6. File

Nội dung

- File văn bản
- Các chế độ mở file văn bản
- Vào ra với file văn bản

6.1 Các thao tác cơ bản

6. File

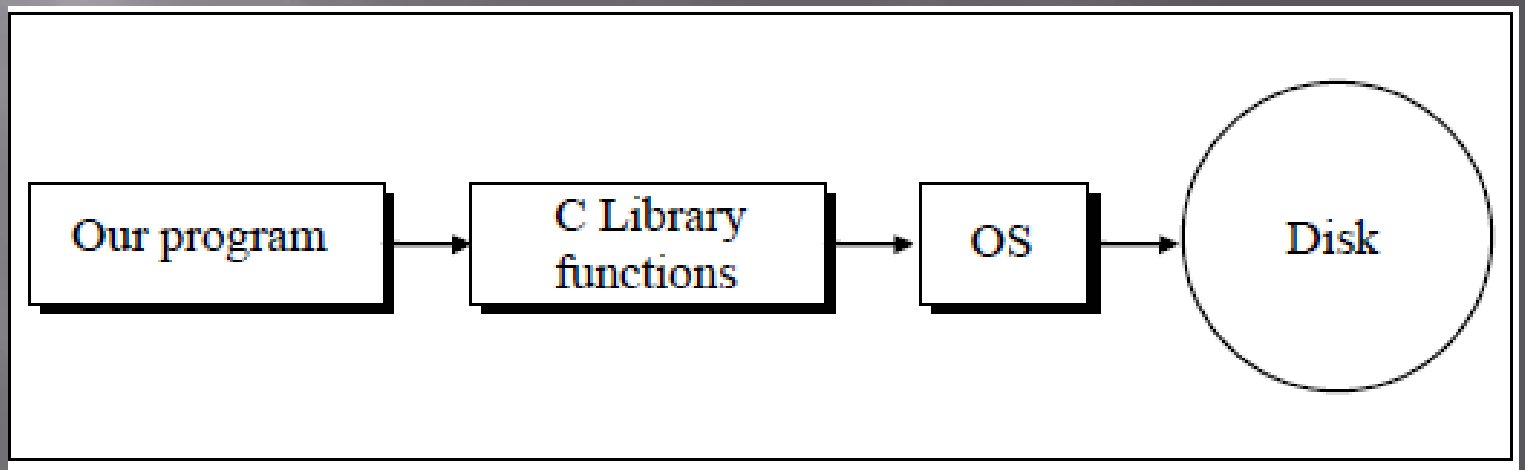
- Màn hình máy tính có khả năng hiển thị hữu hạn
- Bộ nhớ trong của máy tính có dung lượng nhỏ, dữ liệu lưu trong bộ nhớ trong sẽ bị mất khi chương trình kết thúc hoặc tắt máy

  ➔ ***làm thế nào để lưu các dữ liệu với kích thước lớn ?***
  ➔***Làm sao để ta không cần nhập lại dữ liệu mỗi khi chạy chương trình?***

- ***Giải pháp :*** *lưu trữ dữ liệu bằng file trên bộ nhớ thứ cấp (bộ nhớ ngoài)*

6. File

- Cách tổ chức dữ liệu trên đĩa:
  - Dữ liệu được tổ chức thành các file và thư mục
  - Dữ liệu được lưu trữ dưới dạng nhị phân
  - Cách lưu trữ dữ liệu nhị phân khác nhau trong các hệ thống khác nhau
  - Hệ điều hành quản lý việc lưu trữ dữ liệu, chương trình C sử dụng các hàm viết cho các hệ thống khác nhau để thực hiện vào ra dữ liệu



6.1 Các thao tác cơ bản

- Các thao tác cơ bản với file:
  - Tạo file mới
  - Mở một file đã có
  - Đọc dữ liệu từ file
  - Ghi dữ liệu ra file
  - Di chuyển đến một vị trí trong file (seeking)
  - Đóng file

- VD. Chương trình mở file data.dat nằm trên ổ đĩa C và hiển thị nội dung của file ra màn hình.

6.1 Các thao tác cơ bản

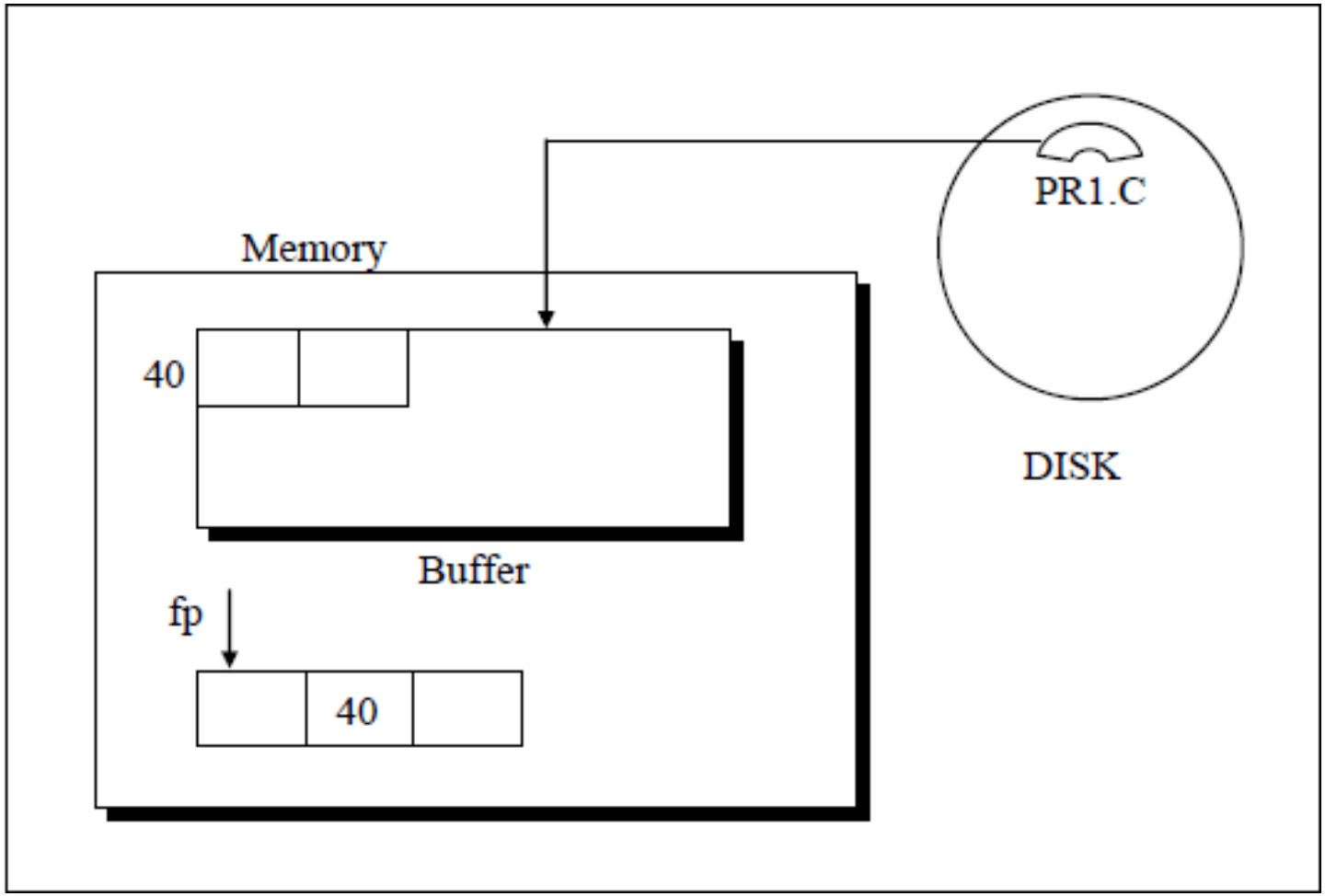
```
/* Hiển thị nội dung file data.dat ra màn hình. */
# include "stdio.h"
int main(void)
{
    FILE *fp ;
    char ch ;
    fp = fopen ( "C:\\data.dat", "r" ) ;
    do
    {
        ch = fgetc ( fp ) ;
        if (ch != EOF)
           printf ( "%c", ch ) ;
    }
    while(ch != EOF);
    fclose ( fp ) ;
    return 0;
}
```

6.1 Các thao tác cơ bản

- **Mở file**: để đọc (hoặc ghi) file thì trước hết cần mở file
- Dùng hàm **fopen()** với tham số là :
  - Tên file (và đường dẫn)
  - Xâu tham số (VD. Để đọc thì là “r”, để ghi là “w”)

- Các thao tác thực hiện khi mở file ở chế độ “r”
  1. Tìm file trên đĩa
  2. Nạp file từ đĩa vào một nơi trong bộ nhớ (gọi là buffer)
  3. Tạo ra con trỏ char trỏ vào ký tự đầu tiên trong buffer

6.1 Các thao tác cơ bản



6.1 Các thao tác cơ bản

- Khi đọc thành công fopen() sẽ trả về các thông tin được chứa trong cấu trúc FILE, fopen() trả về địa chỉ của cấu trúc này.

- Phải khai báo một biến con trỏ kiểu FILE để chứa địa chỉ trả về: **FILE *tên_biến_file;**

  VD. **FILE *fp;**

- Cấu trúc FILE được định nghĩa trong stdio.h

6.1 Các thao tác cơ bản

Kiểm tra lỗi khi mở FILE

- Khi mở file bị lỗi (file không tồn tại, hoặc đường dẫn sai ...) thì con trỏ file nhận giá trị trả về NULL
- Kiểm tra lỗi khi mở bằng giá trị con trỏ trả về!

```
FILE *fp ;
fp = fopen ( "C:\\data.dat", "r" ) ;
if ( fp == NULL )
{
    puts ( "cannot open file" ) ;
    exit( ) ;
}
```

6.1 Các thao tác cơ bản

Đọc nội dung từ FILE

- Đọc một ký tự tại con trỏ của file : fgetc()

```
ch=fgetc(fp);
```

- FILE được đánh dấu kết thúc bằng ký tự đặc biệt là **EOF** (end-of-file) (được định nghĩa trong stdio.h)

```
while(ch != EOF);
```

- Có thể dùng getc() thay cho fgetc()

```
ch=getc(fp);
```

6.1 Các thao tác cơ bản

Đóng file: sau khi thao tác xong với file ta cần đóng file để các ứng dụng khác có thể sử dụng file đó.

fclose(tên_biến_file);

VD. **fclose(fp);**

- Các thao tác khi đóng file
  - Các ký tự trong bộ nhớ đệm (buffer) sẽ được ghi ra file trên đĩa
  - Ký tự EOF (mã ASCII là 26) sẽ được ghi vào cuối file
  - Bộ nhớ đệm sẽ được giải phóng khỏi bộ nhớ

6.1 Các thao tác cơ bản

- Ví dụ: đọc một file văn bản, đếm và in ra màn hình số lượng ký tự, số lượng tab('\t'), cách trống(' ') và xuống dòng('\n').

```
# include "stdio.h"
int main(void)
{
    FILE *fp ;
    char ch ;
    int noLine = 0, noTab = 0, noBlank = 0, noChar = 0 ;
    fp = fopen ( "C:\\data.txt", "r" ) ;
    if(fp!=NULL)
    {
        do
        {
            ch = fgetc ( fp ) ;
            if ( ch != EOF )
                noChar++ ;
```

```
      if ( ch == ' ' )
                  noBlank++ ;
      if ( ch == '\n' )
                  noLine++ ;
            if ( ch == '\t' )
                  noTab++ ;
        }
        while(ch!=EOF);
        fclose ( fp ) ;
        printf ( "\nSo luong ky tu = %d", noChar ) ;
        printf ( "\nSo luong cach trong = %d", noBlank ) ;
        printf ( "\nSo luong tab = %d", noTab ) ;
        printf ( "\nSo luong dong = %d", noLine ) ;
    }
    else
        printf("Co loi khi mo file.\n");
    return 0;
}
```

6.1 Các thao tác cơ bản

Ghi file: ghi một ký tự ra file **fputc()**

- Ví dụ. Chương trình copy nội dung 2 file dùng fgetc() và fputc()

```
FILE *fSource,*fTarget ;
char ch ;
fSource = fopen ( "C:\\data.txt", "r" ) ;
fTarget = fopen ( "C:\\data_backup.txt", "w" ) ;
if(fSource!=NULL && fTarget!=NULL) {
    do{
        ch = fgetc (fSource) ;
        fputc(ch,fTarget);
    }
    while(ch!=EOF);
    fclose (fSource);  fclose (fTarget);
}
```

6.2 Các chế độ mở File

6.2 Các chế độ mở File

Tham số	Tác dụng	Khả năng
"r"	Tìm file, nếu có thì nạp vào bộ nhớ và trả về con trỏ trỏ vào ký tự đầu tiên, ngược lại trả về NULL	Đọc dữ liệu từ file
"w"	Tìm file, nếu đã tồn tại thì sẽ bị ghi đè, nếu không thì tạo ra một file mới. Trả về NULL nếu bị lỗi khi thực hiện	Ghi dữ liệu vào file
"a"	Tìm file, nếu tồn tại thì nạp nội dung vào bộ nhớ, con trỏ file trỏ vào ký tự cuối cùng. Nếu file chưa tồn tại thì tạo file mới. Trả về NULL nếu lỗi khi thực hiện	Ghi thêm dữ liệu mới vào cuối file

6.2 Các chế độ mở File

Tham số	Tác dụng	Khả năng
"r+"	Tìm file, nếu có thì nạp vào bộ nhớ và trả về con trỏ trỏ vào ký tự đầu tiên, ngược lại trả về NULL	Đcọ nội dung cũ, thêm nội dung mới, sửa đổi nội dung cũ
"w+"	Tìm file, nếu đã tồn tại thì sẽ bị ghi đè, nếu không thì tạo ra một file mới. Trả về NULL nếu bị lỗi khi thực hiện	Ghi nội dung mới vòa file, đọc lại, sửa đổi nội dung vừa ghi
"a+"	Tìm file, nếu tồn tại thì nạp nội dung vào bộ nhớ, con trỏ file trỏ vào ký tự cuối cùng. Nếu file chưa tồn tại thì tạo file mới. Trả về NULL nếu lỗi khi thực hiện	Đọc nội dung cũ, thêm nội dung mới vào cuối file (không thể sửa đổi nội dung cũ)

6.3 Đọc ghi File bằng string

6.3 Đọc ghi File bằng string

- Trong nhiều trường hợp, đọc ghi với xâu ký tự hiệu quả hơn đọc từng ký tự.
- **Đọc một xâu** :

```
fgets(tên_biến,độ_dài,tên_biến_file);
```

VD. fgets(str, 80, fp);

Khi đọc đến cuối file, nếu đọc tiếp thì giá trị sẽ nhận được là NULL → kiểm tra cuối hết file bằng kiểm tra giá trị NULL

- **Ghi một xâu** :

```
fputs(biến_xâu, tên_biến_file);
```

6.3 Đọc ghi File bằng string

- VD1. Chương trình nhập các xâu ký tự từ bàn phím và ghi vào một file.

```
 FILE *fp ;
char s[81] ;
fp = fopen ("C:\\POEM.TXT", "w") ;
if ( fp !=NULL)
{
    printf ("Nhap dong tiep theo:\n" ) ;
    while (strlen(gets(s)) > 0 )
    {
        fputs (s, fp) ;
        fputs ("\n", fp) ;
    }
    fclose(fp) ;
}
else
    printf("Co loi khi tao file.\n");
```

6.3 Đọc ghi File bằng string

- VD 2. Đọc nội dung file và hiển thị ra màn hình

```
FILE *fp ;
char s[81] ;
fp = fopen ("C:\\POEM.TXT", "r") ;
if ( fp !=NULL)
{
    while(fgets(s, 80, fp) != NULL)
    printf("%s", s) ;
    fclose(fp) ;
}
else
    printf("Co loi mo file.\n");
```

6.4 Đọc ghi File bằng các hàm vào ra

6.4 Hàm vào ra chuẩn với file

- Làm thế nào để đọc/ghi các dữ liệu có kiểu khác nhau với file? Làm thế nào để đọc/ghi dữ liệu số?

- Hàm vào ra với file trong thư viện chuẩn : **fprintf**() và **fscanf**(), cách dùng giống với các hàm vào ra với màn hình và bàn phím printf(), scanf()
- VD:

```
fprintf(fp,"%s %d %f\n", name, age, bs);

fscanf(fp,"%s %d %f", name, &age, &bs );
```

6.4 Hàm vào ra chuẩn với file

- Ví dụ. Đọc vào từ bàn phím điểm thi của cá thành viên trong lớp và ghi ra file.

```
# include "stdio.h"
#include <string.h>
int main(void)
{
    FILE *fp ;
    char hoTen[51];
    float diemThi;
    char traLoi;

    fp = fopen ( "C:\\diemThi.txt", "w" ) ;
    if(fp!=NULL)
    {
```

```
  do
  {
     printf("Nhap ho ten, diem tieo theo\n");
     printf("Ho ten: "); scanf("%s",hoTen);
     printf("diem thi: "); scanf("%f",&diemThi);

     fprintf(fp,"%s %f\n",hoTen,diemThi);

     printf ("Nhap them (Y/N): ");
     fflush (stdin);
     traLoi = getchar();
     fflush (stdin);
   }
   while(traLoi=='Y' ||traLoi=='y');
   fclose (fp) ;
}
else
    printf("Co loi khi mo file.\n");
    return 0;
}
```

6.4 Hàm vào ra chuẩn với file

- VD. Đọc lại nội dung file điểm thi và in ra màn hình

```
FILE *fp ;
char hoTen[51];
float diemThi;

fp = fopen ( "C:\\diemThi.txt", "r" ) ;
if(fp!=NULL)
{
   while ( fscanf ( fp, "%s %f", hoTen, &diemThi ) != EOF )
   printf ( "\n%s %f", hoTen, diemThi) ;
   fclose (fp) ;
}
else
   printf("Co loi khi mo file.\n");
```

6.4 Hàm vào ra chuẩn với file

- Vào/ra file có định dạng

fprintf(pfile, "%12d%12d%14f", num1, num2, fnum1);

fscanf(pfile, "%12d%12d%14f", &num1, &num2, &fnum1);

- Cách định dạng vào/ra với file giống với cách định dạng khi đọc dữ liệu và khi hiển thị ra màn hình.

Một số hàm xử lý file

- ***Đổi tên file***
  int rename(const char *oldname, const char *newname);

```
if(rename( "C:\\temp\\myfile.txt", "C:\\temp\\myfile_copy.txt"))
        printf("Failed to rename file.");
else
        printf("File renamed successfully.");
```

- ***Ghi nội dung trong bộ nhớ đệm ra file***: **fflush()**
  VD. fflush(fp);

- ***Xóa file***: **remove()**
  VD. remove(“C:\\pfile.txt");

stdin, stdout, và stderr

- Khi chương trình C thực hiện, có 3 file được từ động mở bởi hệ thống để sử dụng cho chương trình : stdin, stdout, và stderr.
- Các hàm vào chuẩn nhận đầu vào từ stdin (mặc dù không chỉ ra tham số này).

  VD. có thể dùng hàm fscanf() để đọc từ stdin giống như hàm scanf()

  ```
  fscanf(stdin, "%i",&x);   ≡   scanf("%i",&x);
  ```

- Tương tự có thể dùng fprintf() để đưa ra đầu ra chuẩn – stdout (mặc định là màn hình)

  ```
  fprintf(stdout, "x=%i",x);  ≡   printf("x=%i",x);
  ```

- stderr là file lỗi chuẩn, thường chỉ noi ghi các lỗi trong chương trình

  ```
  fprintf (stderr, "Loi khi mo file.\n");
  ```

Hàm exit

- Trong một số tình huống ta bắt buộc phải thoát khỏi chương trình đang thực hiện (vd. phát hiện có lỗi).
- Chương trình bình thường chỉ dừng khi thực hiện hết lện trong hàm main hoặc khi gặp return.
- Để thoát khỏi chương trình tại vị trí bất kỳ ta dùng exit()

```
exit(n);
```

n: trạng thái khi dừng (thường là các mã lỗi)

```
if ( (inFile = fopen (file, "r")) == NULL ) {
    fprintf (stderr, "Loi khi mo file %s.\n", file);
    exit (EXIT_FAILURE);
}
```