Xử lý tệp (file)

Giới thiệu

- Lưu trữ dữ liệu
  - Mảng, biến là tạm thời
  - Files là lâu dài
    - Đĩa từ, đĩa quang

Nội dung bài

Tạo, cập nhât, xử lý file

Truy cập tuần tự, truy cập ngẫu nhiên

Xử lý file có định dạng, file nhị phân

Phân cấp dữ liệu

Từ nhỏ nhất tới lớn nhất

Bit

- 1 hoặc 0
- Mọi thứ trong máy tính biểu diễn bằng bits
- Khó hiểu với con người
- Tập ký tự
  - số, chữ cái, ký hiệu để biểu diễn dữ liệu
  - Ký tự được biểu diễn bằng chuỗi bit

Byte: 8 bits

- có thể lưu ký tự (**`char`**)

Phân cấp dữ liệu

Từ nhỏ nhất tới lớn nhất (tiếp)

Trường (Field): nhóm ký tự có ý nghĩa

- Ví dụ: tên

Bản ghi (Record): nhóm trường có liên quan

- **`struct`** hoặc **`class`** trong C++
- Hệ thống quản lý nhân sự: mã, tên, địa chỉ, lương
- Khoá: trường dùng để định danh bản ghi (mỗi bản ghi có 1 khoá khác nhau)

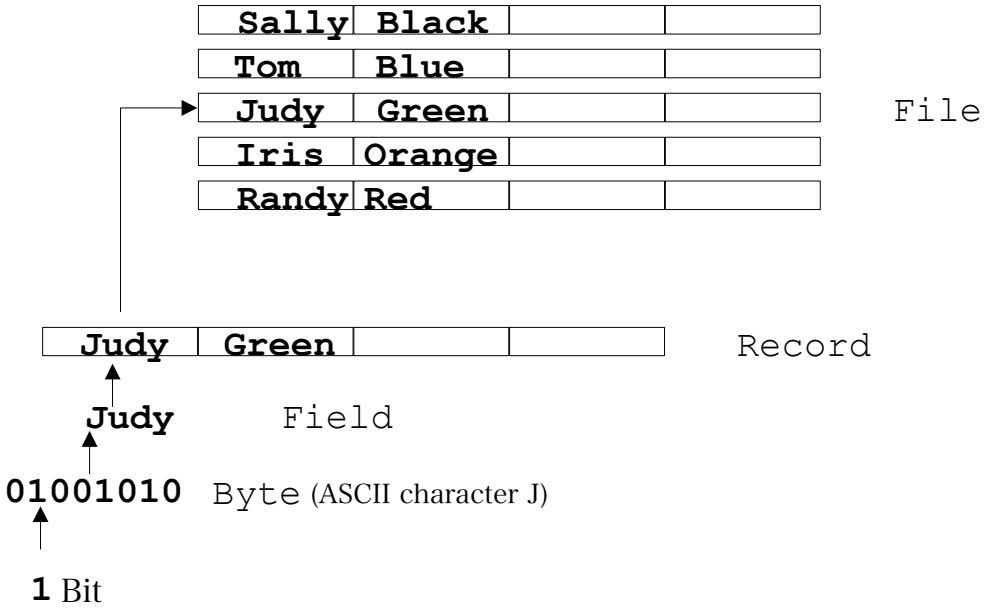
File: nhóm bản ghi có liên quan

- Danh sách nhân viên của cả công ty

Database: Nhóm file có liên quan

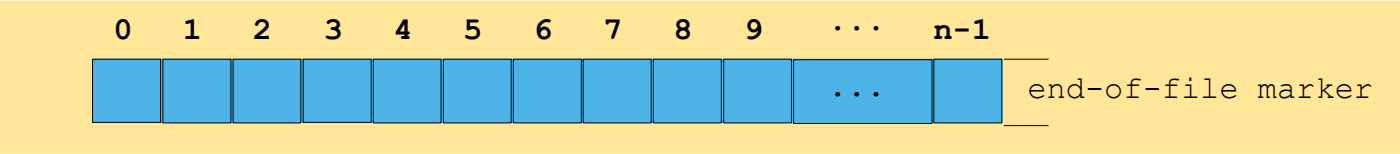
- Nhân viên, Lương, Chấm công…

Phân cấp dữ liệu



Files và Luồng (Streams)

- C++ xử lý file bằng chuỗi bytes
  - Kết thúc bằng ký hiệu *end-of-file*



Trong C++, khi thao tác với một file dữ liệu, cần thực hiện tuần tự theo các bước như sau:

- Mở (hoặc tạo mới) file
- Thực hiện các thao tác đọc, ghi trên file đang mở
- Đóng file

Cần: #include<fstream.h>

Khai báo biến file

Cú pháp: fstream <Tên biến file>(<Tên file>, <Chế độ mở file>);

Trong đó:

Tên biến file: tuân thủ theo quy tắc đặt tên biến trong C++.

Tên file: là tên file dữ liệu mà ta cần thao tác trên nó.

Chế độ mở file: chỉ ra rằng ta đang mở file tin ở chế độ nào: đọc hoặc ghi, hoặc cả đọc lẫn ghi.

Ví dụ: `fstream myFile("abc.txt", ios::in);`

Lưu ý:

Tên file có đường dẫn thư mục "\" phải được viết thành "\\"

Ví dụ: `fstream myFile("myDir\\abc.txt", ios::in);`

Tạo một truy cập tuần tự tới file

- C++ không dùng cấu trúc cho file
  - Khái niệm “bản ghi” phải do lập trình viên tự viết
- Để mở file, tạo đối tượng
  - Tạo "đường liên lạc" từ đối tượng tới file
  - Classes
    - **`ifstream`** (chỉ vào)
    - **`ofstream`** (chỉ ra)
    - **`fstream`** (vào/ra - I/O)
  -

Tạo một truy cập tuần tự tới file

- Chế độ mở file

Mode	Description
ios::app	Viết tiếp output vào cuối file.
ios::ate	Mở một file để ghi và di chuyển đến cuối file (thường dùng để nối dữ liệu vào file). Dữ liệu có thể được viết vào vị trí tùy ý trong file.
ios::in	Mở file để đọc
ios::out	Mở file để ghi.
ios::trunc	Loại bỏ nội dung file nếu nó tồn tại (mặc định đối với **ios::out**)
ios::binary	Mở file nhị phân (i.e., không phải file text) để đọc hoặc ghi.

  - **ofstream** mặc định là mở để ghi
    - **ofstream ghi_file("clients.dat", ios::out);**
    - **ofstream ghi_file("clients.dat");**

Tạo một truy cập tuần tự tới file

- Toán tử
  - Nạp chồng toán tử **!**
    - **!ghi_file**
    - Trả về nonzero (true) nếu **badbit** hoặc **failbit** bật
      - Mở file không tồn tại hoặc không được quyền mở
  - **Toán tử void***
    - Chuyển đối tượng dòng thành con trỏ
    - **0** khi **failbit** hoặc **badbit** bật, ngược lại nonzero
      - **failbit** bật khi gặp EOF
    - **while (cin >> myVariable)**
      - Ngầm chuyển đổi **cin** thành pointer
      - Lặp tới khi EOF

Tạo một truy cập tuần tự tới file

- Hoạt động
  - Ghi vào file (giống **`cout`**)
    - **`ghi_file << myVariable`**
  - Đóng file
    - **`ghi_file.close()`**
    - Tự đóng khi hàm huỷ được gọi

fig14_04.cpp

Đọc dữ liệu tuần tự

- Đọc files
  - `ifstream doc_file( "filename", ios::in );`
  - Overloaded `!`
    - `!doc_file` kiểm tra file đã được mở đúng hay chưa
  - `operator void*` chuyển đổi sang con trỏ
    - `while (doc_file >> myVariable)`
    - Dừng khi gặp EOF (nhận giá trị `0`)

fig14_07.cpp

Đọc dữ liệu tuần tự

con trỏ vị trí ghi số thứ tự của byte tiếp theo để đọc/ghi

các hàm đặt lại vị trí của con trỏ:

- **`seekg`** (đặt vị trí đọc cho **`istream`**)
- **`seekp`** (đặt vị trí ghi cho **`ostream`**)
- **`seekg`** và **`seekp`** lấy các đối số là *offset* và *mốc*
  - Offset: số byte tương đối kể từ mốc
  - Mốc (**`ios::beg`** mặc định)
    - **`ios::beg`** - đầu file
    - **`ios::cur`** - vị trí hiện tại
    - **`ios::end`** - cuối file

các hàm lấy vị trí hiện tại của con trỏ:

- **`tellg`** và **`tellp`**

Đọc dữ liệu tuần tự

- Ví dụ

`doc_file.seekg(0)`: Đến đầu file (vị trí **`0`**) , mặc định đối số thứ 2 là **`ios::beg`**

`doc_file.seekg(n)`: Tới byte thứ n kể từ đầu file

`doc_file.seekg(n, ios::cur)`: Tới byte thứ n kể từ vị trí hiện tại

`doc_file.seekg(y, ios::end)`: quay ngược y byte kể từ cuối file

`doc_file.seekg(0, ios::end)`: Tới byte cuối

`seekp` tương tự

Đọc dữ liệu tuần tự

- Để tìm vị trí con trỏ
  - **`tellg`** và **`tellp`**
  - **`location = doc_file.tellg()`**
- Case study
  - Tạo chương trình quản lý thẻ tín dụng
  - Liệt kê tài khoản rỗng, tài khoản âm, tài khoản dư

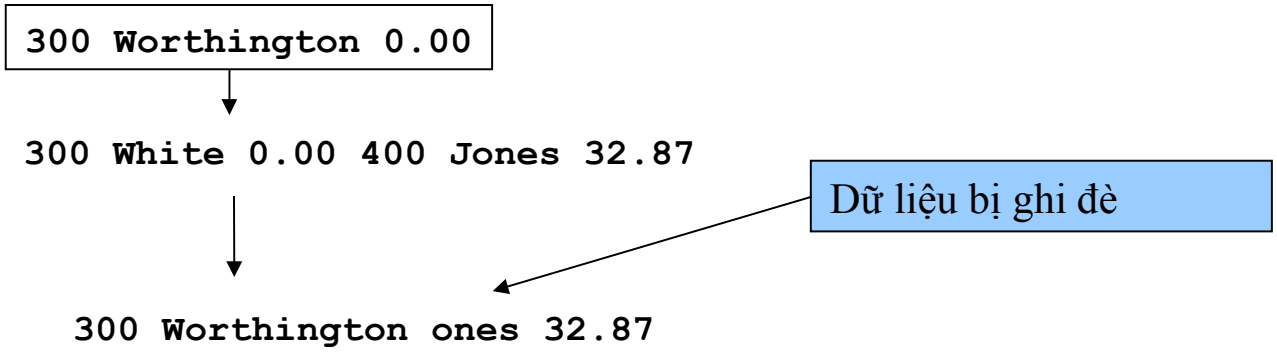
fig14_08.cpp

Cập nhật file tuần tự

- Cập nhật file tuần tự
  - Rủi ro: ghi đè nên dữ liệu khác
  - Ví dụ: đổi tên "White" thành "Worthington"
    - Dữ liệu cũ

```
300 White 0.00 400 Jones 32.87
```

    - Chèn dữ liệu mới



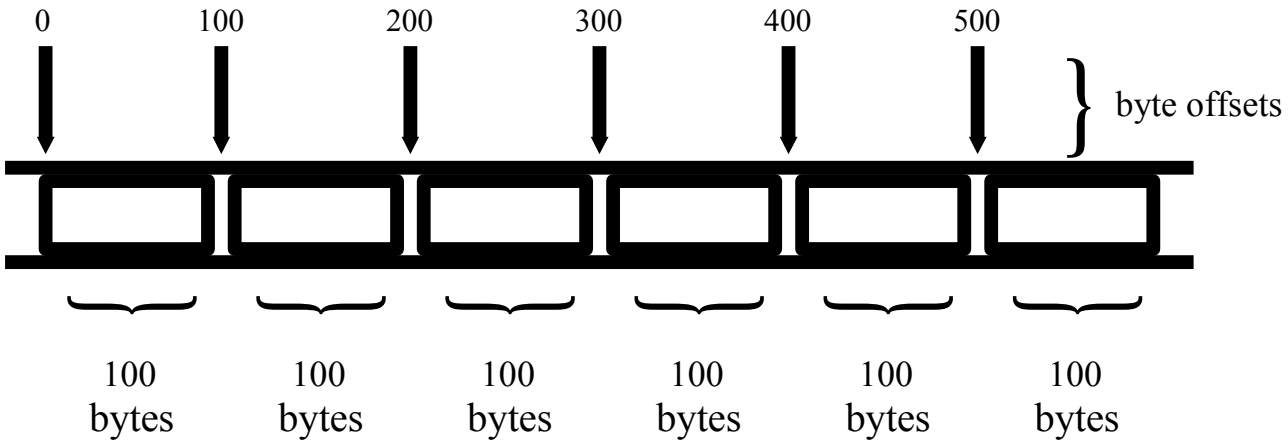
  - Định dạng văn bản khác với biểu diễn cũ.

Truy cập file ngẫu nhiên

- Truy cập tức thời
  - Muốn định vị nhanh bản ghi
    - Đặt vé máy bay, ATMs
  - Các file tuần tự phải duyệt qua từng bản ghi
- Truy cập file ngẫu nhiên là giải pháp cho:
  - Truy cập tức thời
  - Chèn dữ liệu mà không làm hỏng dữ liệu khác
  - Cập nhật/xoá dữ liệu mà không ảnh hưởng tới dữ liệu khác

Truy cập file ngẫu nhiên

- C++ không dùng cấu trúc trên files
  - Lập trình viên phải tạo file truy cập ngẫu nhiên
  - Cách đơn giản nhất; bản ghi kích thước cố định
    - Tính toán vị trí của file từ bản ghi và khoá



Tạo một truy cập file ngẫu nhiên

- **"1234567" (char *)** khác **1234567 (int)**
  - **char *** lấy 8 bytes (1 cho mỗi kí tự + null)
  - **int** lấy >1 bytes (4 bytes)
    - 123 có cùng kích thước với 1234567
- **Toán tử <<** và **write()**
  - **outFile << number**
    - Ghi **number** (**int**) như **char ***
    - Số byte khác nhau
  - **outFile.write(*const char **, *size*);**
    - Ghi ra byte dạng thô (raw)
    - Lấy tham số là con trỏ trỏ tới vị trí bộ nhớ và số byte cần ghi
      - Sao chép dữ liệu trực tiếp từ bộ nhớ sang file
      - Không chuyển sang kiểu **char ***

Tạo một truy cập file ngẫu nhiên

- Ví dụ

```
outFile.write( reinterpret_cast<const char *>(&number),
   sizeof( number ) );
```

  - **`&number`** là một **`int *`**
    - đổi thành **`const char *`** với **`reinterpret_cast`**
  - **`sizeof(number)`**
    - Kích thước của **`number`** (một số **`int`**) đơn vị là bytes
  - **`read`** function tương tự (trình bày sau)
  - Chỉ dùng **`write`**/**`read`** giữa các máy tương thích
    - Chỉ khi dùng dữ liệu thô, không định dạng
  - Dùng **`ios::binary`** cho đọc/ghi thô

Tạo một truy cập file ngẫu nhiên

- Thường ghi dữ liệu có cấu trúc vào file
- Vấn đề
  - Chương trình xử lý tín dụng
  - Lưu nhiều nhất 100 bản ghi kích thước cố định
  - Bản ghi
    - Số tài khoản (key)
    - Họ và tên
    - Số tiền
  - Thao tác với tài khoản
    - Cập nhật, tạo mới, xoá, liệt kê
- Tiếp: chương trình để tạo file có 100 bản ghi trắng

fig14_12.cpp

Ghi dữ liệu vào file truy cập ngẫu nhiên

- Dùng **`seekp`** để ghi vào vị trí chính xác trong file
  - Bản ghi đầu tiên bắt đầu ở đâu?
    - Byte 0
  - Bản ghi thứ 2?
    - Byte 0 + sizeof(object)
  - Bản ghi bất kì?
    - (RecordNum - 1) * sizeof(object)

fig14_13.cpp

Đọc dữ liệu vào file truy cập ngẫu nhiên

- **`read`** - tương tự **`write`**
  - Đọc file nhị phân từ file vào bộ nhớ
  - ```
    inFile.read( reinterpret_cast<char *>( &number ),
                 sizeof( int ) );
    ```
    - **`&number`**: ví trí lưu dữ liệu
    - **`sizeof(int)`**: số byte sẽ đọc
  - KHÔNG DÙNG **`inFile >> number`** với dữ liệu nhị phân
    - **`>>`** dùng cho **`char *`**

fig14_14.cpp
    ```

Bài tập

- Viết chương trình (theo cả 2 kiểu file có cấu trúc và file nhị phân):
  - ghi ra file "nhanvien.txt" thông tin của n nhân viên (tên, tuổi, mã Nhân viên, lương) được nhập từ bàn phím.
  - đọc danh sách nhân viên từ file "nhanvien.txt" và ghi ra màn hình
  - sửa lương nhân viên có mã 102 thành 4.000.000
  - xoá nhân viên có tuổi <=18 khỏi file
  - Thêm 1 nhân viên ten = "Nguyen Van A", tuoi = 19, mã = 103, lương = 2.000.000

fig14_14.cpp