



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

TIN HỌC ĐẠI CƯƠNG

Phần II: LẬP TRÌNH C

Nội dung chính

- Chương 1: Tổng quan về ngôn ngữ C
- Chương 2: Kiểu dữ liệu và biểu thức trong C
- Chương 3: Vào ra dữ liệu
- Chương 4: Cấu trúc điều khiển
- Chương 5: Mảng, con trỏ và chuỗi ký tự
- Chương 6: Cấu trúc
- Chương 7: Hàm
- **Chương 8: Tập dữ liệu**

Chương 8: Tập dữ liệu

8.1. Tập và phân loại tập

- Khái niệm và phân loại
- Tập và mảng
- Tổ chức tập

8.2. Các thao tác với tập

- Khai báo
- Mở tập
- Đóng tập
- Truy nhập tập văn bản
- Truy nhập tập nhị phân

Chương 8: Tập dữ liệu

8.1. Tập và phân loại tập

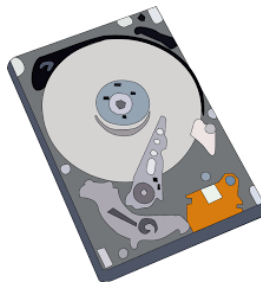
- Khái niệm và phân loại
- Tập và mảng
- Tổ chức tập

8.2. Các thao tác với tập

- Khai báo
- Mở tập
- Đóng tập
- Truy nhập tập văn bản
- Truy nhập tập nhị phân

Khái niệm tệp

- Tệp dữ liệu (File) là một tập hợp các dữ liệu có liên quan với nhau và có cùng kiểu dữ liệu.
- Tệp được lưu trữ trên các **thiết bị nhớ ngoài** (đĩa cứng, USB, thẻ nhớ ...) với một tên nào đó để phân biệt với nhau.
- Vai trò của tệp: Tệp là phương tiện để cất giữ **dữ liệu lâu dài**, dữ liệu lưu trữ trong tệp **không bị mất đi** khi chương trình kết thúc hay khi tắt máy



Khái niệm

- Tập (Tập tin/File):
 - Tập hợp các dữ liệu cùng kiểu
 - Có liên quan tới nhau
- Lưu trữ tập
 - Lưu trữ trên thiết bị lưu trữ ngoài
 - Có tên riêng để phân biệt
- Phân thành 2 loại
 - Tập văn bản (text file)
 - Tập nhị phân (binary file)

Kiểu mảng?

Khái niệm Tập/Tập tin/File

- **Tập** là đơn vị thông tin mà hệ điều hành quản lý trên bộ nhớ ngoài
 - Thường là tập hợp các dữ liệu có liên quan tới nhau, được tổ chức theo một cấu trúc nào đó.
 - Nội dung có thể là chương trình, dữ liệu, văn bản,...
- Mỗi tập được lưu trên đĩa với một tên phân biệt
 - Tên tập được đặt theo quy ước của hệ điều hành
 - Tên tập tin thường có 2 phần:
 - Phần tên (name): Buộc phải có
 - Phần mở rộng (extension)
 - Giữa phần tên và phần mở rộng có một dấu chấm (.) ngăn cách.

Phần tên tệp

- Bao gồm các ký tự chữ từ A đến Z (thường và hoa) chữ số từ 0 đến 9,
- Ký tự khác như #, \$, %, ~, ^, @, (,), !, _, khoảng trắng
- Độ dài tên tệp phụ thuộc hệ điều hành. Với Windows 10 là 32767 ký tự.
- **Lưu ý:** *Nên đặt tên mang tính gợi nhớ.*
 - Hợp lệ: dulieu211212.txt, dulieu\$211212.dat
 - Không hợp lệ: 'dulieu211212.txt, ?abc.dat

Phần mở rộng

- Thường là 3 ký tự hợp lệ, do chương trình ứng dụng tạo ra tập tin tự đặt theo quy ước.
- Một số phần mở rộng trong Windows
 - COM, EXE : Các file khả thi chạy trực tiếp
 - TXT, DOC, DOCX ... : Các file văn bản.
 - PAS, C, CPP, ... : Các file chương trình PASCAL, C, C++ ...
 - XLS, ... : Các file chương trình bảng tính EXCEL ...
 - BMP, GIF, JPG, ... : Các file hình ảnh.
 - MP3, DAT, WMA, ... : Các file âm thanh, video.

Phân loại

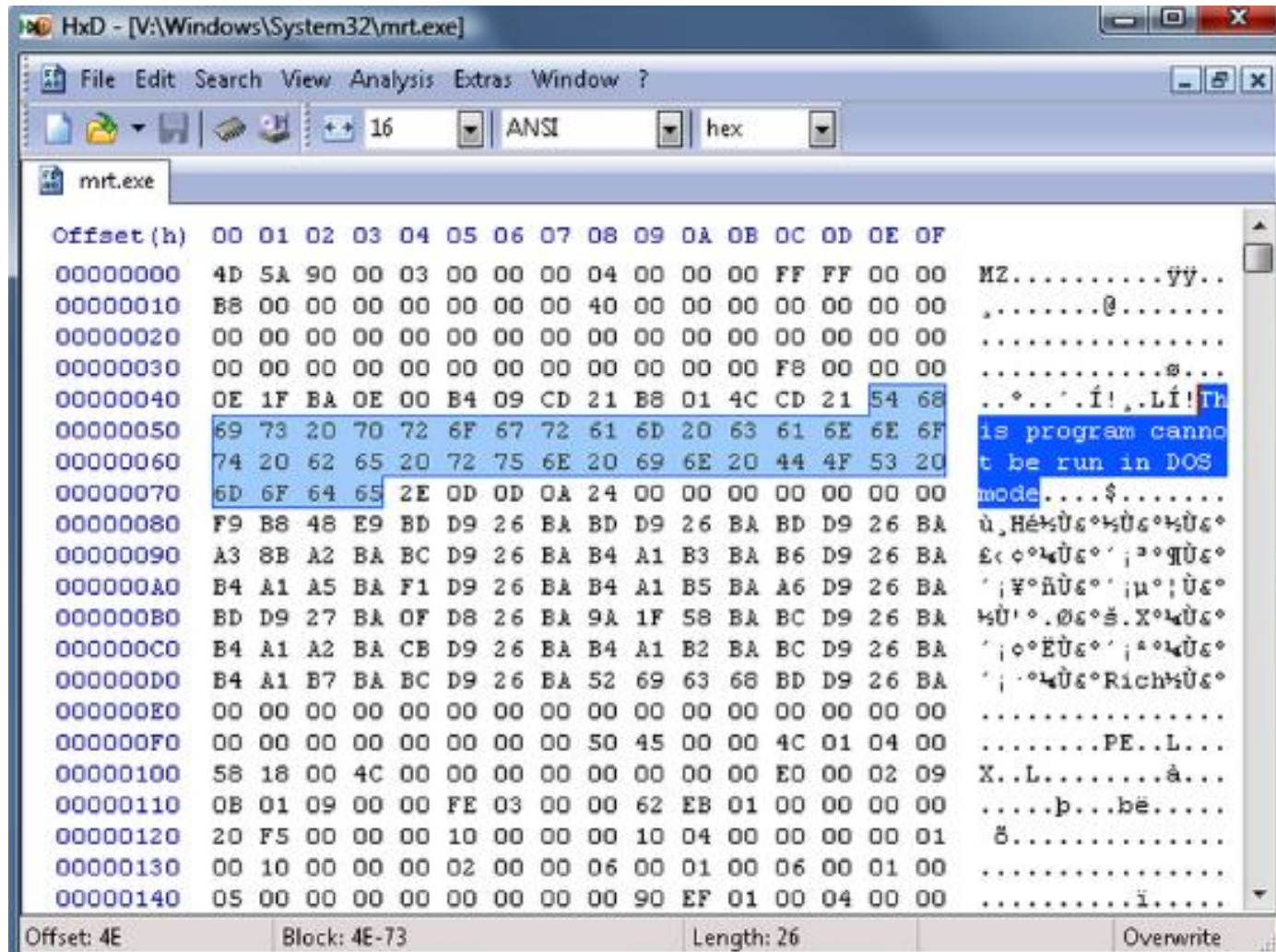
- Tập văn bản
 - Được tổ chức theo từng dòng
 - Trên mỗi dòng là các ký tự ASCII hiển thị được như chữ cái, chữ số, dấu câu,...
 - Cuối mỗi dòng là các ký tự điều khiển
 - CR: Carriage Return - mã ASCII 13
 - LF: Line Feed- Mã ASCII 10
- Tập nhị phân
 - Các phần tử của tập là các số nhị phân dùng mã hóa thông tin
 - Thông tin được mã hóa: số, cấu trúc dữ liệu, ...

ASCII Chart

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2	SPC	!	"	#	\$	%	&	'	()	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{		}	~	DEL

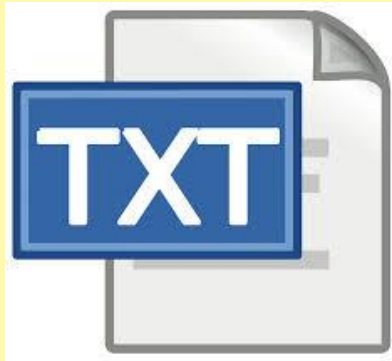
Physical Device Controls: Format Effectors

File nhị phân (Binary file)



Phân loại Tập (File)

File văn bản



ghi_chú.txt

```
1  
#include <stdio.h>  
int main() {  
    FILE *fptr;  
    fptr = fopen("fprintf_test.txt", "w");  
    // Write to file  
    fprintf(fptr, "Learning C with Guru99\n");  
    close(fptr);  
    return 0;  
}
```

2

doc_file.c

File nhị phân



bai_hat.mp3



bao_cao.docx



bang_diem.xls

Tập dữ liệu và mảng

- Mảng

- Được lưu trong bộ nhớ → dữ liệu bị mất đi khi tắt máy
- Truy nhập trực tiếp tới một phần tử qua chỉ số
- Kích thước mảng xác định trước

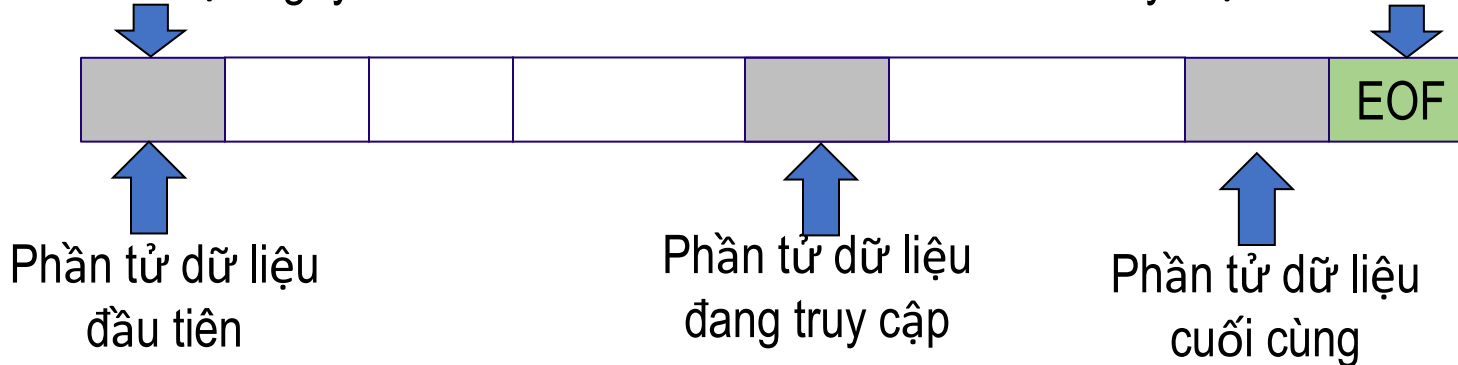
- Tập

- Lưu trữ trên thiết bị lưu trữ ngoài → dữ liệu được lưu trữ lâu dài, không bị mất đi khi tắt máy
- Không truy nhập trực tiếp qua số hiệu phần tử
- Kích thước có thể rất lớn và không cần xác định trước

Tổ chức tệp

Phần tử dữ liệu ngay sau khi mở file

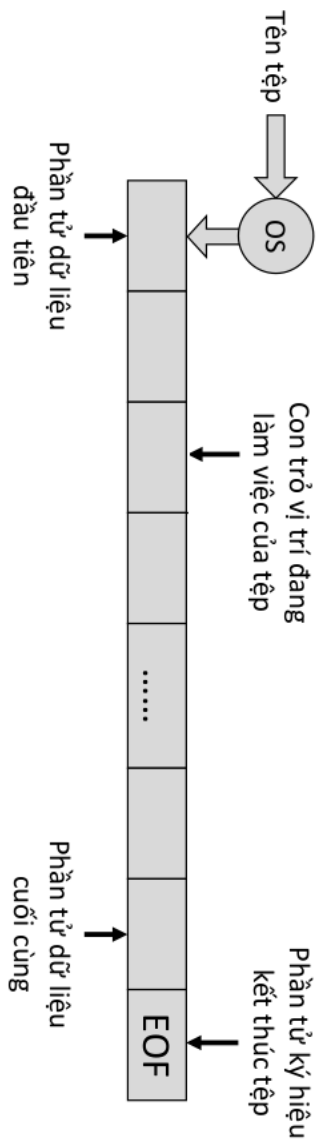
Ký hiệu đánh dấu kết thúc file



- Tệp là dãy các phần tử kế tiếp nhau
 - Sử dụng phần tử đặc biệt (EOF) để đánh dấu kết thúc tệp
- Con trỏ tệp:
 - Xác định vị trí phần tử hiện có thể truy cập
 - Khi mở file, con trỏ tệp luôn ở vị trí phần tử đầu
 - Sau các thao tác đọc/ghi tệp, con trỏ file dịch chuyển về cuối tệp một khoảng bằng số byte đã đọc/ ghi

Con trỏ tệp

- Khi **mở tệp**, con trỏ tệp sẽ luôn trở đến **phần tử đầu tiên** của tệp.
- Sau mỗi thao tác **đọc/ghi** trên tệp, con trỏ tệp sẽ **tự động dịch chuyển về phía cuối tệp**.
- Khoảng cách dịch chuyển (tính theo byte) sẽ bằng số byte đã được đọc từ tệp hoặc ghi lên tệp.



Chương 8: Tập dữ liệu

8.1. Tập và phân loại tập

- Khái niệm và phân loại
- Vai trò của tập
- Tổ chức tập

8.2. Các thao tác với tập

- Khai báo
- Mở tập
- Đóng tập
- Truy nhập tập văn bản
- Truy nhập tập nhị phân

Quy trình

- Bước 1. Khai báo biến tệp
- Bước 2. Mở tệp để làm việc
 - Phân biệt các loại tệp và các mục đích mở tệp
- Bước 3. Truy nhập tệp
 - Truy nhập để đọc/ ghi/thêm mới
 - Phân biệt giữa các loại tệp
- Bước 4. Đóng tệp

Bước 1. Khai báo biến tệp

```
FILE *Con_Trở_Tệp;
```

- Tệp được truy nhập qua con_trở_tệp
- Ví dụ

```
FILE *f1, *f2;
```

Bước 2. Mở tệp

`Con_Trở_Tập = fopen(Tên_Tập, Chế_độ_mở)`

- Hàm `fopen()` khi báo trong thư viện `stdio.h`
- `Tên_Tập`: Kiểu chuỗi, xác định tên tệp trên đĩa
 - Tên đầy đủ của tệp hoặc tệp trên thư mục hiện thời
- `Chế_độ_mở`: Hằng xâu, gồm các ký tự `r/w/a/+t/b`
 - Tùy thuộc kiểu tệp và mục đích sử dụng
 - Kiểu tệp: `t`: text file (ngầm định); `b`: binary file
 - Chế độ: `+` Vừa đọc/ vừa ghi
 - `r` : Mở để đọc; Báo lỗi nếu tệp chưa tồn tại
 - `w` : Mở mới để ghi; Xóa nội dung tệp cũ nếu đã có
 - `a` : Mở để ghi vào cuối; Tạo tệp mới nếu chưa tồn tại
- Trả về `NULL` nếu có lỗi mở tệp

Ví dụ

- Mở tệp văn bản để đọc: `f1 = fopen("c:\\abc.txt", "r");`
- Mở tệp văn bản để vừa đọc và ghi: `f3 = fopen("c:\\abc.txt", "r+");`
- Mở tệp nhị phân để ghi: `f2 = fopen("c:\\ho_so.dat", "wb");`

Kiểm tra lỗi phát sinh khi mở tệp

// Trường hợp mở tệp có lỗi

File *con_trở_tệp;

con_trở_tệp = fopen(tên_tệp, chế_độ_mở_tệp);

if (con_trở_tệp == NULL) {

 // Xử lý cho trường hợp mở tệp không thành công

} else {

 // Xử lý khi mở tệp thành công

}

Đóng tệp

```
int fclose(FILE *Con_Trở_Tệp)
```

- Hàm fclose() khi báo trong thư viện stdio.h
- Con_trở_tệp: Tên biến tệp
- Kết quả trả về
 - 0: Nếu đóng tệp thành công
 - EOF: Nếu có lỗi

Truy nhập tệp văn bản

- Tương tự như với bàn phím/ màn hình
- Yêu cầu chỉ rõ nguồn/đích thông tin
- Các thao tác
 - Đọc dữ liệu từ tệp : `fscanf()` / `fgets()` / `getc()`
 - Ghi dữ liệu ra tệp : `fprintf()` / `fputs()` / `putc()`
 - Dịch chuyển con trỏ tệp : `fseek()` / `rewind()`
 - Kiểm tra kết thúc tệp : `feof()`
- Ví dụ:
 - `fprintf(FILE *fptr, Xâu_định_dạng [,DS giá trị])`
 - `fgets(char * Xâu_ký_tự, int n, FILE *fptr)`

Thao tác với tệp văn bản

- Đọc dữ liệu từ tệp văn bản: `fscanf()`, `fgets()`, `getc()`
- Hàm `fscanf()`:
 - Cú pháp: `int fscanf(FILE* con_trỏ_tệp, chuỗi_định_dạng, [danh_sách_địa_chỉ]);`
 - Đọc từ tệp văn bản tương ứng với con trỏ tệp
 - dãy các dữ liệu
 - định dạng các dữ liệu đó theo khuôn dạng có trong chuỗi_định_dạng
 - sau đó lưu các giá trị đã được định dạng vào các vùng nhớ xác định trong danh_sách_địa_chỉ
 - Kết quả trả về: là số nguyên mô tả số byte đọc được từ tệp nếu thành công, nếu không thành công thì trả về giá trị EOF
 - Ví dụ: `fscanf(fp, "%d %c", &a, &c);`

Char* fgets(char* xâu_kí_tự, int n, FILE* con_trở_tập);

- Đọc từ tệp một xâu kí tự (cho phép chứa dấu cách)
- Gán xâu đọc được cho biến xâu_kí_tự.
- Việc đọc tệp sẽ dừng lại khi fgets() đọc đủ n-1 kí tự hoặc khi gặp kí tự xuống dòng.
- Tự động thêm kí tự xuống dòng (“\n”) và kí tự kết thúc xâu (“\0”, kí tự NULL) vào cuối xâu_kí_tự.
- Giá trị trả về:
 - Nếu đọc thành công: trả về xâu kí tự trở bởi biến xâu_kí_tự
 - Nếu có lỗi: trả về con trỏ NULL

int getc(FILE* con_trỏ_tệp);

- Đọc từ tệp một kí tự (tức là một byte dữ liệu), sau đó chuyển đổi kí tự đó sang dạng số nguyên int (bằng cách thêm byte cao 0x00) rồi lấy giá trị số nguyên thu được làm giá trị trả về của hàm.
- Giá trị trả về:
 - Thành công: trả về kí tự đọc được sau khi đã chuyển sang dạng int.
 - Nếu không thành công: trả về EOF

Ghi dữ liệu lên tệp

- Để ghi dữ liệu lên tệp : fprintf(), fputs(), putc()
- int fprintf(FILE* con_trỏ_tệp, xâu_định_dạng, [danh_sách_tham_số]);
- Nếu thành công, hàm fprintf() trả về một giá trị số nguyên là số byte dữ liệu đã ghi lên tệp.
- Nếu không thành công, hàm fprintf() trả về giá trị EOF.

int fputs(char* xâu_kí_tự, FILE* con_trỏ_tệp);

- Ghi nội dung của xâu_kí_tự lên tệp tương ứng với con_trỏ_tệp
- Không tự động ghi thêm kí tự xuống dòng lên tệp.
- Giá trị trả về:
 - Nếu thành công: trả về kí tự cuối cùng mà đã ghi được lên tệp.
 - Nếu không thành công: trả về EOF.

int putc(int ch, FILE* con_trỏ_tệp);

- Ghi nội dung của kí tự chứa trong biến int ch (kí tự được chứa trong byte thấp của biến ch) lên tệp tương ứng với con_trỏ_tệp.
- Giá trị trả về:
 - Nếu thành công: trả về số nguyên (kiểu int) là số thứ tự trong bảng mã ASCII của kí tự đã ghi lên tệp.
 - Nếu không thành công: trả về giá trị EOF.
- Ví dụ:
- int m;
- m = putc(0x2345, stdout);
- // m = 0x2345 = 69, là số thứ tự của kí tự E.

Ghi dữ liệu lên tệp

- Để ghi dữ liệu lên tệp : **fprintf()**, **fputs()**, **putc()**
- Ví dụ:

```
fprintf(fptr, "%s %s %s %d", "We", "are", "in", 2012);
```

```
fputs("This is c programming.", fptr);
```

```
putc(ch, fptr);
```

int feof(FILE* con_trò_tệp);

- Hàm feof() dùng để kiểm tra xem đã duyệt đến vị trí cuối tệp hay chưa.
- Hàm thực hiện việc này bằng cách kiểm tra xem trong khối dữ liệu được đọc vào ở lần thực hiện gần nhất có phần tử EOF hay không (tức là phần tử EOF có được đọc trong lần đọc gần đây nhất hay không),
- Nếu có thì hàm feof() trả về một giá trị khác 0
- Nếu chưa thì trả về giá trị 0.

int feof(FILE* con_trỏ_tệp);

Con trỏ vị trí đang
làm việc của tệp



Các phần tử của tệp

EOF

Chưa đọc phần tử EOF
feof() = 0 => FALSE

Con trỏ vị trí đang
làm việc của tệp



Các phần tử của tệp

EOF

Đã đọc phần tử EOF
feof() != 0 => TRUE

- Hàm feof() dùng để kiểm tra xem đã duyệt đến vị trí cuối tệp hay chưa.

```
while (!feof(fptr))  
{  
    // Đọc dữ liệu từ file  
  
    // Xử lý dữ liệu đọc từ file  
}
```

Hàm fflush():

int fflush(FILE* con_trỏ_tệp);

- Ghi toàn bộ dữ liệu chứa trong vùng đệm của tệp (nằm ở bộ nhớ trong) tương ứng với con trỏ tệp ra vùng nhớ của tệp trên bộ nhớ ngoài.
- Giá trị trả về:
 - thực hiện thành công: trả về 0
 - không thành công: trả về EOF
- Ví dụ: **fflush(fp);**

Ví dụ thao tác với tệp văn bản

- Nhập thông tin điểm thi toán, lý, hóa từ bàn phím
- Ghi các điểm này vào tệp văn bản có tên “my_score.txt”.
- Đọc từ tệp trên, tính điểm trung bình và in kết quả ra màn hình.

Mã nguồn: ghi dữ liệu vào file văn bản

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

```
void main()
```

```
{
```

```
    FILE *f;
```

```
    float dToan, dLy, dHoa;
```

```
    // Nhập điểm từ bàn phím
```

```
    printf("Nhập điểm thi Toán, Lý Hóa: ");
```

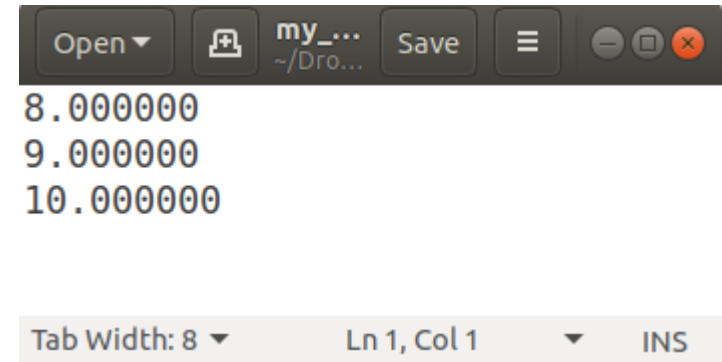
```
    scanf("%f%f%f", &dToan, &dLy, &dHoa);
```

```
    // Lưu vào file văn bản
```

```
    f = fopen("my_score.txt", "w");
```

```
    fprintf(f, "%f\n%f\n%f\n", dToan, dLy, dHoa);
```

```
    fclose(f);
```



Mã nguồn: đọc dữ liệu từ file văn bản

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

```
void main()
```

```
{
```

```
    FILE *f;
```

```
    float dToan, dLy, dHoa, dTb;
```

```
    // Doc du lieu tu file van ban
```

```
    f = fopen("my_score.txt", "r");
```

```
    fscanf(f, "%f%f%f", &dToan, &dLy, &dHoa);
```

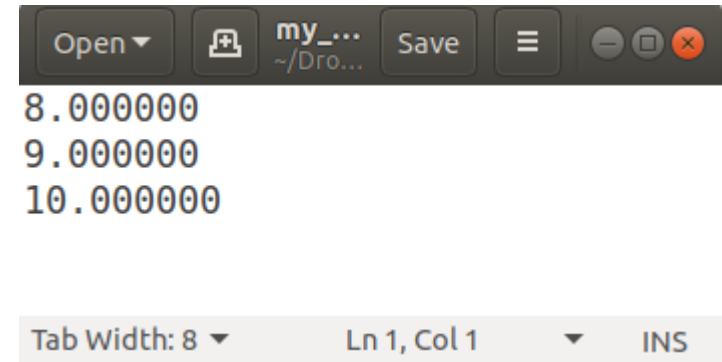
```
    fclose(f);
```

```
    // Tinh diem trung binh
```

```
    dTb = (dToan + dLy + dHoa) / 3;
```

```
    printf("Diem trung binh: %f", dTb);
```

```
}
```



```
Open ▾ my_... ~/.Dro... Save ≡ - □ ×
```

```
8.000000  
9.000000  
10.000000
```

```
Tab Width: 8 ▾ Ln 1, Col 1 ▾ INS
```

```
Diem trung binh: 9.000000  
-----  
(program exited with code: 0)  
Press return to continue
```

File nhị phân

Truy nhập tệp nhị phân

Đọc dữ liệu

```
int fread(void *Địa_Chỉ_Đích, int Kích_thước,  
          int số_phần_tử, FILE *fptr)
```

- Đọc từ file xác định bởi biến: fptr
- một khối dữ liệu kích thước: Số_Phần_Tử x Kích_Thước
- vào vùng nhớ xác định bởi: Địa_Chỉ_Đích
- Nếu đọc thành công: Trả về số phần tử đọc được
- Nếu không thành công: Trả về giá trị 0

- Ví dụ:

```
int Buf[100];  
FILE *fptr = fopen("so.dat","rb");  
fread( Buf, sizeof(int), 100, fptr);
```

Truy nhập tệp nhị phân

Ghi dữ liệu

```
int fwrite(void *Địa_Chỉ_Nguồn, int Kích_thước,  
           int số_phần_tử, FILE *fptr)
```

- Ghi từ vùng nhớ xác định bởi Địa_Chỉ_nguồn
 - một khối dữ liệu có kích thước Số_Phần_Tử x Kích_Thước
 - ra file được xác định bởi biến fptr
 - Nếu ghi thành công: Trả về số phần tử đã ghi
 - Nếu không thành công: Trả về giá trị 0
- Ví dụ:

```
int Buf[100];
```

```
FILE * fptr = fopen("so.dat","wb")
```

```
fwrite(Buf, sizeof(int), 100, fptr);
```


Truy nhập tệp nhị phân

Dịch chuyển con trỏ file

`int fseek(FILE *fptr, long int N, int Vị_Trí_Đầu)`

– Dịch chuyển con trỏ file của file `fptr` đi một khoảng `N` so với `Vị_Trí_Đầu`

- `SEEK_SET / 0`: Vị trí đầu là đầu tệp
- `SEEK_CUR / 1`: Vị trí đầu là vị trí con trỏ file hiện thời
- `SEEK_END / 2`: Vị trí đầu là cuối tệp

– `void rewind(FILE *fptr)`: Đưa con trỏ về đầu tệp

Kiểm tra kết thúc file

`int feof(FILE *fptr)`

– Trả về 0 nếu con trỏ file vẫn còn trỏ tới một phần tử dữ liệu, 1 nếu con trỏ file đang trỏ tới EOF

Ví dụ 1. File nhị phân: ghi mảng số nguyên vào file

```
#include <stdio.h>
int main() {
    FILE *fp;
    int c[3] = {3, 4, 5};
    int i;
    fp = fopen("file_nhi_phan_2.dat", "wb");
    fwrite(c, 3, sizeof(int), fp);
    fclose(fp);
}
```

Name:	file_nhi_phan_2.dat
Type:	Binary (application/octet-stream)
Size:	12 bytes

Ví dụ 2. File nhị phân: đọc mảng số nguyên từ file

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

```
int main() {
```

```
    FILE *fp;
```

```
    int buffer[3];
```

```
    fp = fopen("file_nhi_phan_2.dat", "rb");
```

```
    fread(buffer, 3, sizeof(int), fp);
```

```
    for (i=0; i<3; i++) {
```

```
        printf("\nPhan tu thu %d : %d", i, buffer[i]);
```

```
    }
```

```
    fclose(fp);
```

```
Phan tu thu 0 : 3
```

```
Phan tu thu 1 : 4
```

```
Phan tu thu 2 : 5
```

```
-----
```

```
(program exited with code: 0)
```

```
Press return to continue
```

Sử dụng vòng lặp và lệnh kiểm tra feof

```
FILE *fp;
```

```
int n;
```

```
fp = fopen("file_nhi_phan_2.dat", "rb");
```

```
if (fp == NULL)
```

```
    printf("ERROR : khong the mo file");
```

```
else {
```

```
    // Xử lý khi mở
```

```
    // được file
```

```
}
```

```
fclose(fp);
```

```
while (!feof(fp))
```

```
{
```

```
    fread(&n, 1, sizeof(int), fp);
```

```
    printf("n = %d", n);
```

```
}
```

Ví dụ 3

Tạo file Songuyen.dat ghi 100 số lẻ đầu tiên.

```
#include <stdio.h>
int main() {
    FILE * f = fopen("SoNguyen.Dat", "wb");
    int i, n;
    for(i = 0; i <100; i++) {
        n = 2*i+1;
        fwrite(&n, sizeof(int), 1, f);
    }
    fclose(f);
    return 0;
}
```

Ví dụ 4

Đọc file Songuyen.dat, đưa ra màn hình các số lẻ từ vị trí số thứ 50 của file

```
#include <stdio.h>
int main() {
    FILE * f = fopen("SoNguyen.Dat", "rb");
    int n;
    fseek(f, 50*sizeof(int), SEEK_SET);
    while(!feof(f)){
        fread(&n, sizeof(int), 1, f);
        printf("%4d", n);
    }
    fclose(f);
    return 0;
}
```

Ví dụ 5

- Nhập danh sách từ bàn phím các thí sinh dự thi, mỗi thí sinh gồm họ tên, số báo danh, khoa dự thi và điểm thi.
- Dữ liệu nhập được ghi vào file ThiSinh.dat. Kết thúc nhập khi gặp một thí sinh có tên là « *** »
- Đọc từ file ThiSinh.Dat, đưa ra màn hình danh sách các thí sinh thi vào ngành CNTT có điểm thi lớn hơn 21 theo quy cách
STT Số Báo Danh Họ Tên Điểm Thi
- Từ file ThiSinh.Dat, tạo file CNTT.Dat chỉ chứa danh sách các thí sinh thi vào khoa CNTT
- Nhập vào một số báo danh, tìm trong file ThiSinh.Dat và in ra họ tên, điểm thi và khoa đăng ký của thí sinh nếu tìm thấy. Nếu không tìm thấy thí sinh thì đưa ra thông báo « không tìm thấy »

Ví dụ 5

```
#include <stdio.h>
#include <string.h>
typedef struct {
    char Ten[30];
    long SBD;
    char Khoa[10];
    float Diem;
} SinhVien;

int main() {
    FILE *f1,*f2;
    SinhVien SV;
    int i, SBD;
    //Nhap thông tin cho file ThiSinh.Dat
    //Tạo file CNTT.Dat
    return 0;
}
```


Ví dụ 5

```
//Nhap thong tin cho file ThiSinh.Dat
f1 = fopen("ThiSinh.Dat", "wb");
i = 1;
do {
    printf("Thi sinh %d :\n", i);
    printf("  Ho Ten : "); fflush(stdin); fgets(SV.Ten, sizeof(SV.Ten), stdin);
    SV.Ten[strlen(SV.Ten) - 1] = '\0'; // Bỏ ký tự xuống dòng
    if(strcmp(SV.Ten, "***") == 0) break;
    printf("  So Bao Danh: "); scanf("%d", &SV.SBD);
    printf("  Khoa : "); fflush(stdin); fgets(SV.Khoa, sizeof(SV.Khoa), stdin);
    SV.Khoa[strlen(SV.Khoa) - 1] = '\0'; // Bỏ ký tự xuống dòng
    printf("  Diem : "); scanf("%f", &SV.Diem);
    fwrite(&SV, sizeof(SinhVien), 1, f1);
    i++;
} while(1);
fclose(f1);
```

Ví dụ 5

```
printf("\n\n DANH SACH BAN DAU \n");
f1 = fopen("ThiSinh.Dat", "rb");
i = 0;
while (fread(&SV, sizeof(SinhVien), 1, f1) > 0)
    printf("%-3d %-5d %-20s %-20s %-5.1f\n", ++i, SV.SBD,
           SV.Ten, SV.Khoa, SV.Diem);

printf("\n\n Thi Sinh thi CNTT tren 21.0\n");
i = 0;
rewind(f1);

while (fread(&SV, sizeof(SinhVien), 1, f1) > 0)
    if (strcmp(SV.Khoa, "CNTT") == 0 && SV.Diem > 21.0)
        printf("%-3d %-5d %-20s %-5.1f\n", ++i, SV.SBD,
               SV.Ten, SV.Diem);
```

Ví dụ 5

```
printf("\n\n Tao file CNTT.Dat\n");
i = 0;
rewind(f1);
f2 = fopen("CNTT.Dat", "wb");
while (fread(&SV, sizeof(SinhVien), 1, f1) > 0)
    if (strcmp(SV.Khoa, "CNTT") == 0 )
        fwrite(&SV, sizeof(SinhVien), 1, f2);
fclose(f2);

f2 = fopen("CNTT.Dat", "rb");    //doc lai file
while (fread(&SV, sizeof(SinhVien), 1, f2) > 0)
    printf("%-3d %-5d %-20s  %-5.1f\n", ++i, SV.SBD, SV.Ten,
        SV.Diem);
fclose(f2);
```

Ví dụ 5

```
printf("\n\nTim Sinh Vien\n");
printf(" So Bao Danh "); scanf("%d", &SBD);
rewind(f1);
while (fread(&SV, sizeof(SinhVien), 1, f1))
    if (SV.SBD == SBD){
        printf("Tim thay sinh vien %s", SV.Ten);
        break;
    }

if (feof(f1)) printf("Khong thay");
fclose(f1);
```

Bài tập 1. Thao tác với File văn bản

- Mở một tệp văn bản, đếm xem trong văn bản đó có bao nhiêu kí tự.

Bài tập 2. Thao tác với tệp văn bản

- Nhập thông tin điểm thi giữa kỳ và cuối kỳ của 2 sinh viên từ bàn phím, biết điểm thi có giá trị từ 0 đến 10.
- Ghi các điểm này vào tệp văn bản có tên “my_score.txt”.
- Đọc từ tệp trên, tính điểm theo thang chữ của từng sinh viên (biết điểm thi giữa kỳ hệ số 0.3 và điểm thi cuối kỳ hệ số 0.7) và in kết quả ra màn hình.

Bài tập 3. Thao tác với file nhị phân

- Nhập vào từ bàn phím 5 số thực và ghi vào file nhị phân POSITIVE_NUMBERS.DAT các số thực > 0 trong các số đã nhập
- Đọc từ file POSITIVE_NUMBERS.DAT và đưa ra màn hình số thực có giá trị lớn nhất

Bài tập 4. Thao tác với file nhị phân

- Viết chương trình mã hóa một file văn bản theo nguyên tắc sau: mỗi ký tự từ 'A' đến 'Z' sẽ được thay thế bởi ký tự sau đó 3 vị trí (ví dụ 'A' => 'D', 'B' => 'E', ...), nếu quá ký tự 'Z' thì sẽ quay vòng về ký tự 'A', 'B', 'C',... (ví dụ: 'X' => 'A', 'Y' => 'B', 'Z' => 'C')
- Chương trình sẽ nhập vào tên file văn bản đầu vào và tên file văn bản cần mã hóa.
- Viết chương trình giải mã: nhập vào tên file văn bản mã hóa và tên file văn bản được giải mã.

Bài tập 5. Thao tác với file nhị phân

- Cho một mảng gồm 4 số thực:

`float arr[4] = {6.5, 7.5, 8.5, 9.5};`

- A. Hãy ghi mảng này vào file `my_file_bt2.dat`
- B. Hãy đọc mảng này từ file `my_file_bt2.dat`. Hãy tính và in ra màn hình trung bình cộng các giá trị vừa nhập.

Bài tập 6

- Nhập danh sách từ bàn phím danh sách hàng hóa, mỗi hàng hóa gồm tên hàng, số lượng, đơn giá.
- Dữ liệu nhập được ghi vào file HangHoa.dat. Kết thúc nhập khi gặp một hàng hóa có tên là « *** »
- Đọc từ file HangHoa.Dat, đưa ra màn hình danh sách các hàng hóa theo quy cách

STT	Tên Hàng	Số Lượng	Đơn Giá	Thành Tiền
-----	----------	----------	---------	------------

- Nhập vào một số lượng, tìm trong file HangHoa.Dat và in ra thông tin của những hàng hóa có số lượng như vậy.. Nếu không tìm thấy thì đưa ra thông báo « không tìm thấy »

Bài tập trắc nghiệm 1

Phát biểu nào về tệp dưới đây là đúng?

- a. Việc truy cập cùng lúc vào nhiều phần tử của tệp là có thể nếu chế độ mở tệp cho phép
- b. Tệp văn bản là trường hợp đặc biệt của tệp nhị phân
- c. Sau mỗi thao tác đọc, ghi trên tệp, con trỏ tệp sẽ tự động trở về vị trí đầu tiên của tệp.
- d. EOF là một hằng số nguyên được định nghĩa trong thư viện `conio.h`

Bài tập trắc nghiệm 2

Khi dùng hàm fopen thì kiểu nào sau đây làm đối số của hàm này, dùng để mở một tệp để ghi bổ sung theo kiểu văn bản, nếu tệp chưa tồn tại thì tạo tệp mới?

- a. “at”
- b. “wt”
- c. “w+t”
- d. “ab”

Bài tập trắc nghiệm 3

Điền vào chỗ trống để hoàn thiện chương trình sau:

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

```
void main(){
```

```
    .....(1).....
```

```
    f = fopen("solieu.txt","wt");
```

```
    fprintf(f, "%2d\n%2d",11,09);
```

```
    .....(2).....
```

```
}
```

- a. (1) điền FILE *f; (2) điền fclose(f);
- b. (1) điền FILE f; (2) điền fclose(f);
- c. (1) điền FILE f; (2) điền close(f);
- d. (1) điền FILE *f; (2) điền FCLOSE(f);

