Tin học cơ sở 4

Files

File

- Hầu hết các chương trình đều cần nhập/xuất dữ liệu.
- User nhập data theo dạng data tự nhiên của con người.
- Chương trình xuất dữ liệu ra theo dạng của con người.
- Có sự khác biệt về mô tả thông tin giữa người và máy. Người: 12 , máy 1010
- Tập tin là nơi chứa dữ liệu nên cũng có thể là nguồn dữ liệu cho chương trình.
- User mở được một tập tin để ghi dữ liệu vào thì chương trình cũng có thể mở một tập tin để ghi các dữ liệu của chương trình vào.

File

- Dữ liệu nhập → Chương trình → Dữ liệu xuất

  Bàn phím

  Đĩa (file)

  Modem

  .....

  Màn hình

  Đĩa (file)

  Modem

  Máy in

  ....

File

- ***File*** = A complete, named collection of information, such as a program, a set of data used by a program, or a user-created document. A file is the basic unit of storage that enables a computer to distinguish one set of information from another. A file is the "glue" that binds a conglomeration(sự kết hợp) of instructions, numbers, words, or images into a coherent (chặt chẽ, mạch lạc) unit that a user can retrieve, change, delete, save, or send to an output device (Microsoft Computer Dictionary)

Directory

- ***Directory*** :A catalog for filenames and other directories stored on a disk. A directory is a way of organizing and grouping the files so that the user is not overwhelmed (tràn) by a long list of them. The uppermost directory is called the root directory; the directories within a directory are called subdirectories. Depending on how an operating system supports directories, filenames in a directory can be viewed and ordered in various ways—for example, alphabetically, by date, by size, or as icons in a graphical user interface. What the user views as a directory is supported in the operating system by tables of data, stored on the disk, that indicate characteristics and the location of each file. In the Macintosh and Windows 9x operating systems, directories are called folders (MS Computer Dictionary)

Cơ bản về tập tin

- Lưu trữ trong tập tin: Trị nhị phân
- Trường hợp lưu trữ chuẩn (không mã hóa)

  - Ký tự: Mã ASCII- UNICODE

  - Chuỗi ký tự: Chuỗi mã ASCII của từng ký tự

  - Số: chuỗi Mã ASCII của ký số hoặc biểu diễn nhị phân của số.
- Đặc điểm truy xuất data trong tập tin: tuần tự từng byte.

Cơ bản về tập tin

- Hệ điều hành xác định 1 file bằng Absolute path (đường dẫn tuyệt đối) hay Relative path (đường dẫn tương đối). Đó là 1 chuỗi ký tự.
- Một thí dụ về absolute path trong C

  “C:\\TM1\\TM11\\f1.txt” hoặc

  “C:/TM1/TM11/f1.txt” hoặc
- Với 1 file nằm trong thư mục hiện hành, chỉ cần chỉ định file này bằng tên file ngắn gọn (relative path). Thí dụ : “nhanvien.dat”

Phân loại tập tin

- Tập tin văn bản : Dữ liệu lưu trữ là mã ASCII của ký tự ký số.
- Tập tin nhị phân: Dữ liệu lưu trữ là dạng nhị phân của dữ liệu.

Phân loại file

Hai dạng lưu trữ dữ liệu trong file của C:

- Text File: Mọi dữ liệu được lưu dạng ký tự ký số (mã ascii). Thí dụ có 2 dữ liệu "AB", 12 được lưu lên text file cách nhau khoảng trống:

  01000001 01000010 00100000 00110001 00110010
  'A' 'B' ' ' '1' '2'

  Có chuyển dạng lưu trữ từ số → chữ

- Binary file: Mọi dữ liệu được lưu dạng biểu diễn nhị phân của dữ liệu ➔ Dữ liệu chữ được lưu dạng mã ASCII, còn dữ liệu số được lưu dạng nhị phân của số. Thí dụ có 2 dữ liệu "AB", 12(int) lưu lên binary file:

  01000001 01000010 00000000 00001100
  'A' 'B' binary format của 12

  Không cần chuyển dạng lưu trữ → hiệu qủa

Phân loại file

- Hai loại lưu trữ khác nhau nên việc truy xuất dữ liệu từ file ra biến cần đến các hàm thư viện khác nhau.
- Binary file được dùng để lưu trữ dữ liệu dạng cấu trúc có kích thước cố định. Khi ghi biến cấu trúc lên file sẽ ghi 1 khối có kích thước cố định, và khi đọc từ file ra biến cấu trúc cũng đọc từ file theo từng khối có kích thước cố định.
- Muốn lập trình với file cần phải biết dạng lưu trữ cũng như ý nghĩa của dữ liệu trong file.
- Input file : file có dữ liệu sẽ được truy cập để đưa vào biến.
- Output file: File sẽ chứa trị của biến khi trị của biến được ghi vào.

Khai báo

FILE* f ;

Chú ý:

- Kiểu FILE được khai báo trong thư viện stdio.h
- Khai báo File * f; là SAI.

Các bước đọc file

(1) Chọn sách	(1) Chọn file (hàm main làm)
(2) Mở sách để đọc	(2) Mở file để đọc
(3) Chọn vị trí sẽ đọc	(3) Chọn vị trí sẽ đọc
(4) Lặp khi chưa xong	(4) Lặp khi chưa xong
(4.1) Đọc data	(4.1) Đọc data từ file→biến
(4.2) Xử lý data	(4.2) Xử lý biến
(5) Đóng sách	(5) Đóng file

- **Nếu đọc dữ liệu từ đầu file thì bỏ qua bước chọn vị trí sẽ đọc. Vị trí hiện hành của file ngay sau khi mở file là đầu file.**
- **Nếu viết hàm thì tên file là tham số.**

Các bước ghi file

(1) Chọn sách
(2) Mở sách để ghi
(3) Chọn vị trí sẽ ghi
(4) Lặp khi chưa xong
  (4.1) Chuẩn bị data
  (4.2) Ghi data lên file
(5) Đóng sách

(1) Chọn file (hàm main làm)
(2) Mở file để ghi
(3) Chọn vị trí sẽ ghi
(4) Lặp khi chưa xong
  (4.1) Chuẩn bị giá trị cho biến
  (4.2) Ghi biến → File
(5) Đóng file

- **Nếu ghi dữ liệu ngay từ đầu file thì bỏ qua bước chọn vị trí sẽ ghi. Vị trí hiện hành của file ngay sau khi mở file là đầu file.**
- **Nếu viết hàm thì tên file là tham số.**
- **Nếu giá trị của biến đã có rồi thì bỏ qua bước 4.1**

Mở file

- **FILE* fopen (const char* filename, const char* mode);**
- mode

Text file	Binary file	Description
"r", "rt"	"rb"	read
"w", "wt"	"wb"	write, ghi đè nếu file đã tồn tại
"a"	"a"	append, mở mới nếu file chưa có
"r+", "r+t"	"r+b"	mở để vừa đọc vừa ghi
"w+", "w+t"	"w+b"	mở để vừa đọc vừa ghi
"a+", "a+t"	"a+b"	mở để vừa đọc + ghi ở cuối file, mở mới nếu file chưa tồn tại.

- Giá trị trả về: NULL : thất bại Khác NULL: thành công.
- Thí dụ: FILE* f = fopen("c:/tm1/tm2/f1.txt" , "r");

Đóng file

- **int fclose(FILE* f);**
- Trả trị: 0: thành công
- Trả trị EOF (hằng -1) thất bại
- Thí dụ:

```
int  FileExist (const char* fname){
   FILE* f = fopen(fname, “r”);
   int result = (f==NULL)? 0 : 1;
   if (f)
       fclose(f);
   return result;
}
```

Kiểm tra hết file chưa?

- int feof(FILE* f)
- Trả trị: 1: Hết file rồi
- Trả trị 0: chưa hết file

Di chuyển trong file

- Về đầu file : **void rewind (FILE* f);**
- Dời n byte từ 1 vị trí:

  fseek (FILE* f, long n , long pos);

  pos = SEEK_SET (hằng 0) : từ đầu file

  pos = SEEK_CUR (hằng 1): từ vị trí hiện hành

  pos = SEEK_END (hằng 2): từ cuối file

  n>0 : dời tới, n<0: dời lui, n=0: đứng tại chỗ

  Dời qúa biên: gây lỗi

Thao tác với i vị trí hiện hành

- Lấy vị trí hiện hành trong file:
  long ftell(FILE* f);
- Lưu vị trí hiện hành
  int fgetpos(FILE *f, fpos_t *pos);
- Thiết lập vị trí hiện hành từ vị trí đã lưu
  int fsetpos(FILE *f, const fpos_t *pos);
- Thí dụ:

```
long filelength(const char* fname){
  FILE *f = fopen(fname, “r”);
   fseek(f , 0, SEEK_END);
   long len = ftell(f);
   fclose(f);
   return len;
}
```

Đổi tên/ Huỷ 1 file đã đóng

- int rename (const char* oldName, const char* newName);
- int remove (const char* filename);
- Trả trị 0: thành công
- Trả trị -1: thất bại
- OS không thể đổi tên/ hủy 1 file đang mở

Đọc/ghi dữ liệu với file đã mở

Thao tác với	Đọc file → biến	Ghi biến → file
Ký tự char c	c=fgetc(f);	fputc(c,f);
Chuỗi ký tự char S[];	fgets(S, n, f)	fputs (S,f);
Số	fscanf(f,"format", &var);	fprintf(f,"format", var);
n Cấu trúc nhị phân	size_t fread (&Rec,sizeof(struct),n,f)	int fwrite (&Rec,sizeof(struct), n,f)