopen("taptin.txt", "rb");
fseek(fp, 0L, SEEK_END);
long size = ftell(fp);
printf("Kích thước tập tin là %ld\n", size);
```

## 2.5 Dấu hiệu kết thúc tập tin

Cách 1: Xác định kích thước tập tin rồi căn cứ vào đó xác định dấu hiệu kết thúc:

- Khi sử dụng fwrite để lưu n mẫu tin thì kích thước = n \* sizeof(1 mẫu tin);

- Sử dụng hàm fseek kết hợp hàm ftell để lấy kích thước.

Cách 2: Khi chưa biết kích thước tập tin, có thể sử dụng các cách sau:

- Hằng số EOF ( $= -1$ ): chỉ cho tập tin văn bản. Ví dụ: `while ((c = fgetc(fp)) != EOF) ...`
- Hàm `int feof(FILE *fp)` (cho cả 2 kiểu tập tin) trả về số 0 nếu chưa đến cuối tập tin, ngược lại trả về số khác 0.

### 3 Bài tập

**Bài 1:** Viết chương trình ghi 3 số nguyên a, b, c được nhập từ bàn phím vào một tập tin.

**Bài 2:** Viết chương trình đọc 3 số nguyên a, b, c từ một tập tin, sau đó giải phương trình  $ax^2 + bx + c = 0$  rồi ghi kết quả vào một tập tin khác.

**Bài 3:** Viết chương trình đọc n số nguyên từ một tập tin cho trước, sau đó sắp xếp tăng dần rồi ghi kết quả vào 1 tập tin khác.

**Bài 4:** Viết chương trình ghi các dòng văn bản được nhập từ bàn phím lên tập tin.

**Bài 5:** Viết chương trình in nội dung của một tập tin lên màn hình.

**Bài 6:** Viết chương trình đếm số ký tự chữ cái trong tập tin và xuất kết quả ra một tập tin khác.

**Bài 7:** Viết chương trình đếm số từ trong tập tin và xuất kết quả ra một tập tin khác.

**Bài 8:** Viết chương trình mở tập tin văn bản đã có trên đĩa, sao chép nó thành một tập tin văn bản mới với điều kiện là các chữ thường đổi thành chữ hoa, tất cả các ký tự khác không đổi.

**Bài 9:** Viết chương trình ghép 2 tập tin văn bản, nội dung tập tin thứ hai được ghép sau tập tin thứ nhất.

**Bài 10:** Viết chương trình ghi một danh sách cấu trúc xuống tập tin sau đó đọc lên và xuất ra để kiểm tra lại.