Chương VI

File (Tập Tin)

ĐẠI HỌC GTVT TP HỒ CHÍ MINH

Nội dung chính

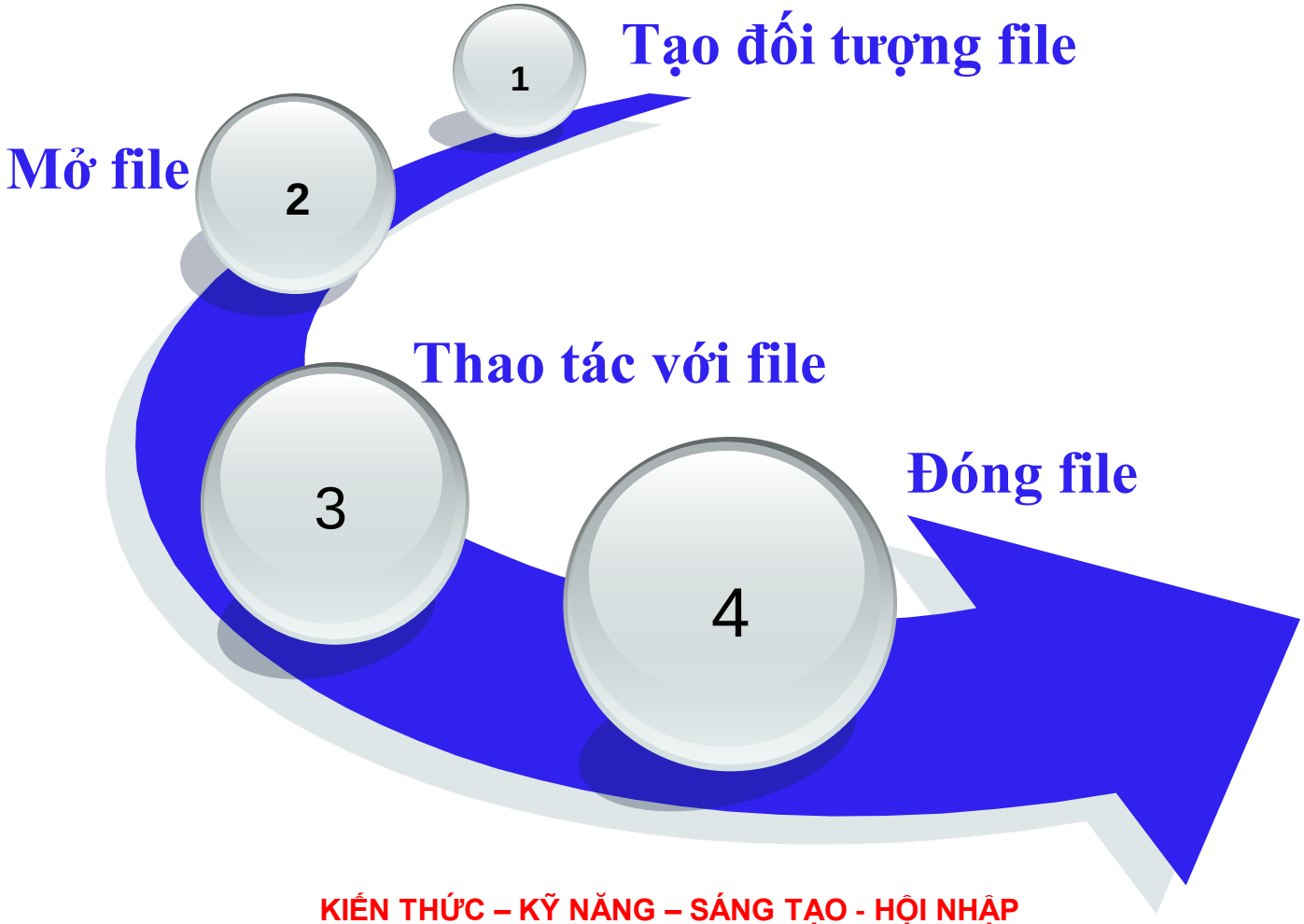
6.1 Dẫn nhập

- File là hình thức lưu trữ phổ biến trên bộ nhớ phụ, gồm 2 loại:
  - File văn bản (text file).
  - File nhị phân (binary file).
- **File văn bản:** chỉ lưu trữ thuần túy văn bản, trong đó các kí tự được biểu diễn bằng mã ASCII của nó, người dùng có thể đọc được.
  - Tính chất
    - Dễ truy xuất và xử lý
    - Độ bảo mật kém
    - Tốc độ truy xuất kém
    - Kích thước lớn

6.1 Dẫn nhập

- **File nhị phân:** là file chứa dữ liệu mà có ít nhất một vài chuỗi bit không thể biểu diễn ở dạng văn bản trơn. Do đó file này chỉ có máy đọc được, còn người không đọc được.
  - Tính chất
    - Truy xuất và xử lý phức tạp
    - Tốc độ truy xuất nhanh
    - Tính bảo mật cao hơn

Quy trình làm việc với file



6.2 Khai báo kiểu tập tin

Trong C++, để làm việc với các hàm xử lý file cần khai báo đầu chương trình: **#include <fstream>**

Trong thư viện fstream thì ta có 3 lớp stream cơ bản sau :

- **ifstream**: *Dùng cho file nhập vào. Loại này chỉ có thể được dùng để đọc dữ liệu từ file vào bộ nhớ mà thôi.*
- **ofstream**: *Dùng cho file xuất ra. Loại này thì có thể dùng để tạo ra files và chép dữ liệu vào chúng.*
- **fstream**: *Đây là kênh file.(File stream). Loại này thì có thể vừa tạo file, vừa ghi dữ liệu vào file và đọc dữ liệu từ file vào luôn.*

6.2 Khai báo kiểu tập tin

- Để định nghĩa một đối tương file ta chọn các cách sau:

  fstream *tên biến file***;**

  ifstream *tên biến file***;**

  ofstream *tên biến file***;**

- Mở file:

  *tên biến file***.open(***tên file***,** *chế độ mở file***);**

- Vừa khai báo vừa mở file:

  fstream/ofstream/ifstream *tên biến file***(***tên file***,** *chế độ mở***);**

- Đóng file:

  *tên biến file***.close();**

6.2 Khai báo kiểu tập tin

- Chế độ mở file:
  - ios::in: Mở một file để đọc.
  - ios::out: Mở một file có sẵn để ghi.
  - ios::app: Mở một file có sẵn để thêm dữ liệu vào cuối file.
  - ios::ate: Mở file và đặt con trỏ file vào cuối file.
  - ios::trunc: Nếu file đã có sẵn thì dữ liệu của nó sẽ bị mất.
  - ios::nocreate: Mở một file, file này bắt buộc phải tồn tại.
  - ios::noreplace: Chỉ mở file khi file chưa tồn tại.
  - ios::binary: Mở một file ở chế độ nhị phân.
  - ios::text: Mở một file ở chế độ văn bản.

6.2 Khai báo kiểu tập tin

- Mặc định:

ofstream	ios::out
ifstream	ios::in
fstream	ios::in | ios::out

Ví dụ: Khai báo một biến myFile, dùng để mở file có tên và đường dẫn là myDir\abc.txt ở chế độ để đọc dữ liệu (ios::in):

```
fstream myFile("myDir\\def.txt", ios::in);
```

hay:
```
fstream myFile;
myFile.open ("myDir\\def.txt", ios::in);
```

hay:
```
ifstream myFile("myDir\\def.txt");
```

hay:
```
ifstream myFile;
myFile.open ("myDir\\def.txt");
```

6.2 Khai báo kiểu tập tin

- Chú ý:
  - Mở một file đồng thời ở nhiều chế độ khác nhau, bằng cách kết hợp các bít chỉ thị tương ứng bằng phép toán or bít "|".

Ví dụ: Mở một file abc.txt để đọc (ios::in) đồng thời với để ghi (ios::out) dưới chế độ văn bản (ios::text), ta khai báo như sau:

```
fstream myFile("abc.txt",
ios::in|ios::out|ios::text);
```

6.2 Khai báo kiểu tập tin

- **Mở file văn bản:**
  - Mở một file dưới chế độ văn bản:

    fstream *Tên biến file*(*Tên file*, ios::text);
  - Khi đó các thao tác đọc, ghi trên biến file được thực hiện theo đơn vị là các từ, được phân cách bởi dấu trống (space bar) hoặc dấu xuống dòng (enter).

Ví dụ:

Mở một file BaiTho.txt dưới chế độ văn bản (ios::text), ta khai báo như sau:

fstream myFile("BaiTho.txt", ios::text);

6.2 Khai báo kiểu tập tin

- **Mở file nhị phân:**
  - Mở một file dưới chế độ nhị phân:

    fstream *Tên biến file(Tên file,* ios::binary*);*
  - Khi đó các thao tác đọc, ghi trên biến file được thực hiện theo đơn vị byte theo kích thước các bản ghi (cấu trúc) được ghi trong file

Ví dụ:

Mở một file DuLieu.dat dưới chế độ nhị phân (ios::binary), ta khai báo như sau:

fstream myFile("DuLieu.dat", ios::binary);

Tập tin văn bản

- **Thao tác trên file văn bản**

  1. Mở file theo chế độ để ghi/đọc/... bằng đối tượng ofstream/ifstream.

     ofstream *Tên biến file(Tên file)*;

     ifstream *Tên biến file(Tên file)*;

  2. Ghi/đọc dữ liệu vào file bằng << , >> hay put(), get().

     Tên biến file << *Dữ liệu*;

     Tên biến file >> *Biến dữ liệu*;

  3. Đóng file bằng lệnh close().

     Tên biến file.close();

6.3 Các thao tác truy xuất tập tin

Ví dụ 1: Minh họa việc ghi dữ liệu vào file

- *Tên file được người dùng tự nhập vào từ bàn phím.*
- *Chương trình sẽ ghi vào file các kí tự do người dùng gõ vào từ bàn phím, mỗi kí tự được phân cách nhau bởi dấu trống (space bar).*
- *Chương trình dừng lại khi người dùng nhập kí tự '.'. Và file được kết thúc bằng một dấu xuống dòng "endl".*

```
#include <iostream>
#include <fstream>
const int length = 25; // Độ dài tối đa tên file
using namespace std;
int main () {
    char fileName[length], input;
    cout << "Ten tap tin: ";
    cin.get(fileName, length); // Nhập tên file
```

6.3 Các thao tác truy xuất tập tin

```
  // Mở file
  ofstream fileOut(fileName); // Khai báo và mở file
  if(!fileOut) {
     cout << "Khong the tao duoc tap tin " << fileName << endl;
     exit(1);
   }
  // Ghi dữ liệu vào file
  do {
        input=cin.get(); // Đọc kí tự từ bàn phím
        fileOut << input << ' '; // Ghi kí tự vào file
  }  while((input != '.'); //Kết thúc khi nhập vào dấu .
  fileOut << endl; // Xuống dòng cuối file
  // Đóng file
  fileOut.close(); // Đóng file
  return;
}
```

6.3 Các thao tác truy xuất tập tin

- Có thể thay << bằng put():

```
// Ghi dữ liệu vào file sử dụng put
    do {
        input=cin.get(); // Đọc kí tự từ bàn phím
        fileOut.put(input); ; // Ghi kí tự vào file
        fileOut.put(‘ ‘);
    }  while((input != ‘.’); //Kết thúc khi nhập vào dấu .
    fileOut.put(‘\n’); // Xuống dòng cuối file
```

6.3 Các thao tác truy xuất tập tin

Ví dụ 2: Minh hoạ việc đọc dữ liệu từ tập tin vừa sử dụng trong ví dụ 1 ra màn hình:

- *Tên tập tin được người dùng tự nhập vào từ bàn phím.*
- *Chương trình sẽ đọc các kí tự trong file và hiển thị ra màn hình, mỗi kí tự được phân cách nhau bởi dấu trống (space bar).*
- *Chương trình dừng lại khi kết thúc tập tin.*

```
#include <iostream>
#include <fstream>
const int length = 25; // Độ dài tối đa tên file
using namespace std;
int main () {
    char fileName[length], input;
    cout << “Ten tap tin: ”;
    cin.get(fileName, length); // Nhập tên file
```

6.3 Các thao tác truy xuất tập tin

```
    // Mở file
    ifstream fileIn(fileName); // Khai báo và mở file
    if(!fileIn) {
       cout << “Khong the mo duoc tap tin ” << fileName << endl;
       exit(1);
     }
    /* Đọc dữ liệu từ file ra màn hình */
    while(fileIn){
            fileIn >> intput; // Đọc kí tự từ file
            cout << intput; // Ghi kí tựra màn hình
    }
    cout << endl; // Xuống dòng trên màn hình
    /* Đóng file */
    fileIn.close(); // Đóng file
    return;
}
```

6.3 Các thao tác truy xuất tập tin

- Có thể thay >> bằng get():

```
// Đọc dữ liệu từ file sử dụng get

…
   while(fileIn.get(iutput){
            cout << iutput; // Ghi kí tự ra màn hình
   }
   cout << endl; // Xuống dòng trên màn hình

…
```

6.3 Các thao tác truy xuất tập tin

Ví dụ 3: Minh hoạ việc copy toàn bộ nội dung của một tập tin sang một tập tin mới:

- *Tên tập tin nguồn và tập tin đích được nhập từ bàn phím bởi người dùng.*
- *tập tin nguồn được mở ở chế độ đọc.*
- *tập tin đích được mở ởchế độ ghi.*
- *Đọc từng kí tự từ tập tin nguồn và ghi ngay vào tập tin đích.*
- *Đóng các tập tin khi kết thúc*

```
#include <iostream>
#include <fstream>
const int length = 25; // Độ dài tối đa tên file
using namespace std;
int main () {
    char sourceFile[length], targetFile[length], data;
```

6.3 Các thao tác truy xuất tập tin

```
 cout << “Ten tap tin nguon: ”;
 cin.get(sourceFile, length);
 cout << “Ten tap tin dich: ”;
 cin.get(targetFile, length);

/* Mở file nguồn */
ifstream fileIn(sourceFile);// Khai báo và mở file nguồn
if(!fileIn){ // Không mở được file nguồn
    cout << “Khong the mo duoc tap tin nguon ” << sourceFile << endl;
    exit(1);
}
/* Mở file đích */
ofstream fileOut(targetFile);// Khai báo và mở file đích
if(!fileOut){ // Không mở được file đích
    cout << “Khong the tao duoc tap tin dich ” << targetFile << endl;
    exit(1);
}
```

6.3 Các thao tác truy xuất tập tin

```
    /* Đọc dữ liệu từ file nguồn ghi ra và vào file đích */
    while(fileIn) {
        fileIn >> data; // Đọc kí tự từ file nguồn
        fileOut << data; // Ghi kí tự ra file đích
    }
    /* Sử dụng get và put
        while (fileIn.get(data))
            fileOut.put(data);
    */
    /* Đóng các file */
    fileIn.close(); // Đóng file nguồn
    fileOut.close(); // Đóng file đích
    return;
}
```

Tập tin nhị phân

- **Nhập xuất file nhị phân bằng read và write:**

1. Mở file theo chế độ để ghi nhị phân bằng đối tượng fstream.
   fstream *Tên biến file*(*Tên file*, ios::out|ios::binary);
   fstream *Tên biến file*(*Tên file*, ios::in|ios::binary);

2. Ghi/đọc dữ liệu vào file bằng write()/read().
   *Tên biến file.*write(char* *Dữ liệu, int Kích thước dữ liệu);*
   *Tên biến file.*read(char* *Dữ liệu ra, int Kích thước dữ liệu);*

3. Đóng file bằng close().
   Tên biến file.close();

6.2 Khai báo kiểu tập tin

- Tham số trong write:
  - Tham số thứ nhất là con trỏ kiểu char trỏ đến vùng dữ liệu cần ghi vào file. Vì con trỏ bắt buộc có kiểu char nên khi muốn ghi dữ liệu có kiểu khác vào file, ta dùng hàm chuyển kiểu:

    reinterpret_cast <char *>(Dữ liệu);
  - Tham số thứ hai là kích cỡ dữ liệu được ghi vào file. Kích cỡ này được tính theo byte, nên thông thường ta dùng toán tử:

    sizeof(Kiểu dữ liệu);

6.3 Các thao tác truy xuất tập tin

- Chú ý:
  - Phải sử dụng chế độ đọc/ghi file nhị phân khi muốn đọc, ghi các dữ liệu có cấu trúc (struct) vào file.
  - Khi đọc/ghi dữ liệu kiểu cấu trúc, để toán tử sizeof() thực hiện chính xác thì các thành viên của cấu trúc không được là kiểu con trỏ.

6.3 Các thao tác truy xuất tập tin

Ví dụ 4: Minh hoạ việc ghi dữ liệu vào tập tin nhị phân, dữ liệu là kiểu cấu trúc:

- *Tên file và số lượng mẫu tin được người dùng tự nhập vào từ bàn phím.*
- *Chương trình sẽ ghi vào file các mẫu tin có cấu trúc do người dùng gõ vào từ bàn phím.*

```
#include <iostream>
#include <fstream>
const int length = 25; // Độ dài tối đa tên file
using namespace std;
struct Date {
        int day; // Ngày
        int month; // Tháng
        int year; // Năm
};
```

6.3 Các thao tác truy xuất tập tin

```
struct  Employee {
            char name[20]; // Tên nhân viên
            Date birthDay; // Ngày sinh của nhân viên
            char role[20]; // Chức vụ của nhân viên
            float salary; // Lương của nhân viên
};
int main() {
   char fileName[length];
   cout << “Ten tap tin: ”; cin.get(fileName, length); // Nhập tên file
   int recordNumber; // Số lượng record
   cout << “So luong mau tin: ”;   cin >> recordNumber;
   // Khai báo và mở file
   fstream fileOut(fileName, ios::out|ios::binary);
   if(!fileOut) { // Không mở được file
      cout << “Khong the tao duoc tap tin ” << fileName << endl;
      exit(1);
   }
```

6.3 Các thao tác truy xuất tập tin

```
/* Ghi dữ liệu vào file */
   Employee myEmployee;
   for(int i=0; i<recordNumber; i++) {
      cout << “Mau tin thu ” << i+1 << endl;
      cout << “Name: ”;
      cin >> myEmployee.name; // Nhập tên nhân viên
      cout << “Day of birth: ”;
      cin >> myEmployee.birthDay.day;// Nhập ngày sinh
      cout << “Month of birth: ”;
      cin >> myEmployee.birthDay.month;// Nhập tháng sinh
      cout << “Year of birth: ”;
      cin >> myEmployee.birthDay.year;// Nhập năm sinh
      cout << “Role: ”;
      cin >> myEmployee.role; // Nhập chức vụ
      cout << “Salary: ”;
      cin >> myEmployee.salary; // Nhập tiền lương
```

6.3 Các thao tác truy xuất tập tin

```
    // Ghi dữ liệu vào file
    fileOut.write(reinterpret_cast <char *>(&myEmployee), sizeof(Employee));
    }
    /* Đóng file */
    fileOut.close(); // Đóng file
    return;
}
```

6.3 Các thao tác truy xuất tập tin

Ví dụ 5: Minh hoạ việc đọc dữ liệu từ tập tin vào biến có cấu trúc:

- *Tên tập tin được người dùng tự nhập vào từ bàn phím.*
- *Chương trình sẽ đọc các cấu trúc nhân viên trong tập tin và hiển thịra màn hình.*
- *Chương trình dừng lại khi kết thúc tập tin.*

```
#include <iostream>
#include <fstream>
const int length = 25; // Độ dài tối đa tên file
using namespace std;
struct Date {
        int day; // Ngày
        int month; // Tháng
        int year; // Năm
};
```

6.3 Các thao tác truy xuất tập tin

```
struct Employee{
        char name[20]; // Tên nhân viên
        Date birthDay; // Ngày sinh của nhân viên
        char role[20]; // Chức vụ của nhân viên
        float salary; // Lương của nhân viên
};
int main() {
    char fileName[length];
    cout << “Ten tap tin: ”; cin.get(fileName, length); // Nhập tên file
// Khai báo và mở file
    fstream fileIn(fileName, ios::in|ios::binary);
    if(!fileIn) { // Không mở được file
        cout << “Khong the tao duoc tap tin ” << fileName << endl;
        exit(1);
    }
```

6.3 Các thao tác truy xuất tập tin

```
/* Đọc dữ liệu từ file ra màn hình */
   Employee myEmployee;
   while(fileIn){
      fileIn.read(reinterpret_cast<char *>(&myEmployee), sizeof(Employee));
       // Đọc kí tự từ file
       cout << myEmployee.name << " " << myEmployee.birthDay.day << "/"
            << myEmployee.birthDay.month << "/"  <<
            myEmployee.birthDay.year << " "  << myEmployee.role << " "  <<
            myEmployee.salary << endl; // Ghi ra màn hình
   }
   /* Đóng file */
   fileIn.close(); // Đóng file
   return;
}
```

Truy xuất trực tiếp

- Truy nhập file trực tiếp
  - Con trỏ file: trỏ vào một vị trí xác định của file.
  - C++ cho phép truy nhập trực tiếp đến một vị trí xác định trên file bằng các phép toán:
    - Truy nhập vị trí hiện tại của con trỏ file
    - Dịch chuyển con trỏ file đến một vị trí xác định
- Truy nhập vị trí hiện tại của con trỏ file
  - Nếu biến file là kiểu mở file để đọc ifstream thì cú pháp là:

    Tên biến file.**tellg()**;
  - Nếu biến file là kiểu mở file để ghi ofstream thì cú pháp là:

    Tên biến file.**tellp()**;

6.3 Các thao tác truy xuất tập tin

Ví dụ 6a: Minh hoạ việc xác định vị trí hiện thời của con trỏ file sau một số thao tác đọc file trước đó.

```
#include <iostream>
#include <fstream>
const int length = 25; // Độ dài tối đa tên file
using namespace std;
int main(){
    char fileName[length], output;
    cout << “Ten tap tin: ”;
    cin.get(fileName, length); // Nhập tên file
    /* Mở file */
    ifstream fileIn(fileName, ios::in); // Khai báo và mở file
    if(!fileIn){ // Không mở được file
        cout << “Khong the mo duoc tap tin ” << fileName << endl;
        exit(1);
    }
```

6.3 Các thao tác truy xuất tập tin

```
/* Đọc dữ liệu từ file ra màn hình
   Ghi vi trí con trỏ file ra màn hình cứ sau 5 lần đọc kí tự */
   int index = 0;
   while(fileIn){
      fileIn >> output; // Đọc kí tự từ f ile
      cout << output; // Ghi kí tự ra màn hình
      if(index % 5 == 0) // Ghi ra vị trí con trỏ file
         cout<< endl << "Vi tri con tro tap tin: " << fileIn.tellg() << endl;
         index ++;
   }
   cout << endl; // Xuống dòng trên màn hình
   /* Đóng file */
   fileIn.close(); // Đóng file
   return;
}
```

6.3 Các thao tác truy xuất tập tin

Ví dụ 6b: Minh hoạ Minh hoạ việc xác định vị trí hiện thời của con trỏ file sau một số thao tác ghi vào file trước đó .

```
#include <iostream>
#include <fstream>
const int length = 25; // Độ dài tối đa tên file
using namespace std;
int main(){
    char fileName[length], input;
    cout << “Ten tap tin: ”;
    cin.get(fileName, length); // Nhập tên file
    /* Mở file */
    ofstream fileOut(fileName, ios::out);// Khai báo và mở file
    if(!fileOut){ // Không mở được file
        cout << “Khong the tao duoc tap tin ” << fileName << endl;
        exit(1);
    }
```

6.3 Các thao tác truy xuất tập tin

```
/* Ghi dữ liệu vào file
   Hiện ra màn hình vị trí con trỏ file sau khi ghi đuọc 5 kí tự */
   int index = 0;
   do {
      cin >> input; // Đọc kí tự từ bàn phím
      fileOut << input << ‘ ‘; // Ghi kí tự vào file
      if(index%5 == 0) // Hiển thị vị trí con trỏ file
         cout << ”Vi tri con tro file: ” << fileOut.tellp() << endl;
      index++;
   }  while((input != ‘e’)&&(fileOut));
   fileOut << endl; // Xuống dòng cuối file
   /* Đóng file */
   fileOut.close(); // Đóng file
   return;
}
```

Con trỏ tập tin

Dịch chuyển con trỏ file

- Nếu biến file có kiểu là mở file để đọc ifstream:

  Tên biến file.**seekg**(*Kích thước, Mốc dịch chuyển*);

- Nếu biến file có kiểu là mở file để ghi ofstream:

  Tên biến file.**seekp**(*Kích thước, Mốc dịch chuyển*);

Trong đó:

- Kích thước: khoảng cách dịch chuyển so với vị trí mốc dịch chuyển. Đơn vị tính là byte, có kiểu là số nguyên.
- Mốc dịch chuyển:

  ios::beg: Dịch chuyển từ đầu file.

  ios::cur: Dịch chuyển từ vị trí hiện thời của con trỏ file.

  ios::end: Dịch chuyển từ vị trí cuối cùng của file.

6.3 Các thao tác truy xuất tập tin

❖ **Ví dụ:**

Dịch chuyển con trỏ file đến kí tự (kiểu char) thứ 7+1 = 8 trong file tin abc.txt để đọc (giả sử file abc.txt lưu các kí tự kiểu char).

```
ifstream fileIn("abc.txt");
fileIn.seekg(sizeof(char)*7, ios::beg);
```

❖ **Chú ý:**

- Khoảng cách cách dịch chuyển có thể nhận giá trị âm hoặc dương.
- Nếu vị trí dịch chuyển đến nằm ngoài phạm vi file sẽ nảy sinh lỗi, khi đó Tên biến file == false.

6.3 Các thao tác truy xuất tập tin

Ví dụ 7: Cài đặt chương trình truy nhập tập tin trực tiếp để đọc giá trị kí tự (kiểu char) trong file:

- *Tên tập tin (chứa dữ liệu kiểu char) do người dùng nhập vào từ bàn phím.*
- *Sau đó, mỗi khi người dùng nhập vào một số nguyên, chương trình sẽ dịch chuyển đến vị trí mới, cách vị trí cũ đúng bằng từng ấy kí tự, tính từ vị trí hiện thời của con trỏ fike.*
- *Chương trình sẽ kết thúc khi người dùng nhập vào số 0.*

.

```
#include <iostream>
#include <fstream>
const int length = 25; // Độ dài tối đa tên file
using namespace std;
```

6.3 Các thao tác truy xuất tập tin

```
int main(){
    char fileName[length], output;
    cout << “Ten tap tin: ”;
    cin.get(fileName, length); // Nhập tên file
    /* Mở file */
    ifstream fileIn(fileName); // Khai báo và mở file
    if(!fileIn) {   // Không mở được file
        cout << “Khong the mo duoc tap tin ” << fileName << endl;
        exit(1);
    }
    /* Đọc dữ liệu từ file ra màn hình
        Ghi vi trí con trỏ file ra màn hình cứ sau 5 lần đọc kí tự*/
    int index = 1
```

6.3 Các thao tác truy xuất tập tin

```
do {
        cout << “So ki tu dich chuyen: ”;
        cin >> index;
        // Dịch chuyển con trỏ file từ vị trí hiện thời
        fileIn.seekg(sizeof(char)*index, ios::cur);
        if(fileIn){ // Đúng
            fileIn >> output; // Đọc kí tự từ file
            // Ghi kí tự ra màn hình
            cout << “Vi tri: ” << fileIn.tellg() << output;
        } else // Ra khỏi phạm vi file
            fileIn.clear(); // Về vị trí đầu file
    } while(index);
    /* Đóng file */
    fileIn.close(); // Đóng file
return;
}
```

6.4 Các hàm tập tin

Một số hàm xử lý chung

Phương thức	Chức năng
open	Mở file
close	Đóng file
is_open	Kiểm tra xem file có đang mở hay không.
flush	Làm sạch vùng đệm của file đang mở
good	Kiểm tra trạng thái của file có tốt để đọc/ghi không
eof	Kiểm tra xem đã đọc đến cuối file chưa

6.4 Các hàm tập tin

Một số hàm xử lý file văn bản

Phương thức	Chức năng
operator >>	Đọc dữ liệu từ file
operator <<	Ghi dữ liệu vào file
getline	Đọc một dòng dữ liệu từ file
get	Đọc một kí tự hoặc một chuỗi từ file, tùy thuộc vào tham số của phương thức.
put	Ghi một kí tự xuống file

6.4 Các hàm tập tin

Một số hàm xử lý file nhị phân

Phương thức	Chức năng
read	Đọc mẫu tin (theo cấu trúc định trước) từ file
write	Ghi mẫu tin (theo cấu trúc định trước) vào file
seekg	Di chuyển con trỏ file đến vị trí xác định
tellp	Cho biết vị trí hiện tại của con trỏ file

6.4 Các hàm tập tin

A. Hàm get() - Đọc một kí tự bất kì từ file.

Ví dụ sau sẽ xuất ra màn hình nội dung của tệp Demo1.txt

```
#include<fstream>
#include<iostream>
using namespace std;
int main()
{
	ifstream dataFile;
	char data;
	dataFile.open("Demo1.txt");
	while(dataFile.get(data))
		cout<<data;
	dataFile.close();
}
```

6.4 Các hàm tập tin

B. Hàm put() - Ghi một kí tự vào file.

Ví dụ sau sẽ ghi nội dung 1 câu văn vào file Vidu.txt

```
#include<fstream>
#include<iostream>
using namespace std;
int main()
{
        char ch;
        fstream dataFile("Vidu.txt", ios::out);
        cout << "Nhap noi dung muon ghi va ket thuc bang dau cham :\n";
        cin.get(ch);
        while (ch != '.')
        {       dataFile.put(ch);       cin.get(ch);    }
        dataFile.put(ch); //Ghi dấu chấm cuối câu.
        dataFile.close();
}
```

6.4 Các hàm tập tin

C. Hàm write() - Ghi file ở dạng nhị phân.

fileObject.write(address, size);

- *fileObject - Đối tượng file.*
- *address – Địa chỉ đầu tiên của 1 vùng nhớ được ghi vào file.*
- *size – Số lượng byte của vùng nhớ mà nó sẽ được ghi.*

Ví dụ 1:

```
char kitu = 'A';
file.write(&kitu, sizeof(kitu));
```

Ví dụ 2:

```
char Data[] = {'A',  'B', 'C', 'D'};
file.write(Data, sizeof(Data));
```

6.4 Các hàm tập tin

D. Hàm read() - Đọc dữ liệu nhị phân từ file vào bộ nhớ.

fileObject.read(address, size);

- *fileObject - Đối tượng file.*
- *address – Địa chỉ đầu tiên mà vùng nhớ mà dữ liệu được đọc vào được lưu.*
- *size – Số lượng byte trong bộ nhớ được đọc vào từ file.*

Ví dụ:

```
char kitu = 'A';
file.read(&kitu, sizeof(kitu));
```

```
char Data[5]
file.read(Data, sizeof(Data));
```

E. Hàm xóa và đổi tên file :

remove (“đường dẫn tập tin”);

rename (“ tên tập tin cũ “, “ tên tập tin mới “);

Ví dụ sử dụng hàm write() và hàm read()

```
#include <iostream>
#include <fstream>
using namespace std;
int main()
{
        const int SIZE = 4;
        char data[SIZE] = {'A', 'B', 'C', 'D'}, data2[SIZE];
        fstream file;
        file.open("VD.dat", ios::out | ios::binary);
        file.write(data, sizeof(data));
        file.close();
        file.open("VD.dat", ios::in | ios::binary);
        file.read(data2, sizeof(data2));
        for (int count = 0; count < SIZE; count++)
                cout << data2[count] << " ";
        cout << endl; file.close();
}
```

6.4 Các hàm tập tin

F. Các hàm di chuyển con trỏ file

- Hàm seekp(n) - Di chuyển con trỏ đến byte thứ n (các byte được tính từ 0).
- Hàm seekp(n, vị trí xuất phát) - Di chuyển đi n byte (có thể âm hoặc dương) từ vị trí xuất phát. Vị trí xuất phát gồm:
  - ios::beg : Từ đầu file
  - ios::end : Từ cuối file
  - ios::cur : Từ vị trí hiện tại của con trỏ.
- Hàm tellp(n) - Cho biết vị trí hiện tại của con trỏ.
- Để làm việc với dòng nhập tên các phương thức trên được thay tương ứng bởi các tên : seekg và tellg

G. Hàm gcount() - Cho biết số kí tự hàm read đọc được

…

cstdio

(đọc thêm)

cstdio (stdio.h)

- **Khai báo biến file:**

  FILE danh sách các biến con trỏ;

- **Mở file:**

  Tên biến file = fopen(“path”,”chế độ mở”);

Chế độ mở:

- "r" - *Mở một file có sẵn để đọc.*
- "w" - *Tạo file mới để ghi, nếu file có sẵn thì sẽ bị ghi mới hoàn toàn.*
- "a" - *Mở một file để ghi từ vị trí cuối cùng của file, nếu file không tồn tại sẽ tạo mới.*
- "r+" - *Mở một file có sẵn để đọc và ghi.*
- "w+" - *Tạo file mới để đọc và ghi, nếu file có sẵn thì sẽ bị ghi mới hoàn toàn.*
- "a+" - *Mở một file để để đọc và ghi. Có thể đọc từ đầu file, nhưng khi ghi thì ghi từ vị trí cuối cùng của file, nếu file không tồn tại sẽ tạo mới.*

cstdio (stdio.h)

Một số hàm xử lý chung:

Tên hàm	Chức năng
fopen	Mở tập tin
fclose	Đóng file
fcloseall	Đóng tất cả file đang mở
fflush	Làm sạch vùng đệm của một tập tin đang mở
fflushall	Làm sạch vùng đệm của tất cả
remove/unlink	Xóa file
feof	Kiểm tra đã đến cuối file chưa

cstdio (stdio.h)

Một số hàm xử lý tập tin văn bản:

Tên hàm	Chức năng
fprintf	Ghi giá trị dạng text lên tập tin
fscanf	Đọc giá trị dạng text từ tập tin
putc/fputc	Ghi lên tập tin một kí tự (sử dụng tập tin văn bản hoặc nhị phân có khác biệt)
getc/fgetc	Đọc từ tập tin một kí tự (sử dụng tập tin văn bản hoặc nhị phân có khác biệt)
fputs	Ghi một chuỗi vào tập tin
fgets	Đọc một chuỗi từ tập tin

cstdio (stdio.h)

- **Một số hàm xử lý tập tin nhị phân:**

Tên hàm	Chức năng
putw	Ghi một số nguyên lên tập tin
getw	Đọc một số nguyên từ tập tin
fwrite	Ghi các mẫu tin (có cấu trúc định trước) lên tập tin
fread	Đọc các mẫu tin (có cấu trúc định trước) từ tập tin
fseek	Di chuyển con trỏ đến vị trí mong muốn
ftell	Cho biết vị trí hiện tại của con trỏ

cstdio (stdio.h)

Ví dụ:

- Đọc dữ liệu từ file :

  fscanf(f, "%d", &x); /*Đọc dữ liệu dạng số nguyên từ file cho vào biến x */

  char s[80]; fgets(s, 80, f); /*Đưa vào chuỗi s tối đa 80 kí tự lấy từ file */

  char c=getc(f); //Đọc 1 kí tự từ file lưu vào biến c

- Ghi dữ liệu vào file:

  fprintf(f,"%d", x); // Xuất giá trị nguyên x ra file

  fputs("Ky Thuat Lap Trinh", f); //Xuất chuỗi ra file

  putc('c', f); // Xuất ký tự 'c' ra file

cstdio (stdio.h)

Ví dụ 1: Mở và ghi dữ liệu lên file văn bản.

```
#include <iostream>
#include <cstdio>
using namespace std;
int main() {
        FILE *data;
        data = fopen(“myText.txt","w");
        if (data==NULL)
            cout<< “Can not open file!”<< endl;
        else
            fprintf(data, “This is my text !");
        fclose(data);
}
```

cstdio (stdio.h)

Ví dụ 2: Mở và đọc dữ liệu từ file văn bản.

```
#include <iostream>
#include <cstdio>
using namespace std;
int main() {
        FILE *data; char dt[1000];
        data = fopen(“myText.txt","r");
        if (data==NULL)
            cout << “Can not open file!”<< endl;
         else
            while(fscanf(data, "%s", dt)!=EOF)
                    cout << dt;
        fclose(data);
}
```

cstdio (stdio.h)

Ví dụ 3: Mở và ghi các record vào file nhị phân.

```
#include <cstdio>
#include <iostream>
using spacename std;
typedef struct { char name[20]; char birthDay[9]; float Salary;} Employee;
void CreateEmpLst(char *fName){
        Employee lstEmp[4]=
                {{“Smith”, “1-Jan-80”,2000},{“Bean”,”30-Dec-70”,5000},
                 {“Mike”, “12-Apr-71”,1500}, “Bear”, “10-Fre-75”,4500}};
        FILE *fData;
        fData=fopen(fName,”wb”);
        if (fData !=NULL) { fwrite(lstEmp, sizeof(Employee),4,fData);
                            fclose(fData); }
        else cout<<“Can not create File!!”<< endl;
int main() {
        CreatEmpLst(“Employes.dat”); return 0;
}
```

cstdio (stdio.h)

Ví dụ 4: Mở và thêm các record vào file nhị phân.

```
void AppendEmp (char *fName){
        Employee emp;
        FILE *fData;
        cout << "Append new Employee in to " << fName << endl;
        cout << "Name: "; cin >> emp.Name;
        cout << "Birthday: "; cin >> emp.birthDay;
        cout << "Salary: "; cin >> emp.Salary;

        fData=fopen(fName,"ab");
        if (fData !=NULL) {
                fwrite(&emp, sizeof(Employee),1,fData);
                fclose(fData); }
        else cout<<"Can not open File!!"<< endl;
}
```

cstdio (stdio.h)

Ví dụ 5: Đọc danh sách các record từ file nhị phân.

```
void ReadEmpLst (char *fName){
        FILE *fData;
        Employee emp;
        int i;
        fData=fopen(fName,"rb");
        if (fData !=NULL) {
           while (!feof(fData)) {
                  i++;
                  if (fread(&emp, sizeof(Employee),1,fData)==1)
                     cout<<"Name: "<<emp.Name<<" Birthday: "<< emp.birthDay
<< "Sarary: " <<  emp.Salary <<endl;
           }
           fclose(fData);
        }
        else cout<<"Can not open File!!"<< endl;
}
```


Thank You !