ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

BÁO CÁO THỰC HÀNH 2

LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

Sinh viên: Đồng Quốc Thắng - 23521421

Giáo viên hướng dẫn: Nguyễn Ngọc Quí

Ngày ... tháng ... năm ...

Mục lục

1 Bài tập về nhà 1

1.1 Bài 1

Bài tập 1: Để tìm phân số lớn nhất và nhỏ nhất trong một mảng phân số, trước tiên, cần nhập vào số lượng phân số n. Sau đó, lần lượt nhập tử số và mẫu số của từng phân số. Khi mẫu số của bất kỳ phân số nào được nhập bằng 0, người dùng phải nhập lại giá trị hợp lệ cho mẫu số. Sau khi hoàn tất việc nhập liệu, kết quả sẽ xuất ra hai phân số có giá trị nhỏ nhất và lớn nhất theo định dạng: tử số nhỏ nhất / mẫu số nhỏ nhất và tử số lớn nhất / mẫu số lớn nhất. Lưu ý rằng trong bài toán này, không cần thiết phải rút gọn các phân số.

1.1.1 Input, Output và Solution

Input: Một dãy các phân số, nhập theo thứ tự của tử số phân số thứ n, mẫu số phân số thứ n, tử số phân số thứ n+1, mẫu số phân số thứ n+1,...

Output: Xuất ra 2 phân số có giá trị nhỏ nhất và lớn nhất theo định dạng tử số/mẫu số

Solution: Xây dựng lớp PhanSo để lưu trữ phân số và các method gồm có rút gọn, nhập, xuất phân số, getter, setter cơ bản, ngoài ra còn định nghĩa các operator + - > <.

Sau đó xây dựng lớp arrPhanSo để lưu trữ 1 mảng các phân số, lớp này có các method như nhập, xuất, lấy phân số lớn nhất, nhỏ nhất

1.1.2 Code

https://github.com/LowTechTurtle/IT002_OOP/tree/main/Lab2/Lab1_BTVN/bai1

1.1.3 Testcase

```
└─(08:56:31 on main)──> ./main
Nhap so phan so: 4
Nhap tu so cua ps thu 1: 1
Nhap mau so cua ps thu 1: 2
Nhap tu so cua ps thu 2: 3
Nhap mau so cua ps thu 2: 7
Nhap tu so cua ps thu 3: 2
Nhap mau so cua ps thu 3: 9
Nhap tu so cua ps thu 4: 8
Nhap mau so cua ps thu 4: 1
Phan so lon nhat la:
8/1
Phan so nho nhat la:
2/9
┌─(~/IT002_OOP/Lab2/Lab1_BTVN/bai1)
└─(08:56:43 on main)──> ./main
Nhap so phan so: 5
Nhap tu so cua ps thu 1: -1
Nhap mau so cua ps thu 1: 2
Nhap tu so cua ps thu 2: -3
Nhap mau so cua ps thu 2: 5
Nhap tu so cua ps thu 3: 6
Nhap mau so cua ps thu 3: -8
Nhap tu so cua ps thu 4: 2
Nhap mau so cua ps thu 4: 3
Nhap tu so cua ps thu 5: 5
Nhap mau so cua ps thu 5: 3
Phan so lon nhat la:
5/3
Phan so nho nhat la:
6/-8
```

1.2 Bài 2

Bài tập 2: Để giải bài toán tìm phân số lớn thứ k và bé thứ k trong một mảng gồm n phân số, trước tiên, ta cần nhập số lượng phân số n và số nguyên dương k từ bàn phím. Sau đó, các phân số được nhập vào, có thể nhập trên cùng một dòng hoặc mỗi phân số trên một dòng riêng. Sau khi hoàn thành việc nhập liệu, chương trình sẽ tìm và xuất ra phân số lớn thứ k và phân số bé thứ k trong mảng, theo định dạng: tử số của phân số lớn thứ k / mẫu số của phân số lớn thứ k và tử số của phân số bé thứ k / mẫu số của phân số bé thứ k. Nếu không có phân số lớn thứ k hoặc bé thứ k, chương trình sẽ không xuất ra gì. Lưu ý rằng không cần rút gọn các phân số khi nhập vào.

1.2.1 Input, Output và Solution

Input: Nhập số lượng phân số n vào bàn phím, sau đó nhập lần lượt các số nguyên là tử và mẫu của các phân số.

Sau đó nhập vào k_1 để chọn ra phân số lớn thứ k_1 và nhập k_2 để chọn phân số nhỏ thứ k_2

Output: Phân số lớn thứ k_1 và phân số nhỏ thứ k_2 trong các phân số vừa nhập

Solution: Giống với bài 1, ta xây dựng 2 lớp PhanSo và lớp arrPhanSo, nhưng ngoài ra ta còn thêm các hàm dùng để tìm phân số nhỏ thứ k và lớn thứ k. Các hàm này được viết bằng cách sort arrPhanSo theo chiều tăng dần, và chọn lần lượt phân số thứ k và n-k

1.2.2 Code

https://github.com/LowTechTurtle/IT002_OOP/tree/main/Lab2/Lab1_BTVN/bai2

1.2.3 Testcase

```
┌──(~/IT002_OOP/Lab2/Lab1_BTVN/bai2)──
└─(09:03:38 on main)──> ./main
Nhap so phan so: 1
Nhap tu so cua ps thu 1: 2
Nhap mau so cua ps thu 1: 3
Nhap k de tim phan so lon thu k: 1
Phan so lon thu 1 la 2/3
Nhap k de tim phan so nho thu k: 1
Phan so nho thu 1 la 2/3
┌──(~/IT002_OOP/Lab2/Lab1_BTVN/bai2)──
└─(09:03:46 on main)──> ./main
Nhap so phan so: 4
Nhap tu so cua ps thu 1: 1
Nhap mau so cua ps thu 1: 2
Nhap tu so cua ps thu 2: -1
Nhap mau so cua ps thu 2: 2
Nhap tu so cua ps thu 3: 4
Nhap mau so cua ps thu 3: 7
Nhap tu so cua ps thu 4: 2
Nhap mau so cua ps thu 4: -9
Nhap k de tim phan so lon thu k: 2
Phan so lon thu 2 la 1/2
Nhap k de tim phan so nho thu k: 3
Phan so nho thu 3 la 1/2
┌──(~/IT002_OOP/Lab2/Lab1_BTVN/bai2)──
└─(09:07:49 on main)──>
```

1.3 Bài 3

Bài tập 3: Cho một mảng gồm n phân số, mỗi phân số có dạng a_i/b_i với i từ 1 đến n. Nhiệm vụ của bạn là tìm một tập hợp con các phân số sao cho tích của các phân số trong tập hợp con bằng một phân số đích a_k/b_k đã cho. Trong trường hợp có nhiều tập hợp con thoả mãn điều kiện, bạn cần chọn tập hợp con có số lượng phần tử ít nhất. Nếu không có tập hợp con nào thoả mãn, không xuất ra kết quả nào. Để giải bài toán, bạn cần thực hiện các bước sau: nhập số lượng phân số n, nhập các phân số, và nhập phân số đích a_k/b_k. Sau đó, tìm và xuất tập hợp con có tích bằng phân số đích, sắp xếp các phân số trong tập hợp con theo thứ tự từ bé đến lớn. Lưu ý rằng không cần rút gọn các phân số khi nhập vào.

1.3.1 Input, Output và Solution

Input: Một mảng gồm n phân số, và một phân số đích a_k/b_k.

Output: Một tập hợp con của mảng n phân số trên sao cho tích của tất cả các phân số trong tập con bằng với phân số đích. Nếu có nhiều tập con thỏa mãn, phải chọn tập con có ít phần tử nhất.

Solution: ta xét 2^n tập hợp con của tập hợp vừa được nhập vào, với mỗi tập con, ta tìm xem tích của các phần tử trong tập con đó có bằng phân số đích không. Nếu có, và số phần tử của tập con đó ít hơn số phần tử của tập con đang lưu hiện tại thì ta sẽ chọn tập chọn con mới. Ban đầu nếu chưa có tập con nào thì ta sẽ chọn luôn tập ta tìm được đầu tiên.

1.3.2 Code

https://github.com/LowTechTurtle/IT002_OOP/tree/main/Lab2/Lab1_BTVN/bai3

1.3.3 Testcase

```
┌──(~/IT002_OOP/Lab2/Lab1_BTVN/bai3)─────────────
└─(09:26:43 on main)──> ./main
Nhap so luong phan so: 3
Nhap danh sach cac phan so:
Nhap tu so thu 1 1
Nhap mau so thu 1 2
Nhap tu so thu 2 5
Nhap mau so thu 2 7
Nhap tu so thu 3 4
Nhap mau so thu 3 9
Nhap phan so dich: Nhap tu so dich: 5
Nhap mau so dich: 14
Tap hop con nho nhat co tich bang phan so dich:
1/2 5/7
┌──(~/IT002_OOP/Lab2/Lab1_BTVN/bai3)─────────────
└─(09:27:04 on main)──> ./main
Nhap so luong phan so: 2
Nhap danh sach cac phan so:
Nhap tu so thu 1 3
Nhap mau so thu 1 4
Nhap tu so thu 2 -2
Nhap mau so thu 2 3
Nhap phan so dich: Nhap tu so dich: 1
Nhap mau so dich: 1
Ko co tap con nao thoa man
┌──(~/IT002_OOP/Lab2/Lab1_BTVN/bai3)─────────────
└─(09:27:17 on main)──>
```

1.4 Bài 4

Bài tập 4: Arnold's Cat Map là một phép biến đổi ma trận đơn giản nhưng thú vị, có tác dụng chuyển đổi dữ liệu từ một dạng có quy luật thành một dạng có vẻ hỗn độn. Được định nghĩa trên một ma trận vuông kích thước $m \times m$, phép biến đổi Arnold's Cat Map áp dụng công thức $(x, y) = ((2i + j) \mod m, (i + j) \mod m)(x, y)$ để xác định vị trí mới của mỗi phần tử trong ma trận, với (i,j) là chỉ số hàng và cột của phần tử ban đầu. Nhiệm vụ của bài toán là xác định hệ số chu kỳ k của phép biến đổi này, tức là số lần biến đổi cần thiết để ma trận trở về trạng thái ban đầu. Đầu vào của bài toán bao gồm kích thước ma trận và ma trận dữ liệu, và đầu ra là hệ số chu kỳ k. Để giải bài toán, bạn cần áp dụng phép biến đổi Arnold's Cat Map nhiều lần và đếm số lần biến đổi cho đến khi ma trận trở về trạng thái ban đầu

1.4.1 Input, Output và Solution

Input: Một số nguyên n và một ma trận có cỡ nxn

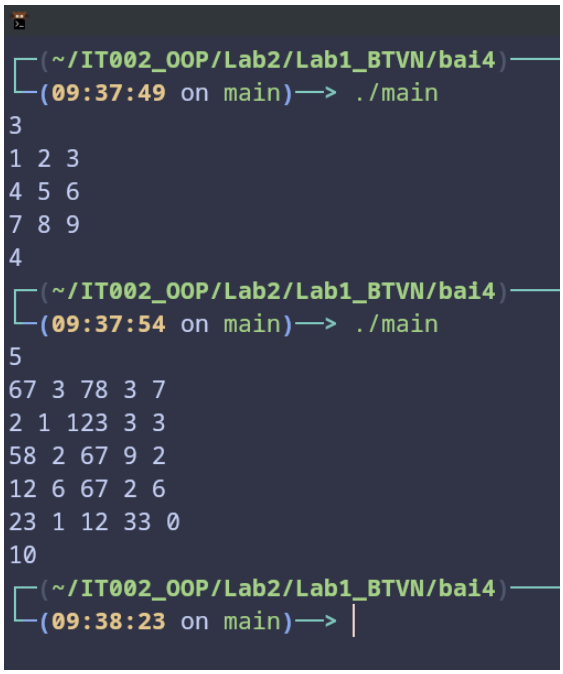
Output: Số phép biển đổi Arnold's Cat để đưa một ma trận trở về ma trận ban đầu.

Solution: Ta sử dụng công thức $(x, y) = ((2i + j) \mod m, (i + j) \mod m)(x, y)$ để biến đổi ma trận, với mỗi lần biến đổi ta so với ma trận ban đầu để xem thử ma trận được biến đổi có giống với ma trận ban đầu không. Nếu có, ta in ra số lần ta đã biến đổi, nếu không, ta tiếp tục biến đổi ma trận.

1.4.2 Code

https://github.com/LowTechTurtle/IT002_OOP/tree/main/Lab2/Lab1_BTVN/bai4

1.4.3 Testcase



1.5 Bài 5

Bài tập 5: Chúng ta có một bức ảnh nhị phân dưới dạng ma trận kích thước $m \times n$, trong đó mỗi phần tử có giá trị 0 hoặc 1. Bức ảnh này chứa các đối tượng hình chữ nhật, với các hình chữ nhật có gốc tọa độ là góc trên bên trái và có tất cả các phần tử bên trong là 1. Nhiệm vụ của bạn là tìm tất cả các hình chữ nhật trong bức ảnh, với điều kiện kích thước tối thiểu là 2 x 2. Để giải quyết bài toán, bạn cần thực hiện các bước sau: đầu tiên, nhập kích thước ma trận mmm và nnn, sau đó nhập ma trận nhị phân kích thước $m \times n$. Cuối cùng, xác định và xuất danh sách các hình chữ nhật theo định dạng [x, y, w, h], trong đó x và y là tọa độ gốc của hình chữ nhật, và w và h lần lượt là chiều rộng và chiều cao của hình chữ nhật. Danh sách các hình chữ nhật cần được sắp xếp theo thứ tự từ trái sang phải và từ trên xuống dưới.

1.5.1 Input, Output và Solution

Input: 2 số nguyên m, n lần lượt là số hàng và số cột của ma trận, sau đó là lần lượt m x n số có giá trị 0 hoặc 1.

Output: Danh sách các hình chữ nhật theo định dạng [x, y, w, h], trong đó x và y là tọa độ gốc của hình chữ nhật, và w và h lần lượt là chiều rộng và chiều cao của hình chữ nhật.

Solution: Ta duyệt qua tất cả các phần tử của ma trận, sau đó ta xét một ma trận có có hợp lệ hay không. Một ma trận con được coi là hợp lệ khi tất cả các phần tử của ma trận con đều là 1 và các hàng, cột xung quanh ma trận con đó không chứa số 1 nào cả.

1.5.2 Code

https://github.com/LowTechTurtle/IT002_OOP/tree/main/Lab2/Lab1_BTVN/bai5

1.5.3 Testcase

```
└─(10:01:08 on main *)──> ./main
10 10
0 0 1 0 0 0 0 0 0 1
0 0 1 0 1 0 0 1 1 0
0 0 0 0 0 0 0 1 1 0
1 1 1 0 0 0 0 0 1 0
1 1 1 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 1 1 0 0 0
0 0 0 0 0 1 1 0 0 0
[0, 3, 3, 2]
[5, 8, 2, 2]
┌─(~/IT002_OOP/Lab2/Lab1_BTVN/bai5)──
└─(10:01:22 on main *)──> ./main
10 10
0 0 1 0 0 0 0 0 0 1
0 0 1 0 1 0 0 1 1 0
0 0 0 0 0 0 0 1 1 0
1 1 1 0 0 0 0 0 1 0
1 1 1 0 0 0 0 0 0 0
0 0 0 0 0 1 1 0 1 1
0 0 0 0 0 1 1 0 1 1
1 1 1 0 0 0 0 0 0 1
1 1 1 0 0 1 1 0 0 0
1 1 1 0 0 1 1 0 0 0
[0, 3, 3, 2]
[5, 5, 2, 2]
[0, 7, 3, 3]
[5, 8, 2, 2]
┌─(~/IT002_OOP/Lab2/Lab1_BTVN/bai5)──
└─(10:01:31 on main *)──>
```

2 Bài tập trên lớp 2

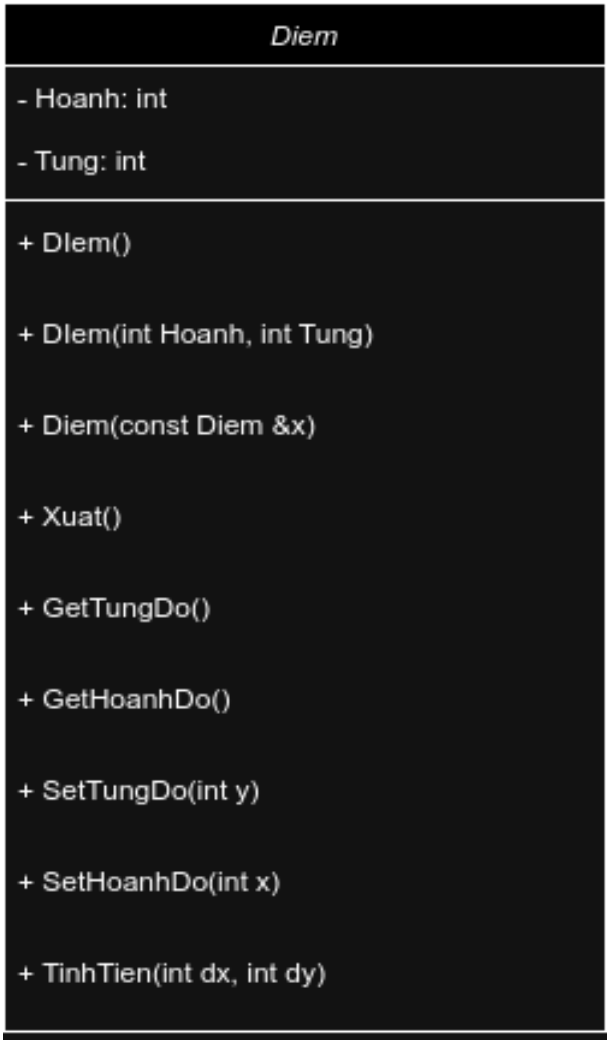
2.1 Bài 1

Bài tập 1: Xây dựng lớp điểm:

- Thuộc tính: iHoanh, iTung
- Phương thức: Diem(), Diem(int Hoanh, int Tung), Diem(const Diem &x), Xuat(), GetTungDo(), GetHoanhDo(), SetTungDo(), SetHoanhDo(), TinhTien()

Yêu cầu: Thực hiện xây dựng lớp, vẽ class diagram và khai báo các thuộc tính, phương thức. Viết nội dung vào các phương thức đã khai báo. Gọi các phương thức trong hàm main()

2.1.1 Class Diagram



2.1.2 Input, Output và Solution

Input: Hoành độ, tung độ của 1 điểm. Ngoài ra khi gọi hàm tịnh tiến, phải đưa ra dx, dy là độ tịnh tiến theo phương x và độ tịnh tiến theo phương y

Output: Xuất ra màn hình tọa độ của điểm đó theo format (x, y)

Solution: Tạo class Diem có các thuộc tính Hoanh, Tung. Method TinhTien một điểm bằng cách cho $x+ = dx$ và $y+ = dy$

2.1.3 Code

https://github.com/LowTechTurtle/IT002_OOP/tree/main/Lab2/Lab2_BTTL/bai1

2.1.4 Testcase

```
┌──(~/IT002_OOP/Lab2/Lab2_BTTL/bai1)
└─(21:14:11 on main *)──> ./main
Tao diem A bang default constructor
(0, 0)
Nhap hoanh do va tung do cua diem B: 2 3
(2, 3)
Tao diem C bang copy constructor, copy diem B
(2, 3)
Hoanh do cua B: 2
Tung do cua B: 3
Nhap do tinh tien cho diem B:
-1 4
Diem B sau khi tinh tien: (1, 7)
┌──(~/IT002_OOP/Lab2/Lab2_BTTL/bai1)
└─(21:14:17 on main *)──> ./main
Tao diem A bang default constructor
(0, 0)
Nhap hoanh do va tung do cua diem B: 3.55 4.77
(3.55, 4.77)
Tao diem C bang copy constructor, copy diem B
(3.55, 4.77)
Hoanh do cua B: 3.55
Tung do cua B: 4.77
Nhap do tinh tien cho diem B:
-2.12 -3.55
Diem B sau khi tinh tien: (1.43, 1.22)
┌──(~/IT002_OOP/Lab2/Lab2_BTTL/bai1)
└─(21:14:27 on main *)──>
```

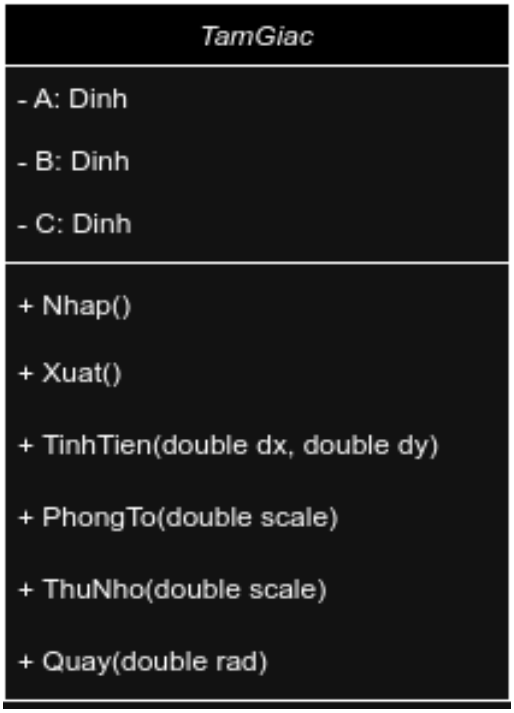
2.2 Bài 2

Bài 2: Xây dựng lớp tam giác:

- Thuộc tính: Đỉnh A, B, C
- Phương thức: Nhap(), Xuat(), TinhTien, PhongTo(), ThuNho(), Quay()

Yêu cầu: Thực hiện xây dựng lớp, vẽ class diagram và khai báo các thuộc tính, phương thức. Viết nội dung vào các phương thức đã khai báo. Gọi các phương thức trong hàm main()

2.2.1 Class Diagram



2.2.2 Input, Output và Solution

Input: Nhập vào 3 đỉnh A, B, C. Nếu sử dụng method PhongTo, ThuNho thì phải nhập scale để phóng to và thu nhỏ. Nếu sử dụng method Quay thì phải nhập góc tính bằng rad

Output: Tọa độ của 3 điểm A, B, C

Solution: Tạo class TamGiac bằng cách tạo 3 instance trong class Diem. Đặt tên là A, B, C. Quay một tam giác thì tức là quay cả 3 điểm cùng 1 góc.

2.2.3 Code

https://github.com/LowTechTurtle/IT002_OOP/tree/main/Lab2/Lab2_BTTL/bai2

2.2.4 Testcase

Log của output:

```
   Nhap toa do diem A: -2.33 4.99
Nhap toa do diem B: 1.213 3.3
Nhap toa do diem C: -4.12 5.17
Toa do tam giac:
A: (-2.33, 4.99)
B: (1.213, 3.3)
C: (-4.12, 5.17)
Nhap lan luot dx va dy de tinh tien: 2.33 4.66

Tam giac sau khi tinh tien:
Toa do tam giac:
A: (0, 9.65)
B: (3.543, 7.96)
C: (-1.79, 9.83)
Nhap scale phong to tam giac: 2.1

Tam giac sau khi phong to:
Toa do tam giac:
A: (0, 20.265)
B: (7.4403, 16.716)
C: (-3.759, 20.643)
Nhap scale thu nho tam giac: 3.2

Tam giac sau khi thu nho:
Toa do tam giac:
A: (0, 6.33281)
B: (2.32509, 5.22375)
C: (-1.17469, 6.45094)
Nhap goc de quay tam giac: 0.6781

Tam giac sau khi quay:
Toa do tam giac:
A: (-3.97267, 4.93178)
B: (-1.46623, 5.52664)
C: (-4.96157, 4.28687)
```

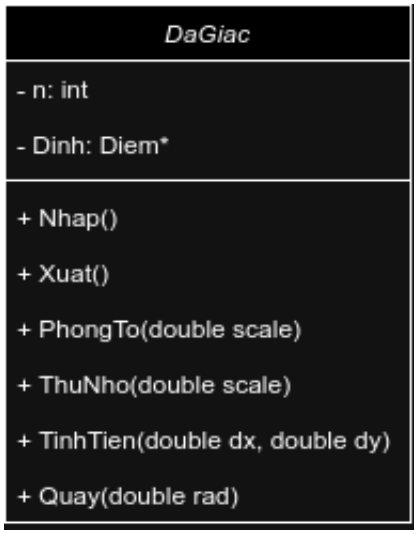
2.3 Bài 3

Bài tập 3: Xây dựng lớp đa giác:

- Thuộc tính: n (số đỉnh đa giác), Diem *Dinh
- Phương thức: Nhap(), Xuat(), TinhTien, PhongTo(), ThuNho(), Quay()

Yêu cầu: Thực hiện xây dựng lớp, vẽ class diagram và khai báo các thuộc tính, phương thức. Viết nội dung vào các phương thức đã khai báo. Gọi các phương thức trong hàm main()

2.3.1 Class Diagram



2.3.2 Input, Output và Solution

Input: Số điểm n và lần lượt hoành độ, tung độ của các điểm. Nếu sử dụng method PhongTo, ThuNho thì phải nhập scale để phóng to và thu nhỏ. Nếu sử dụng method Quay thì phải nhập góc tính bằng rad

Output: Tọa độ của n điểm vừa nhập

Solution: Tạo class DaGiac bằng cách tạo n instance trong class Diem. Quay một đa giác thì tức là quay cả n điểm cùng 1 góc.

2.3.3 Code

https://github.com/LowTechTurtle/IT002_OOP/tree/main/Lab2/Lab2_BTTL/bai3

2.3.4 Testcase

Log của output:

```
   Nhap so dinh cua da giac: 5
Nhap toa do dinh thu 1: -1.2346 3.1234
Nhap toa do dinh thu 2: -3.124 5.6456
Nhap toa do dinh thu 3: 4.342 -2.12
Nhap toa do dinh thu 4: 5.124 2.3123
Nhap toa do dinh thu 5: 2.123 -4.125
Toa do cac dinh cua da giac:
Dinh 1: (-1.2346, 3.1234)

```

```
Dinh 2: (-3.124, 5.6456)

Dinh 3: (4.342, -2.12)

Dinh 4: (5.124, 2.3123)

Dinh 5: (2.123, -4.125)

Nhap lan luot dx va dy de tinh tien:
Da giac sau khi tinh tien:
Toa do cac dinh cua da giac:
Dinh 1: (0.7654, 6.1234)

Dinh 2: (-1.124, 8.6456)

Dinh 3: (6.342, 0.88)

Dinh 4: (7.124, 5.3123)

Dinh 5: (4.123, -1.125)

Nhap scale phong to da giac: 2.31

Da giac sau khi phong to:
Toa do cac dinh cua da giac:
Dinh 1: (1.76807, 14.1451)

Dinh 2: (-2.59644, 19.9713)

Dinh 3: (14.65, 2.0328)

Dinh 4: (16.4564, 12.2714)

Dinh 5: (9.52413, -2.59875)

Nhap scale thu nho dagiac: 4.32

Da giac sau khi thu nho:
Toa do cac dinh cua da giac:
Dinh 1: (0.409276, 3.27432)

Dinh 2: (-0.601028, 4.62299)

Dinh 3: (3.39121, 0.470556)

Dinh 4: (3.80936, 2.8406)

Dinh 5: (2.20466, -0.601562)
```

```
58
59Nhap goc de quay da giac: 0.127
60
61Da giac sau khi quay:
62Toa do cac dinh cua da giac:
63Dinh 1: (-0.00874124, 3.29979)
64
65Dinh 2: (-1.18173, 4.50964)
66
67Dinh 3: (3.3043, 0.896293)
68
69Dinh 4: (3.41889, 3.30022)
70
71Dinh 5: (2.2631, -0.317478)
```

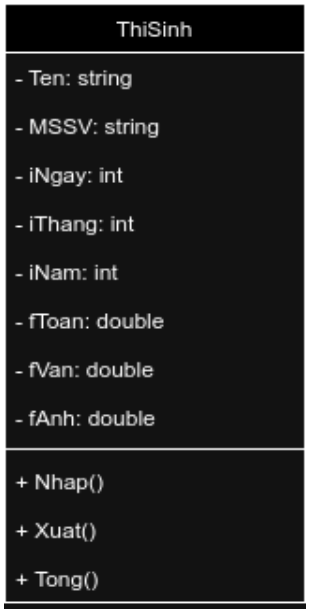
2.4 Bài 4

Bài tập 4: Xây dựng lớp thí sinh:

- Thuộc tính: Ten, MSSV, iNgay, iThang, iNam, fToan, fVan, fAnh
- Phương thức: Nhap(), Xuat(), Tong()

Yêu cầu: Thực hiện xây dựng lớp, vẽ class diagram và khai báo các thuộc tính, phương thức. Viết nội dung vào các phương thức đã khai báo. Trong hàm main(), tạo một mảng nhập vào với n thí sinh, in ra thông tin thí sinh có tổng điểm lớn hơn 15 điểm? In ra thí sinh có điểm cao nhất

2.4.1 Class Diagram



2.4.2 Input, Output và Solution

Input: Các thuộc tính Ten, MSSV, iNgay, iThang, iNam, fToan, fVan, fAnh

Output: Xuất ra các thí sinh có tổng điểm lớn hơn 15 và thí sinh có tổng điểm cao nhất

Solution: Xây dựng class thí sinh gồm có 3 method là Nhap(), Xuat(), Tong(). Lưu các thí sinh vào trong 1 vector và duyệt tuyến tính qua vector đó để tìm thí sinh có tổng điểm $>$ 15 và thí sinh có tổng điểm cao nhất.

2.4.3 Code

https://github.com/LowTechTurtle/IT002_OOP/tree/main/Lab2/Lab2_BTTL/bai4

2.4.4 Testcase

Log của output:

```
   Nhap so thi sinh: 3
Nhap thong tin thi sinh thu 1:
Nhap ten: Nguyen Van A
Nhap MSSV: 123456
Nhap ngay thang nam sinh: 2 3 2000
Nhap diem toan: 7
Nhap diem van: 8
Nhap diem anh: 9
Nhap thong tin thi sinh thu 2:
Nhap ten: Le Van B
Nhap MSSV: 321456
Nhap ngay thang nam sinh: 4 4 2000
Nhap diem toan: 8
Nhap diem van: 4
Nhap diem anh: 5
Nhap thong tin thi sinh thu 3:
Nhap ten: Nguyen Van C
Nhap MSSV: 325612
Nhap ngay thang nam sinh: 8 9 2000
Nhap diem toan: 5
Nhap diem van: 6
Nhap diem anh: 2
Thi sinh co diem lon hon 15
Ten: Nguyen Van A
MSSV: 123456
Ngay sinh: 2/3/2000
Diem Toan: 7
Diem van: 8
Diem Anh: 9
Diem Tong: 24

```

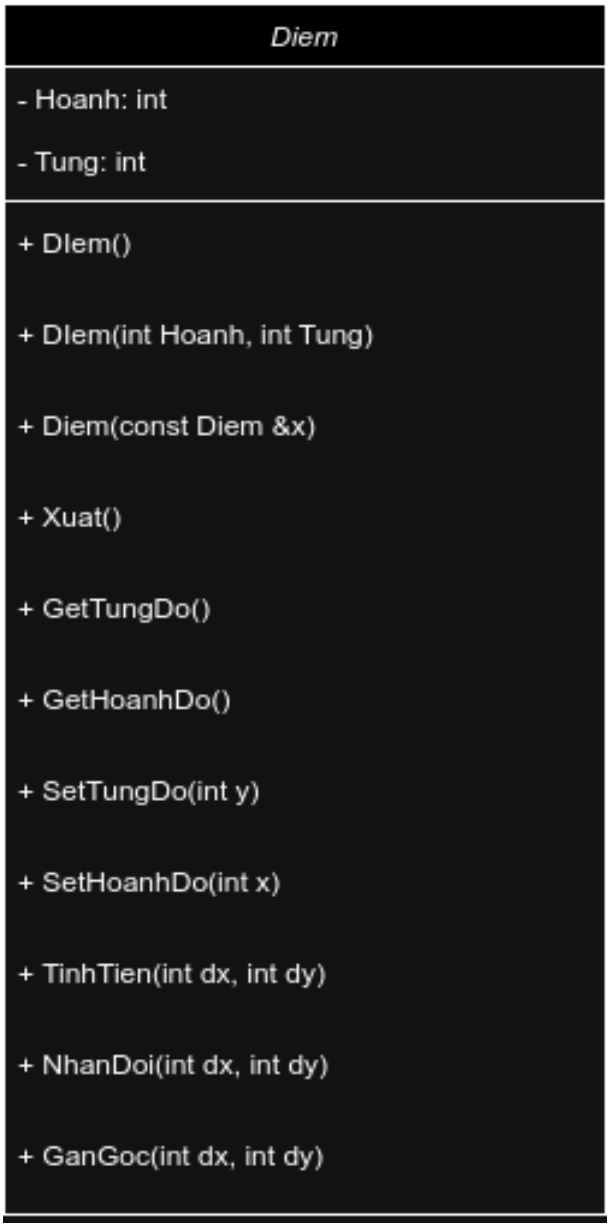
```
Ten: Le Van B
MSSV: 321456
Ngay sinh: 4/4/2000
Diem Toan: 8
Diem van: 4
Diem Anh: 5
Diem Tong: 17

Thi sinh co tong diem cao nhat:
Ten: Nguyen Van A
MSSV: 123456
Ngay sinh: 2/3/2000
Diem Toan: 7
Diem van: 8
Diem Anh: 9
Diem Tong: 24
```

2.5 Bài 5

Bài tập 5: Nhập vào một điểm trong mặt phẳng với hai thành phần là hoành độ và tung độ. Sau đó cho phép người dùng nhập n là số lượng chỉ thị cho chương trình, yêu cầu nhập giá trị các chỉ thị x.

1. Nếu x = 1 thì nhân đôi tung độ và hoành độ
2. Nếu x = 2 thì gán điểm về gốc tọa độ
3. Nếu x = 3 thì yêu cầu người dùng nhập hướng tịnh tiến k (k = 0 tịnh tiến theo trục x, k khác 0 tịnh tiến theo trục y) và độ tịnh tiến d.
4. Nếu x khác 1,2,3 thì không làm gì cả. Sau khi thực hiện hết chỉ thị thì thoát chương trình. Cuối cùng là xuất ra thông tin điểm dưới dạng (a,b)

2.5.1 Class Diagram



2.5.2 Input, Output và Solution

Input: nhập n là số lượng chỉ thị cho chương trình, yêu cầu nhập giá trị các chỉ thị x.

Output: xuất ra thông tin điểm dưới dạng (a,b).

Solution: Dựa theo đề bài để chọn mã cho các chỉ thị, sau đó gọi các hàm tương ứng như NhanDoi, GanGoc,... cho các chỉ thị. Các hàm này tương đối đơn giản nên không giải thích dài dòng ở đây.

2.5.3 Code

https://github.com/LowTechTurtle/IT002_OOP/tree/main/Lab2/Lab2_BTTL/bai5

2.5.4 Testcase

```
┌─(~/IT002_OOP/Lab2/Lab2_BTTL/bai5)─
└─(21:40:01 on main ✱)──> ./main
4 6
3
1
2
3 0 6
(6, 0)
┌─(~/IT002_OOP/Lab2/Lab2_BTTL/bai5)─
└─(21:40:11 on main ✱)──> ./main
-3 6
3
1
3 -2 3
1
(-12, 30)
┌─(~/IT002_OOP/Lab2/Lab2_BTTL/bai5)─
└─(21:41:39 on main ✱)──> ./main
-2 2
4
1
1
1
3 -1 -1
(-16, 15)
```

2.6 Bài 6

Bài tập 6: Class Tam giác Chúng ta cho người dùng nhập vào tọa độ 3 điểm của tam giác, bao gồm 6 biến (x1, y1, x2, y2, x3, y3) trên cùng một dòng. Sau đó nhập hướng tịnh tiến (đơn vị là độ - gốc tọa độ là trục dương x) và độ dài tịnh tiến. Cuối cùng là xuất ra thông tin 3 điểm của tam giác sau khi được tịnh tiến.

Lưu ý: chọn kiểu dữ liệu cho các điểm của tam giác là float để test case không bị sai do thừa phần thập phân. Chọn PI = 3.14

2.6.1 Class Diagram

2.6.2 Input, Output và Solution

Input: Nhập vào tọa độ 3 điểm của tam giác, bao gồm 6 biến (x1, y1, x2, y2, x3, y3) trên cùng một dòng.

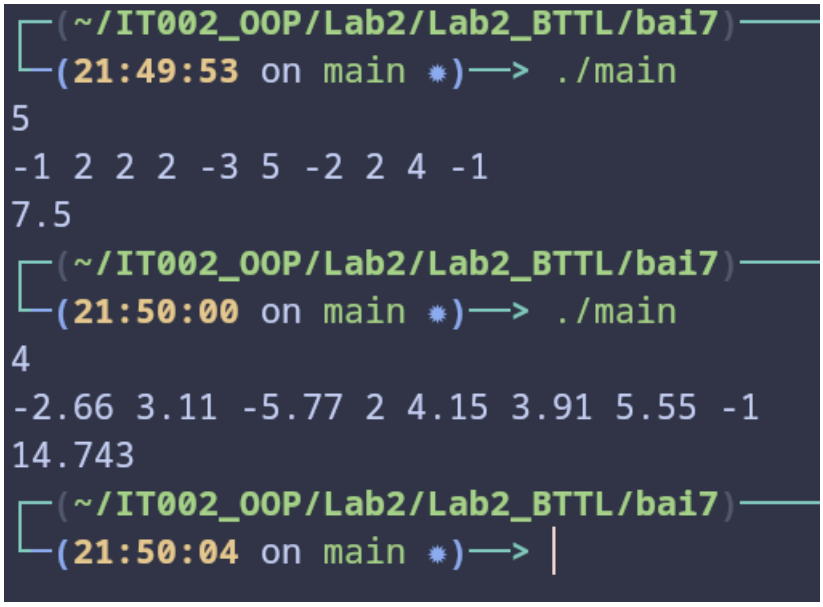
Output: xuất ra thông tin 3 điểm của tam giác sau khi được tịnh tiến.

Solution: Sử dụng lại class tam giác ở bài 2, gọi hàm TinhTien để cho tịnh tiến các điểm phù hợp theo yêu cầu.

2.6.3 Code

https://github.com/LowTechTurtle/IT002_OOP/tree/main/Lab2/Lab2_BTTL/bai6

2.6.4 Testcase



2.7 Bài 7

Bài tập 7: Class Polygon Nhập vào thông tin của một đa giác bao gồm số điểm n và thông tin n tọa độ trên n dòng. Sau đó xuất ra diện tích của hình này.

Lưu ý: số điểm phải lớn hơn 2, bởi vì 2 điểm trở lên thì mới tạo thành một đa giác có diện tích và đa giác ở đây luôn là đa giác lồi hoặc lõm.

2.7.1 Class Diagram

2.7.2 Input, Output và Solution

Input: Nhập vào thông tin của một đa giác bao gồm số điểm n và thông tin n tọa độ trên n dòng

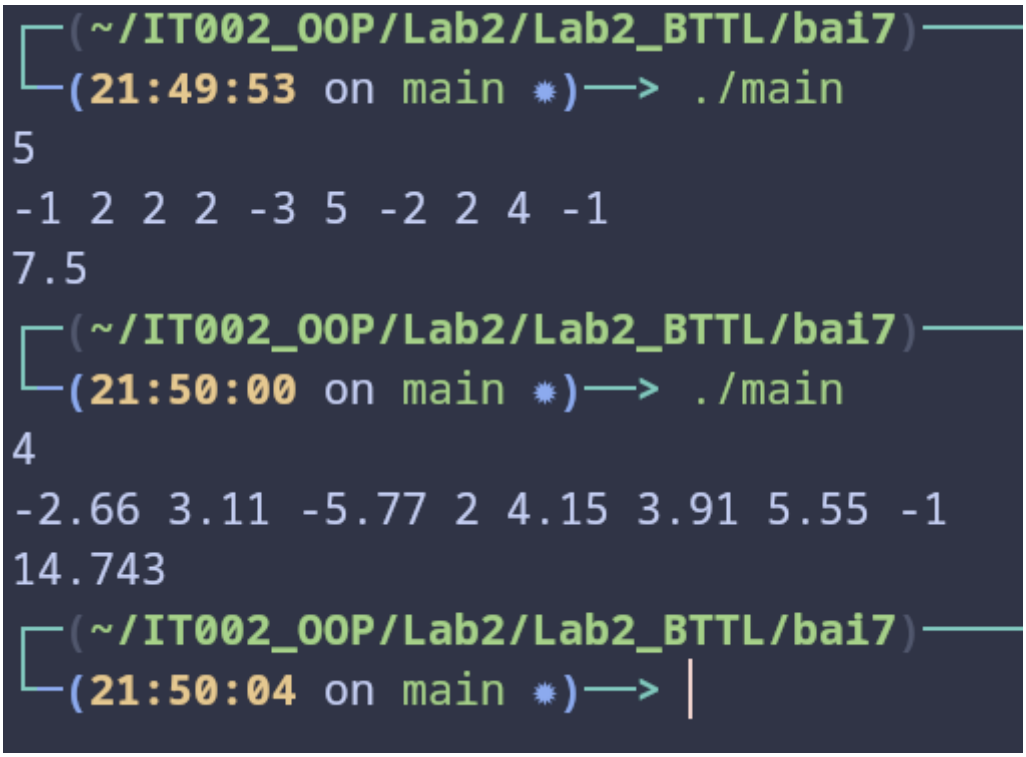
Output: Xuất ra diện tích của hình này

Solution: Sử dụng công thức Shoelace để tính diện tích của đa giác.

2.7.3 Code

https://github.com/LowTechTurtle/IT002_OOP/tree/main/Lab2/Lab2_BTTL/bai7

2.7.4 Testcase



2.8 Bài 8

Bài tập 8: Class List và vấn đề con trỏ List là một class, kiểu dữ liệu đã được đóng gói sẵn trong C++, list có tác dụng quản lý mảng dễ dàng hơn so với việc khai bóa thông thường, ví dụ muốn xóa phần tử có giá trị 5 trong mảng, ta phải viết một vòng for để tìm; trong khi đó chỉ cần list.pop(5) là xong. List gồm 2 thuộc tính là một con trỏ quản lý mảng (*double) và một biến size (unsigned int) để truy xuất kích thước list.

1. Chúng ta khởi tạo list rỗng.
2. Cho người dùng nhập vào chỉ thị n.
   - Nếu n = -1 thì nhảy đến B3.
   - Nếu n = 0 thì cho người dùng nhập x và thêm x vào list.
   - Nếu n = 1 thì cho người dùng nhập x và xóa phần tử đầu tiên có giá trị x ra khỏi list.
   - Nếu n = 2 thì cho người dùng nhập x và xóa tất cả phần tử có giá trị x ra khỏi list.
   - Nếu n = 3 thì cho người dùng nhập x, y và thay đổi phần tử thứ x bằng y, nếu x không hợp lệ thì không làm gì cả. Quay trả lại B2.
3. In ra màn hình list hiện tại theo mẫu [a,b,c,d,...].
4. Kết thúc chương trình.

Lưu ý: nên hiểu rõ cách hoạt động của list và sau này chúng ta sẽ sử dụng list rất nhiều.

2.8.1 Input, Output và Solution

Input: Cho người dùng nhập vào chỉ thị n, Sau đó nhập tiếp vào các giá trị tương ứng với mỗi chỉ thị(theo đề bài ở trên)

Output: In ra màn hình list hiện tại theo mẫu [a,b,c,d,...].

Solution: Sử dụng List trong STL của C++. Sử dụng iterator và hàm erase để xóa một phần tử tại một vị trí được chỉ tới bởi iterator. Sử dụng hàm remove(x) để xóa tất cả các giá trị x có trong list. Sử dụng hàm advance(it, x) cho iterator để cho iterator chạy qua 1 đoạn là x(x là số nguyên)

2.8.2 Code

https://github.com/LowTechTurtle/IT002_OOP/tree/main/Lab2/Lab2_BTTL/bai8

2.8.3 Testcase

```
└─(21:53:34 on main ✹)──> ./main
0 1
0 2
0 6
2 2
3 1 4
-1
[1,4]
┌─(~/IT002_OOP/Lab2/Lab2_BTTL/bai8)─
└─(21:53:46 on main ✹)──> ./main
0 3
0 8
0 2
0 7
0 1
0 2
0 2
1 2
-1
[3,8,7,1,2,2]
┌─(~/IT002_OOP/Lab2/Lab2_BTTL/bai8)─
└─(21:54:17 on main ✹)──> ./main
0 2
0 1
0 1
0 3
0 1
3 3 1
2 1
-1
[2]
┌─(~/IT002_OOP/Lab2/Lab2_BTTL/bai8)─
└─(21:55:07 on main ✹)──>
```