LẬP TRÌNH NÂNG CAO

Bài 13+14+15: vào ra dữ liệu với tập tin

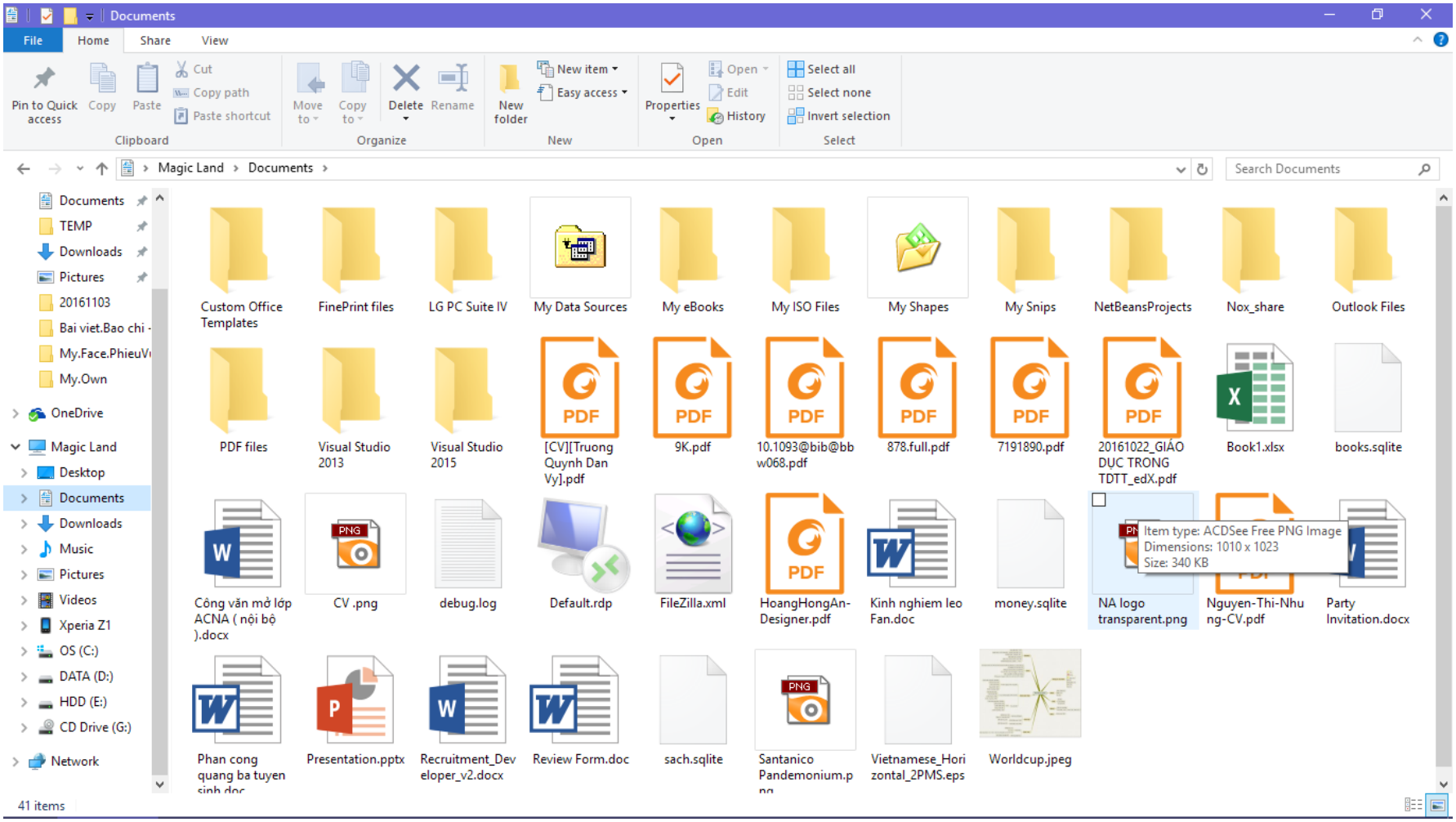
Nội dung

1. Tập tin văn bản và tập tin nhị phân
2. Làm việc với tập tin văn bản
3. Làm việc với tập tin nhị phân
4. Bài tập

Phần 1

Tập tin văn bản và tập tin nhị phân

Làm việc với tập tin



Làm việc với tập tin

- Tập tin (file) là thành phần cơ bản của các thiết bị lưu trữ
- Đa số các ngôn ngữ lập trình (trong đó có C/C++) chia tập tin làm 2 loại:
  - Tập tin dạng nhị phân (binary file): có thể xem như dãy các byte, đọc/ghi theo từng byte
  - Tập tin dạng văn bản (text file): có thể xem như dãy các string, đọc/ghi theo từng dòng
- Biến cin, cout thực chất là các tập tin văn bản đặc biệt
  - cin đại diện cho tập tin đầu vào của chương trình
  - cout đại diện cho tập tin đầu ra của chương trình
- Vì vậy: làm việc với file văn bản cũng tương tự làm việc với cin, cout

Tập tin văn bản

- Dãy các dòng kế tiếp nhau
- Độ dài các dòng không nhất thiết phải giống nhau
- Mỗi dòng kết thúc bằng ký hiệu cuối dòng (end_of_line) hoặc ký hiệu cuối tập tin (end_of_file) – nếu là dòng cuối cùng trong file
  - Dòng không phải là một chuỗi: chuỗi kết thúc bởi ký tự \0
- Khi ghi ký hiệu xuống dòng (\n), hệ thống tự động chuyển thành cặp ký tự CR-LF (về đầu dòng và xuống dòng) trên Windows và thành cặp LF-CR trên Linux/Unix
- Khi đọc thì cặp CR-LF hoặc LF-CR được tự động chuyển thành ký hiệu xuống dòng (\n)

Tập tin văn bản

```
00000000: 23 69 6E 63 6C 75 64 65|20 3C 69 6F 73 74 72 65  | #include <iostre
00000010: 61 6D 3E 0D 0A 75 73 69|6E 67 20 6E 61 6D 65 73  | am>..using names
00000020: 70 61 63 65 20 73 74 64|3B 0D 0A 0D 0A 69 6E 74  | pace std;....int
00000030: 20 6D 61 69 6E 28 29 20|7B 0D 0A 20 20 20 20 64  |  main() {..    d
00000040: 6F 75 62 6C 65 20 6E 3B|0D 0A 20 20 20 20 63 6F  | ouble n;..    co
00000050: 75 74 20 3C 3C 20 22 4E|20 3D 20 22 3B 0D 0A 20  | ut << "N = ";..
00000060: 20 20 20 63 69 6E 20 3E|3E 20 6E 3B 0D 0A 0D 0A  |    cin >> n;....
00000070: 20 20 20 20 64 6F 75 62|6C 65 20 78 20 3D 20 31  |     double x = 1
00000080: 3B 0D 0A 20 20 20 20 78|20 3D 20 28 78 20 2B 20  | ;..    x = (x +
00000090: 6E 2F 78 29 20 2F 20 32|3B 0D 0A 20 20 20 20 78  | n/x) / 2;..    x
000000A0: 20 3D 20 28 78 20 2B 20|6E 2F 78 29 20 2F 20 32  |  = (x + n/x) / 2
000000B0: 3B 0D 0A 20 20 20 20 78|20 3D 20 28 78 20 2B 20  | ;..    x = (x +
000000C0: 6E 2F 78 29 20 2F 20 32|3B 0D 0A 20 20 20 20 78  | n/x) / 2;..    x
000000D0: 20 3D 20 28 78 20 2B 20|6E 2F 78 29 20 2F 20 32  |  = (x + n/x) / 2
000000E0: 3B 0D 0A 20 20 20 20 78|20 3D 20 28 78 20 2B 20  | ;..    x = (x +
000000F0: 6E 2F 78 29 20 2F 20 32|3B 0D 0A 20 20 20 20 78  | n/x) / 2;..    x
00000100: 20 3D 20 28 78 20 2B 20|6E 2F 78 29 20 2F 20 32  |  = (x + n/x) / 2
00000110: 3B 0D 0A 20 20 20 20 78|20 3D 20 28 78 20 2B 20  | ;..    x = (x +
00000120: 6E 2F 78 29 20 2F 20 32|3B 0D 0A 20 20 20 20 78  | n/x) / 2;..    x
00000130: 20 3D 20 28 78 20 2B 20|6E 2F 78 29 20 2F 20 32  |  = (x + n/x) / 2
00000140: 3B 0D 0A 20 20 20 20 78|20 3D 20 28 78 20 2B 20  | ;..    x = (x +
00000150: 6E 2F 78 29 20 2F 20 32|3B 0D 0A 20 20 20 20 78  | n/x) / 2;..    x
00000160: 20 3D 20 28 78 20 2B 20|6E 2F 78 29 20 2F 20 32  |  = (x + n/x) / 2
00000170: 3B 0D 0A 0D 0A 20 20 20|20 63 6F 75 74 20 3C 3C  | ;....    cout <<
00000180: 20 22 53 51 52 54 28 6E|29 20 3D 20 22 20 3C 3C  |  "SQRT(n) = " <<
00000190: 20 78 3B 0D 0A 7D 0D 0A|0D 0A                   |  x;..}....
```

Tập tin nhị phân

- Tập tin nhị phân không phân thành các dòng, mà dữ liệu được xem như một dãy byte nằm liên tục
- Các ký hiệu \n, \0 hoặc các ký tự đặc biệt được coi như các byte dữ liệu thông thường
- Dữ liệu trong tập tin nhị phân phản ảnh chính xác cách bố trí dữ liệu trong bộ nhớ
  - Một số nguyên trong bộ nhớ cỡ 4 byte thì khi ghi xuống tập tin nhị phân cũng sẽ chính xác là 4 byte có nội dung giống hệt như trong bộ nhớ
- Muốn đọc/ghi dữ liệu nhị phân đúng cách cần phải biết chính xác cách bố trí dữ liệu trong tập tin
  - Một số thậm chí được ghi thành tài liệu kĩ thuật

Tập tin nhị phân

```
00000000: 4D 5A 90 00 03 00 00 00|04 00 00 00 FF FF 00 00  | MZ..........ÿÿ..
00000010: B8 00 00 00 00 00 00 00|40 00 00 00 00 00 00 00  | ¸.......@.......
00000020: 00 00 00 00 00 00 00 00|00 00 00 00 00 00 00 00  | ................
00000030: 00 00 00 00 00 00 00 00|00 00 00 00 80 00 00 00  | ............■...
00000040: 0E 1F BA 0E 00 B4 09 CD|21 B8 01 4C CD 21 54 68  | ..º..´.Í!¸.LÍ!Th
00000050: 69 73 20 70 72 6F 67 72|61 6D 20 63 61 6E 6E 6F  | is program canno
00000060: 74 20 62 65 20 72 75 6E|20 69 6E 20 44 4F 53 20  | t be run in DOS
00000070: 6D 6F 64 65 2E 0D 0D 0A|24 00 00 00 00 00 00 00  | mode....$.......
00000080: 50 45 00 00 64 86 11 00|B4 80 6D 60 00 3E 0F 00  | PE..d■..´€m`.>..
00000090: 39 55 00 00 F0 00 27 00|0B 02 02 18 00 10 07 00  | 9U..ð.'.........
000000A0: 00 58 0A 00 00 18 00 00|00 15 00 00 00 10 00 00  | .X..............
000000B0: 00 00 40 00 00 00 00 00|00 10 00 00 00 02 00 00  | ..@.............
000000C0: 04 00 00 00 00 00 00 00|05 00 02 00 00 00 00 00  | ................
000000D0: 00 00 10 00 00 06 00 00|EB 82 1D 00 03 00 00 00  | ........ë■......
000000E0: 00 00 20 00 00 00 00 00|00 10 00 00 00 00 00 00  | .. .............
000000F0: 00 00 10 00 00 00 00 00|00 10 00 00 00 00 00 00  | ................
00000100: 00 00 00 00 10 00 00 00|00 00 00 00 00 00 00 00  | ................
00000110: 00 A0 0A 00 1C 14 00 00|00 00 00 00 00 00 00 00  | . ..............
00000120: 00 50 09 00 24 75 00 00|00 00 00 00 00 00 00 00  | .P..$u..........
00000130: 00 00 00 00 00 00 00 00|00 00 00 00 00 00 00 00  | ................
00000140: 00 00 00 00 00 00 00 00|00 00 00 00 00 00 00 00  | ................
00000150: 20 D0 0A 00 28 00 00 00|00 00 00 00 00 00 00 00  |  Ð..(...........
00000160: 00 00 00 00 00 00 00 00|D4 A4 0A 00 98 04 00 00  | ........Ô¤..■...
00000170: 00 00 00 00 00 00 00 00|00 00 00 00 00 00 00 00  | ................
00000180: 00 00 00 00 00 00 00 00|2E 74 65 78 74 00 00 00  | .........text...
00000190: 58 0F 07 00 00 10 00 00|00 10 07 00 00 06 00 00  | X...............
000001A0: 00 00 00 00 00 00 00 00|00 00 00 00 60 00 50 60  | ............`.P`
000001B0: 2E 64 61 74 61 00 00 00|F0 50 01 00 00 20 07 00  | .data...ðP... ..
000001C0: 00 52 01 00 00 16 07 00|00 00 00 00 00 00 00 00  | .R..............
000001D0: 00 00 00 00 40 00 70 C0|2E 72 64 61 74 61 00 00  | ....@.pÀ.rdata..
000001E0: 60 C2 00 00 00 80 08 00|00 C4 00 00 00 68 08 00  | `Â.....Ä...h..
```

Quy tắc làm việc với tập tin

- Làm việc với tập tin gồm 2 loại:
  - Thao tác tập tin (tạo, xóa, sao chép, thay đổi thuộc tính,...)
  - Thao tác nội dung tập tin (đọc, ghi, xóa, sửa,...)
- Các thao tác tập tin sử dụng các hàm trong thư viện <cstdio>, đây là thư viện cung cấp các hàm cấp thấp làm việc với hệ thống file, tương thích với các mã nguồn cũ
- Thao tác nội dung tập tin (dù là loại gì), đều theo 3 bước:
  1. Mở tập tin
  2. Thao tác nội dung
  3. Đóng tập tin
- Bước mở tập tin sẽ yêu cầu OS chuẩn bị cho thao tác file
- Bước đóng tập tin sẽ thực sự cập nhật hệ thống file

Phần 2

Làm việc với tập tin văn bản

Ghi chuỗi ra tập tin văn bản

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

int main() {
    // khai báo biến có kiểu tập tin văn bản để ghi ra
    ofstream myfile;
    // mở tập tin có tên là "example.txt"
    myfile.open("example.txt");
    // ghi 100 dòng vào tập tin
    for (int i = 0; i < 100; i++)
        myfile << "Dong thu " << i << endl;
    // đóng tập tin lại
    myfile.close();
}
```

Đọc chuỗi từ tập tin văn bản

```
int main() {
    string line;
    // khai báo biến có kiểu tập tin văn bản để đọc vào
    ifstream myfile;
    // mở tập tin có tên là "example.txt"
    myfile.open("example.txt");
    // đọc hết các dòng của tập tin và in ra
    while (!myfile.eof()) {
        getline(myfile, line);
        cout << line << endl;
    }
    // đóng tập tin lại
    myfile.close();
}
```

Thư viện làm việc với file

- C++ cung cấp các thư viện sau để làm việc với file
  - ofstream: để ghi dữ liệu trên file
  - ifstream: để đọc dữ liệu trên file
  - fstream: để đọc và ghi dữ liệu trên file
- Cơ chế làm việc của C++ với file là "luồng" (stream)
- Cách xây dựng thư viện có thể gây bối rối bởi vì có nhiều cách mở tập tin

```
ifstream input("input_file.txt");
ofstream output("output_file.txt");
```

- Hoặc:

```
fstream input("input_file.txt", istream::in);
fstream output("output_file.txt", ostream::out);
```

- Hai cách đều tạo các biến giúp đọc ghi tập tin

Mở / đóng file

- Thay vì mở file ngay khi khai báo biến, có thể mở sau đó
- Hàm: `open(filename, mode);`
- Trong đó các chế độ (mode) mở file có thể là:

`ios::in`	Mở file để ghi
`ios::out`	Mở file để đọc
`ios::binary`	Mở file chế độ nhị phân (binary)
`ios::ate`	Thiết lập vị trí ban đầu ở cuối file, nếu không có cờ này thì vị trí ban đầu ở đầu file
`ios::app`	Nội dung ghi vào tệp sẽ được thêm vào cuối tệp
`ios::trunc`	Nếu file đã tồn tại thì nội dung trong file sẽ được ghi đè bằng nội dung mới

Mở / đóng file

- Có thể kết hợp nhiều chế độ cùng một lúc

```
f.open("abc.txt", ios::out | ios::app | ios::binary);
```

- Các biến loại ofstream, ifstream và fstream có các chế độ mặc định khác nhau, trong trường hợp ta không cung cấp tham số mode

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

- Đôi khi hàm mở file không thành công, ta có thể kiểm tra lại bằng hàm is_open

```
if (myfile.is_open()) {...
```

- Đóng tập tin: `myfile.close();`

Đọc / ghi file văn bản

- Cách thức đọc dữ liệu an toàn nhất là sử dụng getline

```
while (getline(myfile, line)) ...
```

  - Tương tự như đọc chuỗi, có thể dùng thao tác tương tự cin
- Cách thức ghi dữ liệu tương tự như ghi ra cout
- Kiểm tra trạng thái luồng trong quá trình xử lý
  - bad() trả về true nếu đọc / ghi lỗi, ví dụ khi chúng ta ghi dữ liệu vào file chưa được mở
  - fail() trả về true tương tự như bad(), ngoài ra chúng còn trả về true trong trường hợp lỗi định dạng, chẳng hạn như cố gắng đọc một số nguyên nhưng giá trị không phù hợp
  - eof() trả về true nếu đã đến cuối file
  - good() trả về true khi mọi việc hoàn hảo, tức là mọi thao tác tập tin trước đó đều hoạt động tốt

Phần 3

Làm việc với tập tin nhị phân

Đọc ghi tập tin nhị phân

- C++ cung cấp hai hai hàm đọc ghi dữ liệu theo khối, sử dụng riêng cho tập tin nhị phân

```
write(memory_block, size);
read(memory_block, size);
```

- Tham số memory_block là một con trỏ kiểu char * là địa chỉ mảng byte lưu trữ các dữ liệu cần ghi hoặc đọc
- Tham số size là số byte sẽ được đọc hoặc ghi

Ghi dữ liệu vào tập tin nhị phân

```
struct Person {
    char name[50];
    int age;
    char phone[24];
};

int main() {
    Person me = { "txnam", 18, "091.210.2165" };
    Person friends[5000];
    ofstream outfile;
    outfile.open("fb.data", ios::binary | ios::out);
    outfile.write(&me, sizeof(me));
    outfile.write(book, 5000 * sizeof(Person));
    outfile.close();
}
```

Vị trí con trỏ file

- Con trỏ file là vị trí mà đầu đọc / ghi dữ liệu của hệ thống hiện tại đang trỏ tới, khi phát lệnh đọc (hoặc ghi) dữ liệu, đầu đọc sẽ thực hiện ở vị trí hiện tại và dịch chuyển tới vị trí ngay sau đó
  - Giống con trỏ trên màn hình console
- Con trỏ file tồn tại trong cả file văn bản và file nhị phân, nhưng chúng thường được dùng với file nhị phân, do nhu cầu đọc dữ liệu ngẫu nhiên
- Một số luồng không thể đọc chính xác vị trí đầu đọc
- Hai hàm cho phép lấy vị trí hiện tại của con trỏ file:
  - Hàm tellg() trả về vị trí đầu đọc
  - Hàm tellp() trả về vị trí đầu ghi

Vị trí con trỏ file

- Hai hàm cho phép dịch chuyển các đầu đọc này:
  - seekg(position, direction);
  - seekp(position, direction);
- Tham số position là vị trí dịch chuyển tới, tính theo đơn vị byte
- Tham số direction quy định cách thức tính vị trí mới, nếu không có tham số này thì mặc định tính từ đầu tập tin

`ios::beg`	Vị trí mới được tính từ vị trí bắt đầu tập tin
`ios::cur`	Vị trí mới được tính từ vị trí hiện tại của con trỏ
`ios::end`	Vị trí mới được tính từ vị trí cuối tệp tin

Phần 4

Bài tập

Bài tập

1. Nhập vào ba số thực a, b, c và ghi chúng vào file văn bản "doubles.t"
2. Đọc giá trị của ba số thực vừa ghi vào file "doubles.t" ở trên và ghi ra màn hình
3. Nhập vào ba số thực a, b, c và ghi chúng vào file nhị phân "doubles.d"
4. Đọc giá trị của ba số thực vừa ghi vào file "doubles.d" ở trên và ghi ra màn hình
5. Nhập số n và mảng A có n số nguyên, sau đó ghi vào tập tin văn bản "5.t" các dữ liệu theo quy cách sau:
   - Dòng đầu tiên: ghi số n
   - Dòng thứ 2: ghi n giá trị trong mảng A

Bài tập

6. Nhập dữ liệu được ghi vào file văn bản trong file "5.t", sau đó sắp xếp dữ liệu của mảng A giảm dần và ghi lại toàn bộ dữ liệu ra file "6.t" theo định dạng như bài 5
7. Tạo tập tin văn bản "7.t" chứa ngẫu nhiên 5000 số thực trong khoảng từ -1 đến +1.
8. Đọc dữ liệu từ tập tin "7.t" ở trên và ghi ra tập tin nhị phân "8.bin"
9. Hoàn thiện chương trình ở slide 20 để chạy được, sinh ngẫu nhiên tên, tuổi và số điện thoại của 5000 người bạn, sau đó ghi vào tập tin
10. Viết chương trình đọc dữ liệu từ file do bài 9 tạo ra. Nhập số n và hiển thị thông tin về người bạn thứ n.