Chương 8

Quản lý tập tin

Giảng viên: Nguyễn Chiến Thắng
Email: thangnc.haui@gmail.com

Nội dung

- Streams và file
- Các loại streams
- Các hàm xử lý tập tin
- Con trỏ tập tin
- Con trỏ hiện hành

Nhập/Xuất Tập Tin

- Tất cả các thao tác nhập/xuất trong C đều được thực hiện bằng các hàm trong thư viện chuẩn
- Nhập/xuất trong C/C++ có thể theo 2 cách:
  - Dạng nhị phân
  - Dạng văn bản

Streams

- Hệ thống tập tin của C/C++ làm việc với rất nhiều thiết bị khác nhau bao gồm máy in, ổ đĩa và các thiết bị đầu cuối

=> streams

- Vì mọi streams đều hoạt động tương tự, nên việc quản lý các thiết bị khác nhau rất dễ dàng
- Có hai loại streams
  - stream nhị phân
  - stream văn bản

Streams văn bản

- Một streams văn bản là một chuỗi các ký tự có thể được tổ chức thành các dòng kết thúc bằng một ký tự sang dòng mới
- Trong một stream văn bản, có thể xảy ra một vài sự chuyển đổi ký tự khi môi trường yêu cầu
- Số lượng ký tự được ghi (hay đọc) có thể không giống như số lượng ký tự ở thiết bị ngoại vi

Streams nhị phân

- Một streams nhị phân là một chuỗi các byte với sự tương ứng một-một với thiết bị ngoại vi, nghĩa là, không có sự chuyển đổi ký tự.
- Số lượng byte đọc (hay ghi) cũng sẽ giống như số lượng byte ở thiết bị ngoại vi
- Kết thúc của tập tin được xác định bằng kích thước của tập tin

Tập Tin

- Một tập tin có thể tham chiếu đến bất cứ thứ gì từ một tập tin trên ổ cứng đến một thiết bị đầu cuối hay một máy in
- Một tập tin kết hợp với một stream bằng cách thực hiện thao tác mở và ngưng kết hợp bằng thao tác đóng
- Khi một chương trình kết thúc bình thường, tất cả các tập tin đều tự động đóng
- Khi một chương trình kết thúc bất thường, các tập tin vẫn còn mở

Các hàm cơ bản thao tác tập tin

Tên	Chức năng
fopen()	Mở một tập tin
fclose()	Đóng một tập tin
fputc()	Ghi một ký tự vào một tập tin
fgetc()	Đọc một ký tự từ một tập tin
fread()	Đọc từ một tập tin vào một vùng đệm
fwrite()	Ghi từ một vùng đệm vào tập tin
fseek()	tìm một vị trí nào đó trong tập tin
fprintf()	Hoạt động giống như printf(), nhưng trên một tập tin
fscanf()	Hoạt động giống như scanf(), nhưng trên một tập tin
feof()	Trả về true nếu đã đến cuối tập tin
ferror()	Trả về true nếu xảy ra một lỗi
rewind()	Đặt lại con trỏ định vị trí bên trong tập tin về đầu tập tin
remove()	Xóa một tập tin
fflush()	Ghi dữ liệu từ một vùng đệm bên trong vào một tập tin xác định

Con trỏ tập tin

- Một con trỏ tập tin phải cần cho việc đọc và ghi các tập tin
- Nó là một con trỏ đến một cấu trúc chứa thông tin về tập tin. Thông tin bao gồm tên tập tin, vị trí hiện tại của tập tin
- Định nghĩa lấy từ studio.h bao gồm một khai báo cấu trúc tên FILE
- Câu lệnh khai báo cho một con trỏ tập tin:

```
FILE *fp
```

Mở tập tin văn bản

- Hàm fopen() mở một stream để sử dụng và liên kết một tập tin với stream đó
- Hàm fopen() trả về con trỏ kết hợp với tập tin
- Nguyên mẫu của hàm fopen() là:

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

Chế độ	Ý nghĩa
r	Mở một tập tin văn bản để đọc
w	Tạo một tập tin văn bản để ghi
a	Nối vào một tập tin văn bản
r+	Mở một tập tin văn bản để đọc/ghi
w+	Tạo một tập tin văn bản để đọc/ghi
a+f	Nối hoặc tạo một tập tin văn bản để đọc/ghi

Đóng tập tin văn bản

- Việc đóng một tập tin sau khi sử dụng là một điều quan trọng
- Đóng một stream sẽ làm sạch và chép vùng đệm kết hợp của nó ra ngoài, một thao tác quan trọng để tránh mất dữ liệu khi ghi ra ổ cứng
- Hàm fclose() đóng một stream đã được mở bằng hàm fopen()
- Nguyên mẫu của hàm fclose() là :

```
int  fclose(FILE *fp);
```

- Hàm fcloseall() đóng tất cả các streams đang mở
- Nếu đóng file thành công thì trả về giá trị 0, ngược lại trả về EOF (End of file).

Ghi một ký tự (tập tin văn bản)

- Streams có thể được ghi vào tập tin theo cách từng ký tự một hoặc theo từng chuỗi
- Hàm fputc() được sử dụng để ghi các ký tự vào tập tin đã được mở trước đó bằng hàm fopen().
- Nguyên mẫu của hàm:

```
int fputc(int ch, FILE *fp);
```

Đọc một ký tự (tập tin văn bản)

- Hàm fgetc() được dùng để đọc các ký tự từ một tập tin đã được mở bằng hàm fopen() ở chế độ đọc
- Nguyên mẫu của hàm:

```
int fgetc(int ch, FILE *fp);
```

- Hàm fgetc() trả về ký tự kế tiếp của vị trí hiện hành trong stream input, và tăng con trỏ định vị trí bên trong tập tin

Nhập/Xuất chuỗi (tập tin văn bản)

- Các hàm `fputs()` and `fgets()` ghi vào và đọc ra các chuỗi ký tự từ tập tin trên đĩa
- Hàm `fputs()` viết toàn bộ chuỗi vào stream đã định
- Hàm `fgets()` đọc một chuỗi từ stream đã cho cho đến khi đọc được một ký tự sang dòng mới hoặc sau khi đã đọc được `length - 1` ký tự.
- Nguyên mẫu của các hàm này là:

```
int fputs(const char *str, FILE *fp);

char *fgets(char *str, int length, FILE *fp);
```

Mở tập tin nhị phân

- Hàm fopen() mở một stream để sử dụng và liên kết một tập tin với stream đó
- Hàm fopen() trả về con trỏ kết hợp với tập tin
- Nguyên mẫu của hàm fopen() là:

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

Chế độ	Ý nghĩa
rb	Mở một tập tin nhị phân để đọc
wb	Tạo một tập tin nhị phân để ghi
ab	Nối vào một tập tin nhị phân
r+b	Mở một tập tin nhị phân để đọc/ghi
w+b	Tạo một tập tin nhị phân để đọc/ghi
a+b	Nối hoặc tạo một tập tin nhị phân để đọc/ghi

Đóng tập tin nhị phân

- Hàm fclose() đóng một stream đã được mở bằng hàm fopen()
- Nguyên mẫu của hàm fclose() là :

```
int  fclose(FILE *fp);
```

Đọc/Ghi khối dữ liệu

- Hàm fread() và fwrite() là các hàm đọc hoặc ghi dữ liệu không định dạng.
- Chúng được dùng để đọc ra và viết vào tập tin toàn bộ khối dữ liệu.
- Hầu hết các chương trình ứng dụng hữu ích đều đọc và ghi các kiểu dữ liệu do người dùng định nghĩa, đặc biệt là các cấu trúc.
- Nguyên mẫu của các hàm này là:

```
size_t fread(void *buffer, size_t num_bytes,
                    size_t  count, FILE *fp);
size_t fwrite(const void *buffer, size_t num_bytes,
                    size_t  count, FILE *fp);
```

End of file (EOF)

- Hàm feof() trả về true nếu đã đến cuối tập tin, nếu không nó trả về false (0).
- Hàm này được dùng trong khi đọc dữ liệu nhị phân.
- Nguyên mẫu là:

```
int feof(FILE *fp);
```

Hàm rewind()

- Hàm rewind() đặt lại con trỏ định vị trí bên trong tập tin về đầu tập tin
- Đối số truyền vào là con trỏ tập tin
- Cú pháp:

```
rewind(fp);
```

Hàm ferror()

- Hàm ferror() xác định liệu một thao tác trên tập tin có sinh ra lỗi hay không
- Vì mỗi thao tác đặt lại tình trạng lỗi, hàm ferror() phải được gọi ngay sau mỗi thao tác; nếu không, lỗi sẽ bị mất
- Nguyên mẫu của hàm là:

```
int ferror(FILE *fp);
```

Xóa tập tin

- Hàm remove() xóa một tập tin đã cho
- Nguyên mẫu của hàm:

```
int remove(char *filename);
```

Làm sạch các stream

- Hàm fflush() sẽ làm sạch vùng đệm và chép những gì có trong vùng đệm ra ngoài tùy theo kiểu tập tin
  - Một tập tin được mở để đọc sẽ có vùng đệm nhập liệu trống
  - Một tập tin được mở để ghi thì vùng đệm xuất của nó sẽ được ghi vào tập tin
- Nguyên mẫu của hàm là:

```
int fflush(FILE *fp);
```

Con trỏ hiện hành (curp)

- Một con trỏ được duy trì trong cấu trúc FILE để lần theo vị trí nơi mà các thao tác nhập/xuất đang diễn ra
- Mỗi khi một ký tự được đọc từ hay ghi vào một stream, con trỏ kích hoạt hiện hành (gọi là curp) được tăng lên
- Vị trí hiện hành của con trỏ này có thể được tìm thấy bằng sự trợ giúp của hàm ftell().
- Nguyên mẫu của hàm là:

```
long int ftell(FILE *fp);
```

Đặt lại vị trí con trỏ hiện hành

- Hàm fseek() định lại vị trí của curp dời đi một số byte tính từ đầu, từ vị trí hiện hành hay từ cuối stream là tùy vào vị trí được qui định khi gọi hàm fseek()
- Nguyên mẫu của hàm là:

```
int fseek (FILE *fp, long int offset, int origin);
```

origin	Vị trí trong tập tin
SEEK_SET hay 0	Bắt đầu tập tin
SEEK_CUR hay 1	Vị trí của con trỏ trong tập tin hiện hành
SEEK_END hay 2	Cuối tập tin

Thao tác file bằng thư viện <fstream>

Kiểu dữ liệu	Miêu tả
ofstream	Kiểu dữ liệu này biểu diễn Output File Stream và được sử dụng để tạo các file và để ghi thông tin tới các file đó
ifstream	Kiểu dữ liệu này biểu diễn Input File Stream và được sử dụng để đọc thông tin từ các file
fstream	Kiểu dữ liệu này nói chung biểu diễn File Stream, và có các khả năng của cả ofstream và ifstream, nghĩa là nó có thể tạo file, ghi thông tin tới file và đọc thông tin từ file

Thank you!