BÀI TẬP THỰC HÀNH TUẦN 3

NỘI DUNG:

- Thao tác trên chuỗi, vector và tập tin

Qui định nộp bài tập:

- Mỗi bài tập tương ứng với 1 project, tên Project là BaiXX, với XX là thứ tự của bài tập (Ví dụ bài tập 1 tên Project tương ứng là Bai02).

- Tất cả các bài tập được đặt bên trong một thư mục, tên thư mục theo qui định như sau: **BTH3_HoVaTen_MSSV**. Ví dụ Sinh viên Nguyễn Sơn Trà có MSSV là 19521269 thì đặt tên thư mục như sau: **BTH3_NguyenSonTra_19521269**

- Sau đó nén thư mục trên thành tập tin **.zip hoặc .rar** (tên file nén cũng theo qui định như tên thư mục). Ví dụ **BTH3_NguyenSonTra_19521269.rar**

- **Lưu ý xóa thư mục được phát sinh sau khi biên dịch (thư mục Debug, .vs) của mỗi project**

- Hình thức nộp bài: Nộp trên website môn học theo thời gian qui định.

- Những bài nộp không đúng qui định như trên sẽ không được chấm điểm (0 điểm)

- Tất cả các bài làm có tính chất sao chép (copy) sẽ nhận 0 điểm

1. Thao tác trên chuỗi - String

- Trong ngôn ngữ C, một chuỗi các ký tự được lưu trong một biến kiểu char*. Trong C++, một chuỗi các ký tự được lưu trong một biến kiểu string. Kiểu dữ liệu string này giúp đơn giản hóa quá trình xử lý chuỗi phức tạp trên C bằng việc hỗ trợ nhiều hàm xử lý tính toán trên chuỗi.
- Kiểu dữ liệu **string** được định nghĩa trong thư viện string của C++, cần phải khai báo ở đầu chương trình trước khi sử dụng với cú pháp:

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

- Cú pháp để khai báo một biến kiểu string như sau:

```
string myString;
```

- Gán giá trị cho biến kiểu string
  - Gán giá trị trực tiếp thông qua một chuỗi nằm trong dấu nháy kép “”

```
string myString = "Hello World";
```

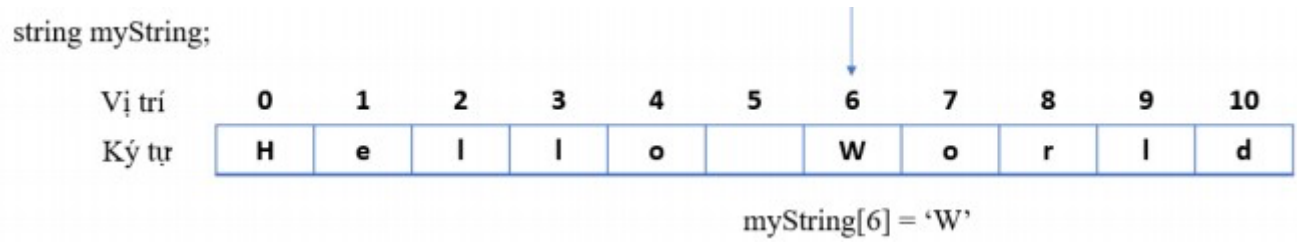
  - Gán giá trị gián tiếp thông qua một biến khác

```
string myString1 = "Hello World";
string myString2 = myString1;
```

- Một số thao tác xử lý trên chuỗi string
  - Lấy ký tự tại vị trí thứ i trong chuỗi: do biến string là một mảng 1 chiều các ký tự nên có thể lấy giá trị theo cách lấy của mảng tương ứng trong dấu ngoặc vuông []

```
string myString = "Hello World";
cout << myString[6] << endl;
```

```
W
Press any key to continue . . .
```



  - Nhập một chuỗi gồm có ký tự trắng: nếu sử dụng cin thông thường, giá trị lấy được dừng lại ở khoảng trắng. Do đó nếu muốn lấy toàn bộ chuỗi ký tự bao gồm khoảng trắng, ta sử dụng hàm getline với cú pháp:

```
string myString;
cin >> myString;
cout << myString;
```

```
Hello World
HelloPress any key to continue . . .
```

getline(cin, tên biến string)

```
string myString;
getline(cin, myString);
cout << myString;
```

```
Hello World
Hello WorldPress any key to continue . . .
```

- Lấy độ dài của chuỗi ký tự: sử dụng hàm length() với cú pháp: lưu ý, độ dài được tính từ 1, vị trí được tính từ 0

```
string myString = "Hello World";
cout << myString.length() << endl;
```

```
11
Press any key to continue . . .
```

- Ghép nối chuỗi: có thể sử dụng toán tử "+" với cú pháp:

```
string myString1 = "Hello ";
string myString2 = "World";
cout << myString1 + myString2 + "\n";
cout << myString1 << myString2 << "\n";
```

```
Hello World
Hello World
Press any key to continue . . .
```

- Một số các hàm chức năng khác có thể tìm tại trang: http://www.cplusplus.com/reference/string/string/

2. Thao tác với Vector

- Vector là lớp đối tượng của thư viện STL hỗ trợ các chức năng giống như mảng thông thường, do đó có tính chất giống như mảng.
- Một vector có thể được xem như là một mảng động. Vector có khả năng tự cấp phát với số phần tử có thể co dãn.
- Vector cung cấp cơ chế tự động quản lý bộ nhớ, đặc biệt là không cần phải sử dụng con trỏ để có mảng động.
- Các thao tác trên mảng như thêm, xóa phần tử, truy xuất phần tử được cài đặt sẵn và hiệu quả hơn so với việc sử dụng mảng động.
    - Vector hiệu quả với thao tác thêm và xóa ở vị trí cuối mảng.
    - Đơn giản hóa rất nhiều thao tác: thay đổi số lượng phần tử của mảng, lấy giá trị tại vị trí thứ bất kỳ, thay đổi giá trị tại vị trí bất kỳ, … đều đã được cài đặt sẵn nên rất tiện cho việc lập trình.
- Để sử dụng vector, ta cần khai báo thư viện như sau:

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

- Khai báo một vector với cú pháp:

```
vector<kiểu dữ liệu> tên biến;

vector<int> v1;
```

    - Ví dụ tại dòng 2 là để khai báo 1 biến vector kiểu số nguyên, tên là v1
    - Vector tự động co dãn kích thước, nên có thể không cần khai báo trước số lượng phần tử.
- Một vector có thể khai báo trước số lượng phần tử hoặc không cần khai báo trước, nếu khai báo thì có thể sử dụng theo cú pháp:

```
vector<kiểu dữ liệu> tên biến(số lượng phần tử);

vector<int> v1(10);
```

- Truy xuất phần tử trong vector giống hệt như của mảng: sử dụng dấu ngoặc vuông

```
v1[5]; //Lấy phần tử ở vị trí thứ 5 trong vector v1
```

- **Thêm** phần tử:
    - Thêm một phần tử vào cuối vector:

```
v1.push_back(5); //Thêm phần tử 5 vào cuối vector v1
```

    - Thêm một phần tử vào vị trí i bất kỳ trong vector:

```
v1.insert(v1.begin()+i,50); //Thêm phần tử 50 vào vị trí thứ i của vector v1
```

- **Xóa** phần tử:

- Xóa phần tử ở cuối vector:

```
v1.pop_back();
```

- Xóa một phần tử ở vị trí i trong vector:

```
v1.erase(v1.begin()+ i); //Xóa phần từ tại vị trí thứ i trong vector
```

- Xóa các phần tử từ tại vị trí đến vị trí j trong vector (Xóa trong khoảng [i,j) và j>=i):

```
v1.erase(v1.begin()+ i,v1.begin() + j);
```

- **Duyệt** các phần tử trong vector:
  - Duyệt lần lượt các phần tử của vector:

```
for(int i : v1)
{
    //Thao tác trên phần tử vector
    cout<<i<<endl;
}
```

  - Duyệt vector sử dụng index:

```
for(int i = 0 ; i < v1.size(); i++)
{
    //Thao tác trên phần tử vector
    cout<<v1[i]<<endl;
}
```

- Cú pháp về thao tác sử dụng (tương tự như gọi hàm) trong vector:

```
<tên biến vector>.<tên phương thức sử dụng>
```

Phương thức hỗ trợ của vector	**Tác dụng**	**Ví dụ** **vector<int> a;**
.size()	Lấy tổng số lượng phần tử hiện tại của vector	a.size();
.push_back (giá trị cần thêm)	Thêm một phần tử vào cuối vector	a.push_back(10); Thêm giá trị 10 vào cuối vector
.pop_back()	Xóa phần tử cuối của vector	a.pop_back();
.clean()	Xóa toàn bộ vector	a.clean();
.resize(số lượng phần tử)	Thay đổi kích thước của vector	a.resize(5); Thay đổi kích thước của vector thành 5 phần tử

- Ví dụ nhập và xuất danh sách Phân số sử dụng vector:

```
std::vector<PhanSo*> listPhanSo;

int n;
cout << "Nhap so luong phan so: ";
cin>>n;

//Nhập danh sách phân số
for(int i = 0 ; i < n ; i++)
{
    PhanSo* temp = new PhanSo();
    temp->Nhap();
    listPhanSo.push_back(temp);
}

//Xuất danh sách phân số
for(PhanSo* i: listPhanSo)
    i->Xuat();
```

3. Thao tác trên tập tin

- Khai báo thư viện hỗ trợ thao tác với tập tin: **iostream, fstream và string**.

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

- Cần tạo một biến dùng để quản lý việc đọc và ghi file, biến này có kiểu dữ liệu đặc biệt là fstream. Các thao tác trên tập tin đều được xử lý thông qua biến này, khai báo với cú pháp như sau:

fstream <tên biến>;

```
fstream myFile;
```

- Để thao tác được với tập tin, cần phải mở tập tin ra trước, giá trị cần truyền vào chính là đường dẫn đến tập tin và kiểu mở tập tin, câu lệnh để mở tập tin:

open("tên file", chế độ mở file)

```
myFile.open("LopHoc.txt", fstream::in);
```

- Nếu để tên file không kèm theo đường dẫn, file được mặc định nằm ở trong folder lúc tạo project Visual Studio.
- Các chế độ mở file được quy định bằng những tham số có sẵn trong chương trình và được gọi thông qua fstream, cú pháp và nội dung chế độ mở file được quy định như sau:

Chế độ mở file	Cú pháp
Mở file để đọc dữ liệu từ file vào chương trình	fstream::in
Mở file để ghi dữ liệu từ chương trình vào file	fstream::out
Mở file và thao tác dưới dạng nhị phân	fstream::binary
Mở file với vị trí bắt đầu thao tác nằm ở cuối file	fstream::ate
Mở file và thêm nội dung vào cuối file (vẫn giữ nội dung file)	fstream::app
Mở file và xóa tất cả nội dung file (ghi đè file mới)	fstream::trunc

- Có thể sử dụng nhiều thao tác cùng lúc bằng toán tử | (toán tử hoặc dùng trong câu điều kiện if), ví dụ mở file để ghi và ghi đè file:

```
myFile.open("LopHoc.txt", fstream::in | fstream::app);
```

- Kiểm tra việc mở file có thực thi thành công hay không, dùng để kiểm tra xem có mở được file chưa, xử lý trong trường hợp không tìm thấy file hay do sai tên file.

```
if (myFile.fail())
{
      //Xử lý không mở file được tại đây
}
```

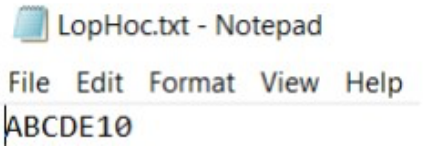
- Kiểm tra đã duyệt hết file bằng câu lệnh: eof() (end of file)
- Sau khi thực hiện các thao tác cần thiết, cần phải đóng file lại với câu lệnh:

```
myFile.close();
```

- Thao tác đọc và ghi dữ liệu có thể thao tác trực tiếp với biến của chương trình, sử dụng toán tử tương tự như nhập (>>) và xuất (<<)
- Để ghi dữ liệu vào trong tập tin, trước tiên cần mở tập tin để ghi, sau đó sử dụng toán tử xuất để ghi giá trị vào trong tập tin, ví dụ:

```
fstream myFile;
myFile.open("LopHoc.txt", fstream::in | fstream::app);
myFile << "ABCDE";
int x = 10;
myFile << x;
```

Ghi chuỗi "ABCDE" và giá trị 10 vào trong tập tin .txt:



- Để đọc dữ liệu vào trong tập tin, trước tiên cần mở tập tin để đọc, sau đó sử dụng toán tử nhập (>>) để đọc giá trị vào trong biến chương trình, ví dụ:

```
fstream myFile;
myFile.open("LopHoc.txt");
string data;
myFile >> data;
```

- Lưu ý, kiểu dữ liệu đọc dược tự ép kiểu về kiểu dữ liệu khai báo trong chương trình, thông thường sẽ là kiểu chuỗi, rồi đến kiểu số.
- Lưu ý, các giá trị được đọc cách nhau bằng khoảng trắng. Nếu muốn đọc theo từng dòng (có khoảng trắng), sử dụng hàm getline như sau:

```
fstream myFile;
myFile.open("LopHoc.txt");
string data;
getline(myFile, data);
```

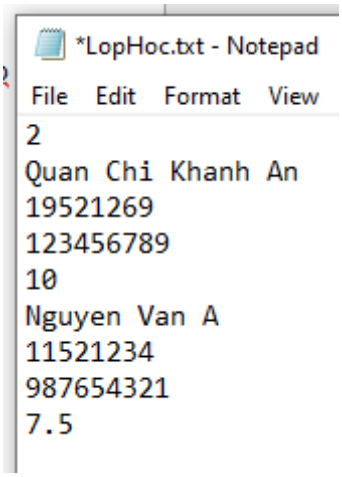
4. Bài tập thực hành

1. Viết chương trình C++ cài đặt đối tượng Học Sinh, trong đó mỗi học sinh gồm có thông tin được mô tả như sau:
    - Họ tên học sinh kiểu string là các ký tự không dấu (bao gồm cả ký tự khoảng trắng) và các ký tự chỉ nằm trong bảng 24 chữ cái tiếng anh.
    - Mã số học sinh: Chuỗi gồm 8 kí tự (Ví dụ: 19521269)
    - Số điện thoại liên lạc: gồm 1 dãy số từ 9 đến 11 ký tự, kiểu string, chỉ bao gồm các số
    - Điểm trung bình: gồm 1 số float giới hạn từ 0 đến 10

Ngoài ra, trong đối tượng học sinh, còn hỗ trợ các thao tác:

- Nhập từ console thông tin về họ tên, mã số học sinh, số điện thoại và điểm trung bình
- Kiểm tra tính hợp lệ của cả 4 thông tin nhập vào, nếu thông tin nhập sai yêu cầu nhập lại
- Xuất toàn bộ thông tin của học sinh ra console

2. Viết chương trình C++ cài đặt đối tượng Lớp Học, trong đó mỗi lớp học gồm có thông tin được mô tả như sau:
    - Danh sách học sinh, là một **vector** các đối tượng Học Sinh
    - Thêm một học sinh mới vào trong danh sách học sinh, thông báo ra màn hình nếu thêm thành công, nếu đã có học sinh đó trong lớp (trùng họ tên theo từng ký tự), xuất ra màn hình “Da co hoc sinh trong lop”
    - Xóa một học sinh đã có trong danh sách khỏi danh sách học sinh, dữ liệu nhập vào là họ tên của học sinh đó, thông báo ra màn hình nếu xóa thành công, nếu không tìm thấy, xuất ra màn hình “Khong co hoc sinh trong lop”
    - Xuất toàn bộ thông tin tất cả học sinh trong lớp học
    - Xuất toàn bộ thông tin các học sinh trong lớp học có điểm trung bình > 8
    - Nhập danh sách học sinh từ tập tin “LopHoc.txt” nằm chung ở thư mục chứa project với cấu trúc tập tin được mô tả như sau: (các dữ liệu từ tập tin này đều là thông tin học sinh hợp lệ)
        - Dòng đầu tiên: 1 số nguyên n, thể hiện cho tổng số học sinh
        - n * 4 dòng tiếp theo, mỗi 4 dòng sẽ gồm có thông tin:
            - Họ tên học sinh
            - Mã số học sinh
            - Số điện thoại
            - Điểm trung bình (điểm phần thập phân được cách bằng ký tự dấu chấm)


*LopHoc.txt - Notepad
File Edit Format View
2
Quan Chi Khanh An
19521269
123456789
10
Nguyen Van A
11521234
987654321
7.5