Lập trình hướng đối tượng

Mục tiêu

1. Tổ chức lớp ra 2 tập tin riêng biệt .h và .cpp
2. Cài đặt thuộc tính và các getter, setter tương ứng
3. Cài đặt hàm tạo không đối, hàm hủy
4. Cài đặt hàm tạo với đối số

n

1 Hướng dẫn khởi

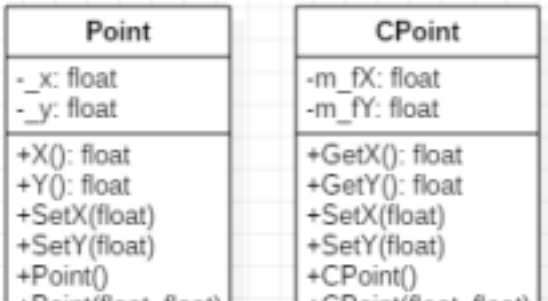
đầu

Mô tả bài tập

Cho trước thiết kế lớp **Điểm** trong không gian hai chiều với 2 thuộc tính **x** và **y.**

Hãy cài đặt cụ thể lớp này với các thành phần:

+ Thuộc tính



+ Các hàm getter setter tương ứng

+ Hàm tạo và hàm hủy

+ Hàm tạo có đối số

Thiết kế tinh gọn và theo kí pháp Hungary

- Tạo ra định nghĩa lớp như sau:

```
class Point {
private:
    float _x;
    float _y;
public:
    float X() { return _x; }
    float Y() { return _y; }
    void SetX(float value) { _x = value; }
    void SetY(float value) { _y = value; }
public:
    Point();
    Point(float, float);
    ~Point();
};
```

Đúng ra getter và setter là các hàm nên cần được cài đặt ở trong tập tin .cpp. Tuy nhiên riêng ở trong tình huống này quá đơn giản, thuần túy return và gán giá trị nên ta có thể viết tắt, đặt cài đặt getter / setter ngay bên trong tập tin .h luôn. Nhưng nên nhớ cách đúng là nên đặt ở file .cpp cho cài đặt, file .h chỉ dành để định nghĩa lớp.

Bước 3: Cài đặt lớp trong file Point.cpp (CPoint.h)

```
#include "Point.h"

Point::Point() {
    this->_x = 0;
    this->_y = 0;
}

Point::Point(float x, float y) {
    this->_x = x;
    this->_y = y;
}

Point::~Point() {

}
```

Chú ý:

- Hiện tại hàm hủy chưa có cài đặt gì nên nó đang rỗng

- Con trỏ this dùng để tham chiếu đến đối tượng gọi hàm hiện tại, có thể không cần cũng được nếu không có nhập nhằng về đặt tên biến

Bước 4: Cài đặt hàm main để test việc cài đặt của lớp Point (CPoint)

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

#include "Point.h"

int main()
{
    Point root; // Test hàm tạo không đối
    cout << "Root: " << root.X() << ", " << root.Y() << endl;

    Point node(7, 12); // Test hàm tạo có đối số
    cout << "Node: " << node.X() << ", " << node.Y() << endl;

    Point* p = new Point(12, 198); // Test việc cấp phát thủ công
    cout << "Node: " << p->X() << ", " << p->Y() << endl;
    delete p; // Tự thu hồi vùng nhớ
}
```

Chạy lên và thấy kết quả như sau:

```
Root: 0, 0
Node: 7, 12
Node: 12, 198
```

2 Bài tập vận dụng

Yêu cầu

1. Sử dụng StarUML thiết kế sơ đồ lớp, chụp hình kết quả nộp lại. 2. Thực hiện định nghĩa lớp theo thiết kế cho trước vào tập tin .h. 3. Thực hiện cài đặt lớp trong tập tin .cpp cho lớp tương ứng. Cần đảm bảo có tối thiểu các thành phần sau:

    + Getter / Setter cho các thuộc tính private

    + Hàm tạo không đối, Hàm tạo có đối số

    + Hàm hủy (trước mắt có thể không có cài đặt)

4. Viết các đoạn mã nguồn kiểm tra việc định nghĩa lớp trong hàm main.

Danh sách bài tập cụ thể

1. Lớp **Đường thẳng** có hai thành phần **Điểm**: **Bắt đầu** và **Kết thúc**. Gợi ý:

Tên project: LineV1

    + Tên lớp: Line / CLine

    + Thành phần: _start, _end

2. Lớp **Hình chữ nhật** có hai thành phần **Điểm**: **Trái trên** và **Phải Dưới** Gợi ý:

Tên project: RectangleV1

    + Tên lớp: Rectangle / CRectangle

+ Thành phần: _topLeft, _bottomRight

3. Lớp **Hình tam giác** có ba thành phần **Điểm** ứng với 3 đỉnh : **a, b, c**. Gợi ý:

Tên project: TriangleV1

+ Tên lớp: Triangle / CTriangle

+ Thành phần: _a, _b, _c

4. Lớp **Hình tròn** có 2 thành phần: **tâm** (Lớp **Điểm**) và **bán kính** (số thực). Gợi ý:

Tên project: CircleV1

+ Tên lớp: Circle / CCircle

+ Thành phần: _center, _radius

5. Lớp **Phân số** có 2 thành phần: **tử** (số nguyên) và **mẫu** (số nguyên) Gợi ý:

Tên project FractionV1

+ Tên lớp: Fraction / CFraction

+ Thành phần: _num, _den

(Tử: Numerator, Mẫu: Denominator)

6. Lớp **Sinh viên** có 3 thành phần: **họ** (chuỗi), **tên lót** (chuỗi) và **tên** (chuỗi). Gợi ý: Tên project StudentV1

+ Tên lớp: Student / CStudent

+ Thành phần: _firstName, _middleName, _lastName. Kiểu dữ liệu: string. Cần include <string>

3 Hướng dẫn nộp bài

Tổ chức bài nộp

+ Mỗi bài tương ứng với 1 folder. Chỉ chứa file .h và .cpp và Makefile + Tổng hợp tất cả folder vào folder MSSV.

+ Nén lại tất cả thành một tập tin duy nhất MSSV.zip.

Để nộp bài, nén tất cả lại và đặt tên với định dạng **MSSV.zip -- HẾT --**