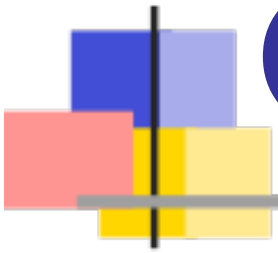
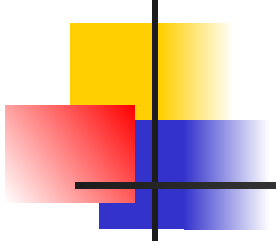


Quản lý tập tin

Bài 12

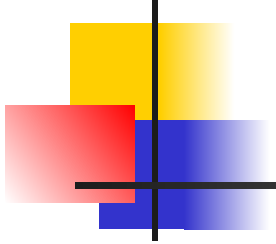


Mục tiêu

- Giải thích streams và file
- Thảo luận về các streams văn bản và streams nhị phân
- Giải thích các hàm xử lý tập tin
- Giải thích về con trỏ tập tin
- Thảo luận về con trỏ hiện hành
- Giải thích về các đối số dòng lệnh

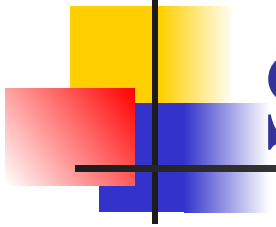
Nhập/Xuất Tập Tin

- Tất cả các thao tác nhập/xuất trong C đều được thực hiện bằng các hàm trong thư viện chuẩn
- Tiếp cận này làm cho hệ thống tập tin của C rất mạnh và uyển chuyển
- Nhập/xuất trong C có thể theo 2 cách: dữ liệu có thể truyền ở dạng biểu diễn nhị phân bên trong của nó hay ở dạng văn bản mà con người có thể đọc được



Streams

- Hệ thống tập tin của C làm việc với rất nhiều thiết bị khác nhau bao gồm máy in, ổ đĩa, ổ băng từ và các thiết bị đầu cuối
- Mặc dù tất cả các thiết bị đều khác nhau, hệ thống tập tin có vùng đệm sẽ chuyển mỗi thiết bị về một thiết bị logic gọi là một stream
- 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 văn bản và stream nhị phân



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
- Vì vậy, mối quan hệ giữa các ký tự được ghi (hay đọc) và những ký tự ở thiết bị ngoại vi có thể không phải là mối quan hệ một-một
- Và cũng vì sự chuyển đổi có thể xảy ra này, 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
- Các stream nhị phân là các chuỗi byte thuần túy, mà không có bất kỳ ký hiệu nào dùng để chỉ ra điểm kết thúc của tập tin hay kết thúc của mẫu tin
- 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 đĩa 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 Về 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, liệu tập tin có đang được đọc hay ghi, và liệu có bất kỳ lỗi nào xuất hiện hay đã đến cuối tập tin
- Định nghĩa lấy từ studio.h bao gồm một khai báo cấu trúc tên FILE
- Câu lệnh khai báo duy nhất cần thiết cho một con trỏ tập tin là: **FILE *fp**

Mở Một 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 Một Tập Tin Văn Bản

- Việc đóng một tập tin sau khi sử dụng là một điều quan trọng
- Thao tác này sẽ giải phóng tài nguyên và làm giảm nguy cơ vượt quá giới hạn số tập tin có thể mở.
- 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 đĩa
- 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

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 này là:

  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 là:

  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 lên

Nhập Xuất Chuỗi

- 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 Một 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 một con trỏ tập tin 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 vào 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);

Hàm fread() và fwrite()

- 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);
```

Sử Dụng feof()

- 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
- Nó lấy con trỏ tập tin làm đối số
- 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);

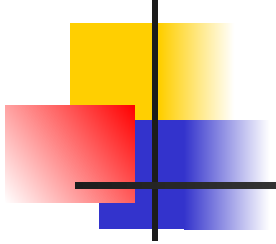
Xóa Tập Tin

- Hàm remove() xóa một tập tin đã cho
- Nguyên mẫu của hàm là:
  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, trong khi 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);
- Hàm fflush(), không có đối số, sẽ làm sạch tất cả các tập tin đang mở để xuất



Các Stream Chuẩn

Mỗi khi một chương trình C bắt đầu thực thi dưới DOS, hệ điều hành sẽ tự động mở 5 stream đặc biệt:

- Nhập chuẩn (stdin)
- Xuất chuẩn (stdout)
- Lỗi chuẩn (stderr)
- Máy in chuẩn (stdprn)
- Thiết bị phụ trợ chuẩn (stdaux)

Con Trỏ Kích Hoạt Hiện Hành

- 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 Hiện Hành - 1

- 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);
```

Đặt Lại Vị Trí Hiện Hành - 2

- origin chỉ định vị trí bắt đầu tìm kiếm và phải có giá trị như sau:

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

fprintf() và fscanf()-1

- Hệ thống nhập xuất có vùng đệm bao gồm các hàm **fprintf**() và **fscanf**() tương tự như hàm **printf**() và **scanf**() ngoại trừ rằng chúng thao tác trên tập tin
- Nguyên mẫu của các hàm này là:

```
int fprintf(FILE * fp,
        const char *control_string,..);
int fscanf(FILE *fp,
        const char *control_string,...);
```

fprintf() và fscanf() - 2

- Mặc dù fprintf() và fscanf() là cách dễ nhất nhưng không phải luôn luôn là hiệu quả nhất
- Mỗi lời gọi phải mất thêm một khoảng thời gian overhead, vì dữ liệu được ghi theo dạng ASCII có định dạng chứ không phải theo định dạng nhị phân
- Vì vậy, nếu tốc độ và độ lớn của tập tin là vấn đề đáng ngại, thì fread() và fwrite() sẽ là lựa chọn tốt hơn