Nhập Môn Lập Trình
Luồng Dữ Liệu

TS. Lê Nguyên Khôi
Trường Đại học Công nghệ, ĐHQGHN

Nội Dung

- Luồng (stream)
  - Tệp Nhập/Xuất
  - Ký Tự Nhập/Xuất
- Công cụ cho Luồng
  - Sử dụng tên tệp cho dữ liệu vào
  - Định dạng dữ liệu ra, cài đặt cờ
- Tệp truy cập ngẫu nhiên

Giới Thiệu

- Luồng dữ liệu (stream)
  - Đối tượng đặc biệt
  - Dùng nhập/xuất dữ liệu của chương trình
- Tệp Nhập/Xuất dùng:
  - Nhập dữ liệu từ tệp vào chương trình
  - Xuất dữ liệu từ chương trình ra tệp

Luồng Dữ Liệu

- Chuỗi các ký tự
- Luồng vào (input stream)
  - Nhập vào chương trình
    - Từ bàn phím
    - Từ tệp
- Luồng ra (output stream)
  - Xuất ra khỏi chương trình
    - Ra màn hình
    - Ra tệp

Sử Dụng Luồng Dữ Liệu

- Chúng ta đã sử dụng luồng
  - **`cin`**
    - Đối tượng luồng nhập kết nối với bàn phím
  - **`cout`**
    - Đối tượng luồng xuất kết nối với màn hình
- Định nghĩa các luồng khác
  - Để nhập/xuất cho tệp
  - Sử dụng tương tự **`cin`**, **`cout`**

Tệp Văn Bản

- Đọc từ tệp
  - Khi nhập dữ liệu cho chương trình
- Ghi vào tệp
  - Khi chương trình xuất dữ liệu
- Bắt đầu từ đầu đến cuối tệp
  - Có nhiều cách thức (đọc/ghi) khác
  - Truy cập tệp văn bản đơn giản

Kết Nối Tệp Với Chương Trình

- Phải kết nối tệp với đối tượng luồng
  - **`cin`**/**`cout`** luôn kết nối với bàn phím/màn hình
  - Phải xác định tệp để kết nối
- Với nhập dữ liệu:
  - Sử dụng kiểu **`ifstream`**
- Với xuất dữ liệu:
  - Sử dụng kiểu **`ofstream`**
- Các lớp **`ifstream`** và **`ofstream`**
  - Được định nghĩa trong thư viện **`fstream`**
  - Có trong không gian tên **`std`**

Thư Viện Tệp Nhập/Xuất

- Cho phép nhập/xuất dữ liệu từ/ra tệp:

```
#include <fstream>
using namespace std;
```

hoặc

```
#include <fstream>
using std::ifstream;
using std::ofstream;
```

Khai Báo Luồng

- Khai báo luồng giống như biến nào:

```
ifstream in_stream1, in_stream2;
ofstream out_stream1, out_stream2;
```

- Sau đó kết nối với tệp:

```
in_stream1.open("infile1.txt");
in_stream2.open("infile2.txt");
out_stream1.open("outfile1.txt");
out_stream2.open("outfile2.txt");
```

  - Thực hiện thao tác mở tệp
  - Sử dụng hàm thành viên **`open`**
  - Có thể chỉ định đường dẫn đầy đủ

Sử Dụng Luồng Dữ Liệu

- Sau khi khai báo sử dụng như **cin**

```
ifstream in_stream;
in_stream.open("infile.txt");
int so_thu_nhat, so_thu_hai;
in_stream >> so_thu_nhat;
in_stream >> so_thu_hai;
```

- Tương tự luồng xuất (giống **cout**)

```
ofstream out_stream;
out_stream.open("outfile.txt");
out_stream << " so1 = " << so_thu_nhat;
out_stream << " so2 = " << so_thu_hai;
```

  - Ghi dữ liệu ra tệp

Tên Tệp

- Đối với chương trình, tệp có 2 tên:
  - Tên tệp ngoại vi
    - Tên tệp trên ổ cứng, đôi khi được gọi tên tệp thực
    - Sử dụng một lần trong chương trình (khi mở tệp)
    - Ví dụ: **`infile.txt`**, **`outfile.txt`**
  - Tên luồng
    - Tên tệp (tên biến) trong chương trình
    - Sử dụng tên này cho tất cả các hoạt động đọc/ghi
    - Ví dụ: **`in_stream`**, **`out_stream`**

Đóng Tệp

- Tệp nên được đóng lại
  - Khi chương trình hoàn tất đọc/ghi dữ liệu
  - Đóng kết nối giữa luồng & tệp
  - Ví dụ:

```
in_stream.close();
out_stream.close();
```

    - Không có tham số
- Tệp tự động đóng khi kết thúc chương trình

Nhập/Xuất Dữ Liệu Sử Dụng Tệp

```
#include <iostream>
#include <fstream>
using namespace std;
int main {
    ifstream in_stream;
    ofstream out_stream;
    in_stream.open("infile.txt");
    out_stream.open("outfile.txt");
    int so1, so2, so3;
    in_stream >> so1 >> so2 >> so3;
    out_stream << "Tong 3 so dau la "
               << (so1 + so2 + so3) << endl;
    in_stream.close();
    out_stream.close();
    return 0;
}
```

Ghi Dữ Liệu Vào Tệp

- Thường một tệp được mở là một tệp rỗng
  - Nếu tệp tồn tại, toàn bộ dữ liệu bị xóa
- Mở tệp để chèn vào cuối tệp (sử dụng cờ):

```
ofstream out_stream;
out_stream.open("outfile.txt", ios::app);
```

  - Nếu tệp không tồn tại, tạo tệp mới
  - Nếu tệp tồn tại, chèn vào cuối tệp
  - Tham số thứ 2 là hằng trong lớp **`ios`**
    - Trong thư viện **`iostream`**
    - Không gian tên **`std`**

Kiểm Tra Mở Tệp Thành Công

- Có thể gặp lỗi khi mở tệp
  - Khi tệp không tồn tại (để đọc dữ liệu)
  - Không có quyền ghi vào tệp
- Sử dụng hàm thành viên **`fail()`**
  - Gọi hàm thành viên **`fail()`** để kiểm tra mở luồng thành công không

```
in_stream.open("infile.txt");
if (in_stream.fail()) {
    cout << "Loi mo file.\n";
    exit(1);
}
```

Kiểm Tra Kết Thúc Tệp

- Dùng vòng lặp để xử lý với tệp đến khi hết tệp
- Sử dụng hàm thành viên **`eof()`**

```
char next;
in_stream.get(next);
while (!in_stream.eof()) {
    cout << next;
    in_stream.get(next);
}
```

  - Đọc từng ký tự cho đến khi hết dữ liệu trong tệp
  - Hàm thành viên **`eof()`** trả về kiểu **`bool`**

Nhập Tên Tệp Khi Chạy Chương Trình

- Tham số cho **`open()`** là kiểu xâu ký tự
- Tên cụ thể hoặc biến

```
string ten_tep;
cout << "Nhap ten tep: ";
cin >> ten_tep;
ifstream in_stream;
in_stream.open(ten_tep);
```

- Linh hoạt hơn đối với tên tệp là biến được nhập từ bàn phím khi chạy chương trình

Định Dạng Dữ Liệu Xuất

- Xuất dữ liệu dưới dạng 2 số thập phân:

```
cout.setf(ios::fixed);
cout.setf(ios::showpoint);
cout.precision(2);
```

  - In ra màn hình (12.50)
  - Hàm thành viên **`precision(x)`**
    - Phần thập phân dưới dạng "x" số sau dấu “.”
  - Hàm thành viên **`setf()`**
    - Cho cài đặt nhiều định dạng
- Có thể sử dụng cho bất cứ luồng xuất nào
- Đối với luồng cho tệp, hàm thành viên giống đối tượng **`cout`**

Tham Khảo

- Đọc sách:
  - Chương 8, Lập Trình Cơ Bản C++
- Tìm hiểu thư viện **`iomanip`**
  - Các định dạng xuất dữ liệu
- Tìm hiểu thao tác với tệp truy cập ngẫu nhiên
  - Hiểu quả cho cơ sở dữ liệu lớn