**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN**

**BÀI TẬP LỚN**

**LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG**

|  |  |
| --- | --- |
| **Giáo viên hướng dẫn**  **Nguyễn Ngọc Long** | **Nhóm Sinh viên thực hiện:**  **Nguyễn Bạch Ngọc My**  **Mã số SV: 18110151**  **Bùi Thị Thanh Phương**  **Mã Số SV: 20110280**  **Lớp: 20TTH\_TN** |

**Thành Phố Hồ Chí Minh, ngày 14 tháng 6 năm 2022**

Mục Lục

I. Nội dung đề tài và lời nói đầu …………………………………………………………………… 3

1. Lời mở đầu …………………………………………………………………………………………………………..3

2. Nội dung đề tài………………………………………………………………………………………………………3

II. Tạo sơ đồ lớp………………………………………………………………………………………………4

III. Tạo đối tượng lớp……………………………………………………………………………………….5

1.Lớp điểm (point)………………………………………………………………………………………………………5

2. Tạo đối tượng shape………………………………………………………………………………..……………..6

3. Lớp hình Ellipse…………………………………………………………………………………….…………………6

1. Tạo lớp Ellipse…………………………………………………………………………………………………………………………7
2. Lớp hình tròn (Circle)……………………………………………………………………………………………………………….7
3. Kiểm tra một điểm nằm trong hình Ellipse………………………………………………………………………………7

4. Lớp hình đa giác (Polygon)…………………………………………………………………………..…………8

1. Lớp hình đa giác …………………………………………………………………………………………………………………….8
2. Lớp hình tam giác, hình chữ nhật và hình vuông……………………………………………………………………9
3. Lớp hình bán nguyệt và hình trăng khuyết……………………………………………………..…………………….9
4. Kiểm tra một điểm nằm trong hình đa giác……………………………………………………………..……………9

IV. Thuật toán tìm phần giao và in ra dấu chữ thập bên trong phần giao……11

1. Thuật toán tìm phần giao giữa 2 đối tượng bất kỳì………………………………………………………11
2. In ra chữ dấu thập phân trong phần giao…………………………………………………………………11

V. Lập trình trong môi trường windows……………………………………………..……..……11

1. Vẽ hình……………………………………………………………………………………………………………………11
2. Các thao tác trong môi trường Windows…………………………………………………….…….……12
   1. Thao tác di chuyển đối tượng bằng cách kéo thả chuột ……………………………..………12
   2. Thao tác thay đổi đối tượng …………………………………………………………………………….13
   3. Thao tác phóng to thu nhỏ……………………………………………………………………………..13

**I. Nội dung đề tài và lời mở đầu**

**1. Lời mở đầu**

Để hoàn thành được đề tài bài tập lớn này, Đầu tiên em xin cảm ơn đến Trường Đại Học Khoa học Tự nhiên đã đưa môn Lập Trình Hướng Đối Tượng vào chương trình dạy, đặt biệt nhóm chúng em xin gửi lời cảm ơn sâu sắc nhất đến **Thầy Nguyễn Ngọc Long,** giảng viên môn học Lập trình hướng đối tượng, vì sự tận tình giảng dạy và hướng dẫn nhóm chúng em trong suốt thời gian học vừa qua. Từ những kiến thức mà thầy truyền tải đã giúp chúng em hoàn thành đồ án cuối kì này. Mặc dù chúng em đã cố gắng hết sức nhưng sẽ không tránh những thiếu sót, chúng em mong thầy xem xét và mong nhận được sự đóng góp ý kiến của thầy để bài đồ án được hoàn thiện hơn.

**2.Nội dung đề tài**

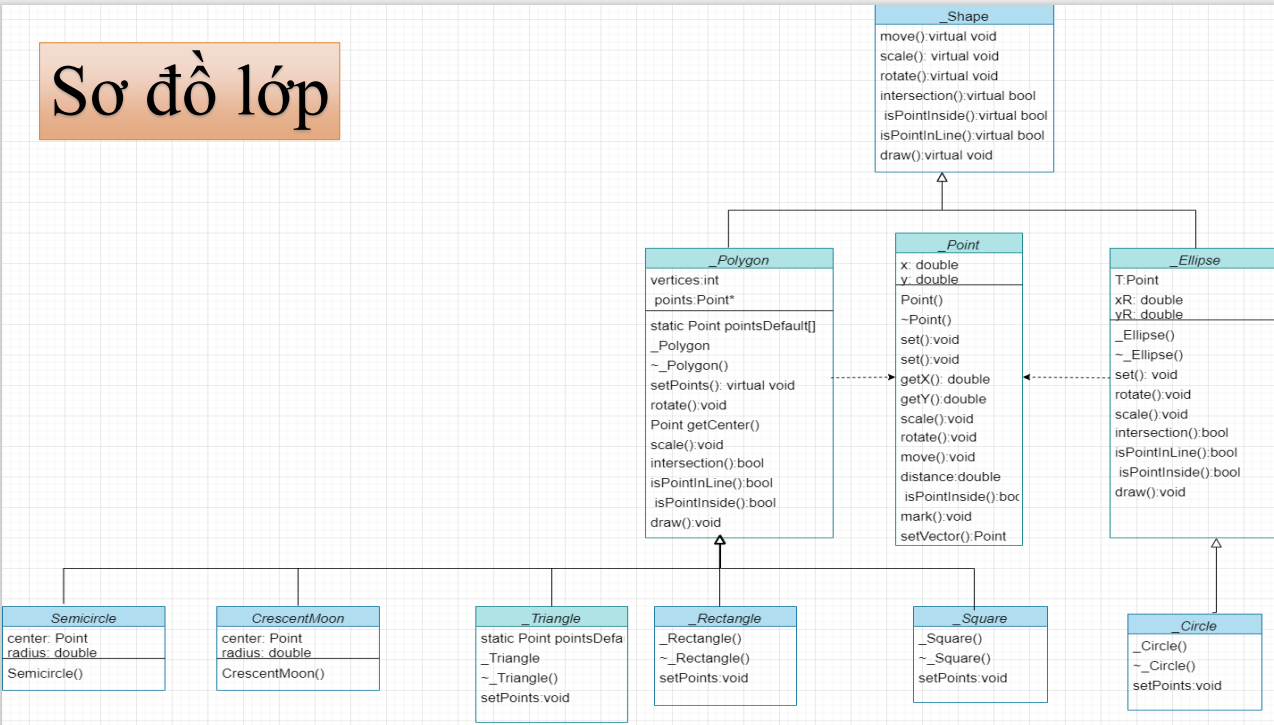
Cho các loại đối tượng: hình tròn, hình ellipse, hình bán nguyệt, đa giác, hình chữ nhật, hình vuông, hình tam giác.

• Viết chương trình ứng dụng cho phép tạo (hoặc nhập) hai hình thuộc một trong các hình kể trên.

• In thông báo cho biết hai hình có giao nhau không, nếu có thì vẽ một dấu chữ thập ở bên trong phần giao.

• Người sử dụng có thể bấm các phím mũi tên để di chuyển một trong hai hình, phím +, -, để phóng to thu nhỏ một trong hai hình. Lập trình trong môi trường Windows.

**II.Sơ đồ lớp**

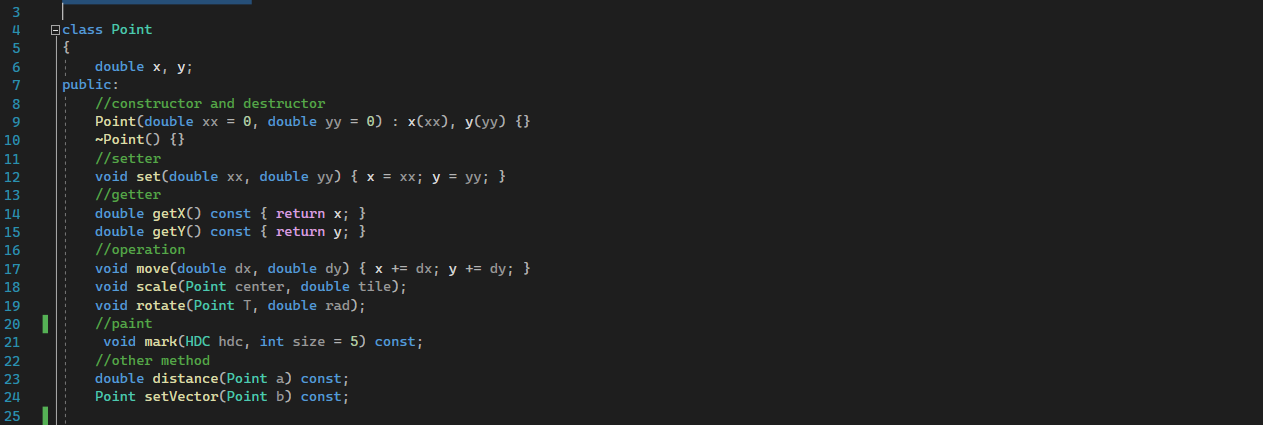


Giải thích:

* Point là lớp lưu thông tin về tọa độ một điểm trong hệ tọa độ 2 chiều.
* x, y là các thuộc tính chỉ hoành độ, tung độ của đối tượng lớp Point.
* Point() là phương thức khởi tạo không tham số
* Point(x: double, y: double) là phương thức khởi tạo có tham số dùng để khởi tạo giá trị cho thuộc tính x (hoành độ) và y (tung độ).
* setX(x: double), getX(), setY(y: double), getY() là các setter và getter mà bạn đã biết.
* setXY(x: double, y:double) là phương thức dùng để thay đổi giá trị cho thuộc tính x và y.
* distance(x: double, y:double) là phương thức trả về khoảng cách giữa đối tượng hiện tại và điểm có hoành độ x, tung độ y.
* distance(another: Point) là phương thức trả về khoảng cách giữa đối tượng hiện tại và đối tượng another (another cũng là một đối tượng thuộc lớp Point).

**III. Tạo đối tượng lớp**

**1.Lớp điểm (point)**



Trong đó:

* Point(double xx = 0, double yy = 0) : x(xx), y(yy) {}

~Point() {}

Sử dụng constructor and destructor để khởi tạo đối tượng và hủy đối tượng

* void move(double dx, double dy) { x += dx; y += dy; }

void scale(Point center, double tile);

void rotate(Point T, double rad);

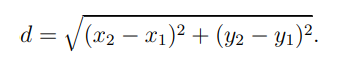
Dùng để di chuyển điểm đến tọa độ mới,sau đó tịnh tiến sau đó thực hiện các thao tác tịnh tiến, vị tự, phép quay tâm T (đơn vị độ) của một điểm.

* void mark(HDC hdc, int size = 5) const;

Dùng thực hiện đánh dấu (+) một điểm trên cửa sổ ứng dụng.

double distance(Point a) const;

Dùng để tính khoảng cách giữa hai điểm. Khoảng cách giữa hai điểm A(x1, y1) và B(x2, y2) được tính thông qua công thức

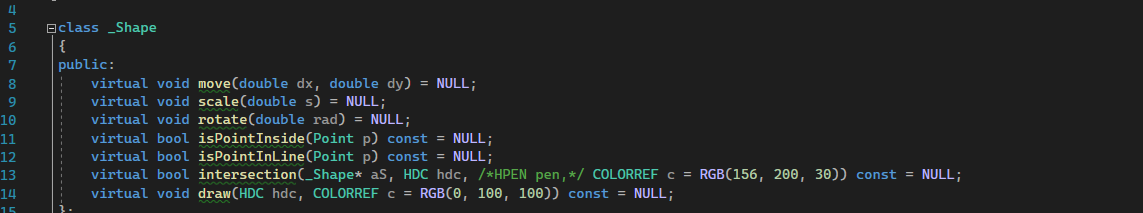


* Point setVector(Point b) const;

Dùng để lấy vector giữa hai điểm. Vector AB được xác định bằng:

Vector(AB) = (xB − xA; yB − yA).

**2. Tạo đối tượng shape**



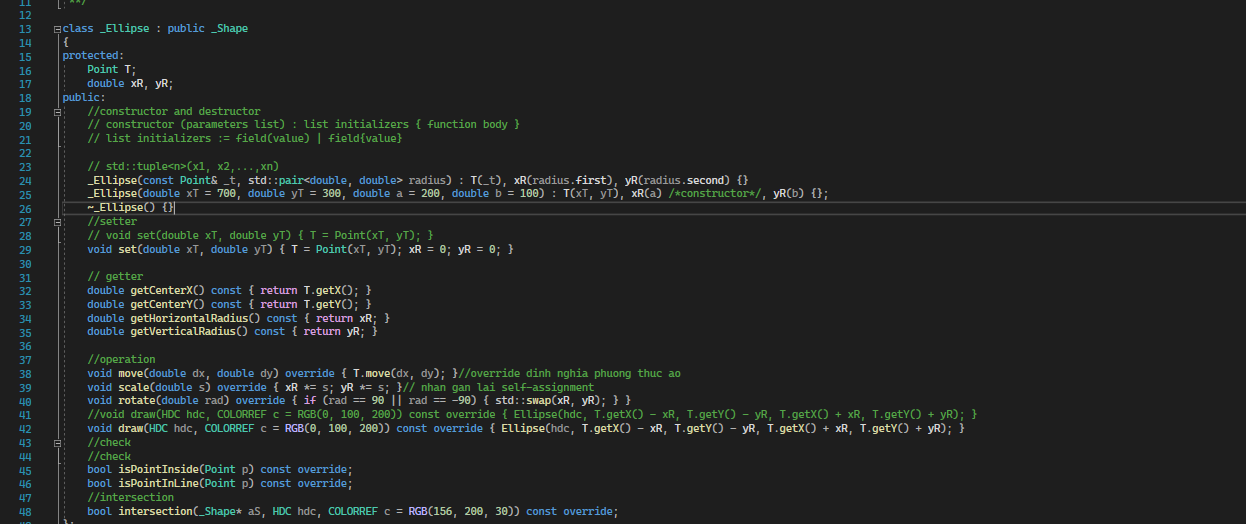
Trong đó:

* virtual bool intersection(\_Shape\* aS, HDC hdc, /\*HPEN pen,\*/ COLORREF c = RGB(156, 200, 30)) const = NULL;

Dùng để thực hiện việc kiểm tra hai đối tượng hình học đang xét có giao nhau hay không. Với phương thức này, ta có thuật toán tổng quát tìm phần giao cho hai đối tượng hình học bất kỳ.

**3.Lớp hình Ellipse**

**a. Tạo lớp Ellipse**



Trong đó:

* class \_Ellipse : public \_Shape

lớp \_Ellipse được kế thừa từ lớp cha (\_Shape), khi đó **\_Ellipse** được thừa hưởng các đặt tính lớp **\_Shape**

* \_Ellipse(const Point& \_t, std::pair<double, double> radius) : T(\_t), xR(radius.first), yR(radius.second) {}

\_Ellipse(double xT = 700, double yT = 300, double a = 200, double b = 100) : T(xT, yT), xR(a) ,yR(b) {};

~\_Ellipse() {}

Sử dụng constructor and destructor để khởi tạo đối tượng (**Ellipse**) hủy đối tượng

* void move(double dx, double dy) override { T.move(dx, dy); }

tịnh tiến hình **Ellipse** bằng cách tịnh tiến tâm T của hình.

* void scale(double s) override { xR \*= s; yR \*= s; }

Chia tỷ lệ 2 trục đồng thời thực hiện phóng to thu nhỏ

* void rotate(double rad) override { if (rad == 90 || rad == -90) { std::swap(xR, yR); } }

thực hiện quay hình **Ellipse**  với góc quay có đơn vị độ. ở phương thức này chỉ cho phép quay hình với góc 90◦

* bool isPointInside(Point p) const override;

kiểm tra một điểm nằm bên trong hình **Ellipse** (không tính phần biên).

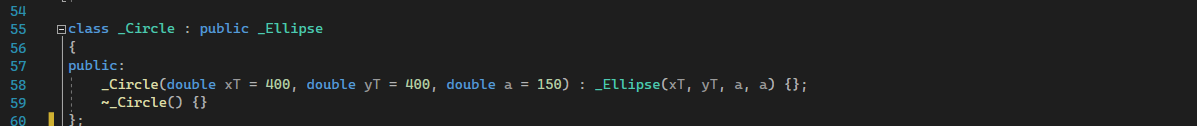
* bool isPointInLine(Point p) const override;

là phương thức kiểm tra một điểm nằm trên đường biên của hình **Ellipse**.

* bool intersection(\_Shape\* aS, HDC hdc, COLORREF c = RGB(156, 200, 30)) const override;

kiểm tra hình Ellipse và một đối tượng hình học giao nhau.

**b. lớp hình tròn (Circle)**



Được thừa kế từ hình Ellipse đồng thời cũng thực hiện phương thức tạo và hủy đối tượng

**c. Kiểm tra một điểm nằm trong hình Ellipse**

Giả sử hình **Ellipse** có tâm là T(x0 ,y0 ) và bán kính ngang, bán kính dọc lần lượt là a, b

• Khi đó, phương trình **Ellipse** là:

• Muốn xác định một điểm P(xp ,yp ) có nằm trong **Ellipse** hay không ta thay tọa độ của P vào phương trình **Ellipse.**

• Tính hiệu của giá trị vừa nhận được và 1.

• Nếu hiệu < một số epsilon cho trước đủ nhỏ thì cho kết quả P nằm trong hình **Ellipse (isPointInside)**

• Nếu giá trị tuyệt đối của hiệu < một số epsilon đủ nhỏ cho trước thì cho kết quả P nằm trên biên của hình **Ellipse (isPointInline)**

**4. Lớp hình đa giác (Polygon)**

**a. Lớp hình đa giác**

Trong đó:

* class \_Polygon : public \_Shape

**Polygon** được thừa kế từ lớp **Shape**

* static Point pointsDefault[];

\_Polygon(int num\_vers = 4, Point\* p = NULL) : vertices(num\_vers) { setPoints(num\_vers, p); }

~\_Polygon() { delete[] points; }

Sử dụng constructor and destructor để khởi tạo đối tượng (**Polygon**) hủy đối tượng

* void move(double dx, double dy) override;

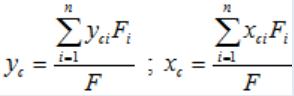
di chuyển hình đa giác theo vector (dx, dy) bằng cách di chuyển từng đỉnh của hình đa giác

* void scale(Point Center, double s);

Thực hiện phóng to thu nhỏ hình

* void rotate(Point Center, double rad);

Thực hiện quay hình (góc quay có đơn vị độ) đa giác theo tâm của hình đa giác, với toạ độ tâm được xác định bằng cách lấy trung bình toạ độ của các đỉnh.



* bool isPointInside(Point p) const override;

kiểm tra một điểm nằm bên trong hình đa giác (không tính phần biên).

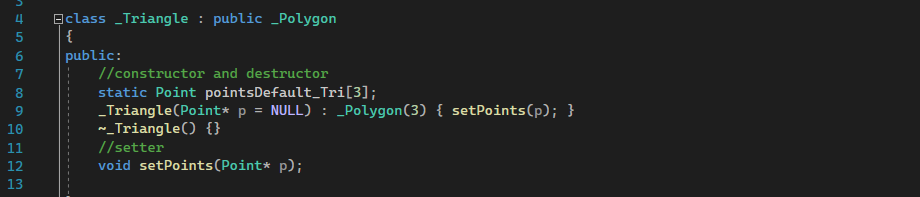
* bool isPointInLine(Point p) const override;

là phương thức kiểm tra một điểm nằm trên đường biên của hình đa giác

* bool intersection(\_Shape\* aS, HDC hdc, COLORREF c = RGB(156, 200, 30)) const override;

kiểm tra hình đa giác và một đối tượng hình học giao nhau.

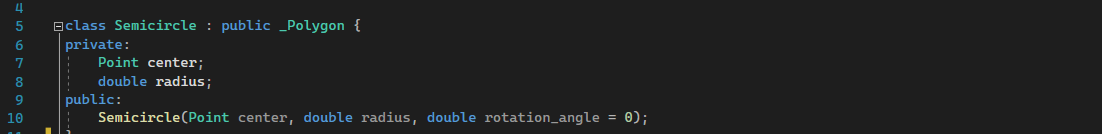
1. **Lớp hình tam giác, hình chữ nhật và hình vuông**



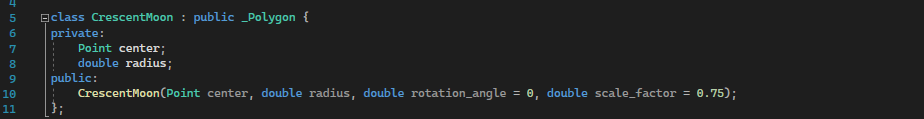
Lớp tam giác được thừa kế từ lớp hình đa giác

Tương tự hình chữ nhật (Rectangle) và hình vuông (Square) cũng được thừa kế từ lớp đa giác (Polygon)

1. **Lớp hình bán nguyệt và hình trăng khuyết**



Cũng được thừa kế từ lớp hình đa giác



1. **Kiểm tra một điểm nằm trong hình đa giác**

Cho một điểm P(xp,yp) trong mặt phẳng xác định điểm P có nằm trong đa giác (C) hay không?

*Thuật toán:*

Kẻ 1 đường thẳng (d) đi qua điểm P, các điểm nằm trên đường thẳng (d) có cùng hoành độ xp

Trên đường thẳng (d), kẻ tia có góc tại P và hướng theo chiều tăng của tung độ

Đếm số giao điểm của tia với các cạnh của đa giác, nếu tia đi qua 1 đỉnh của đa giác thì ta tính tia đó các hai cạnh là của đỉnh đó

A picture containing text, clock, watch

Description automatically generated

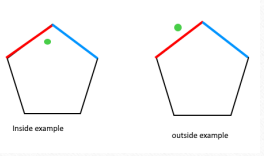
Nếu giao điểm là số chẳn, ta kết luận điểm P nằm bên ngoài đa giác (C)

Nếu số giao điểm là số lẻ, ta kết luận điểm P nằm bên trong đa giác (C)

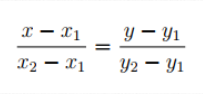
*Bài toán con:*

* Giả sử 2 đỉnh của một cạnh (nào đó) trong đa giác (Polygon) là P1(x1,y1) và

P2(x2,y2). Xác định tia *(Px)* cắt P1P2 hay không?



• Khi đó, phương trình đường thẳng của P1P2 là:



* Cho x bằng xp ta tính được y, nếu y thuộc đoạn [y1,y2] thì tia *Px* cắt P1P2

• Muốn kiểm tra một điểm P có nằm trên cạnh P1P2 của đa giác hay không, chúng ta tính hiệu số giữa: (khoảng cách từ P đến P1 + khoảng cách từ P đến P2) và (khoảng cách từ P1 đến P2)



• So sánh giá trị tuyệt đối của hiệu số trên

• Nếu giá trị này nhỏ hơn một số epsilon đủ bé cho trước thì trả về kết quả điểm P nằm trên cạnh của đa giác, ngược lại kết quả là P không nằm trên cạnh của đa giác.

**IV. Thuật toán tìm phần giao giữa 2 đối tượng bất kỳ và in ra chữ thập phân bên trong phần giao**

**1. Thuật toán tìm phần giao giữa 2 đối tượng bất kỳ**

• Để tìm phần giao giữa hai đối tượng bất kỳ, ta sẽ xét các điểm trên hình chữ nhật nhỏ nhất bao phủ một trong hai đối tượng.

• Dùng phương thức isPointInside và isPointInline để xác định một điểm có nằm trong hoặc nằm trên biên hình đang xét hay không.

• Nếu điểm đang xét thuộc cả ba trường hợp sau thì ta sẽ dùng hàm SetPixel() để tô màu điểm đó: • Nếu điểm nằm trên biên của đối tượng 1 và nằm bên trong đối tượng 2

• Nếu điểm nằm bên trong đối tượng 1 và nằm trên biên của đối tượng 2

• Nếu điểm nằm trên biên của cả hai đối tượng.

• Sau khi xét từng hình chữ nhật ta sẽ có phần giao cần tìm với phần viền được tô đậm.

**2. In ra chữ dấu thập phân trong phần giao**

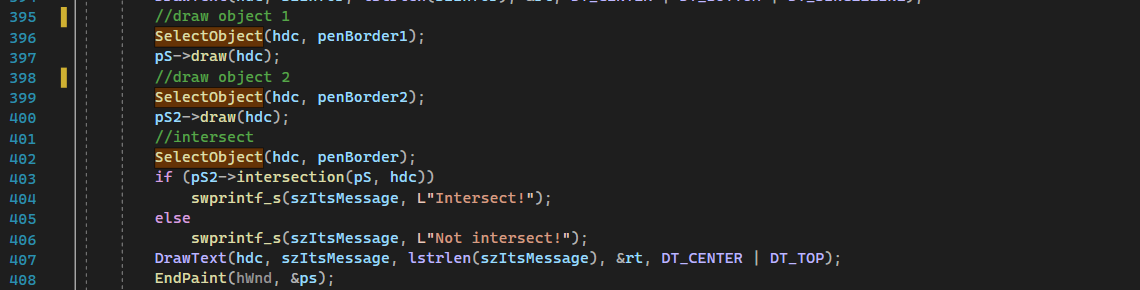
Để vẽ dấu cộng ở phần giao của hai hình bất kì, trước tiên chúng ta xác định phần giao của hai hình bằng phương thức intersection

• Khi đã có phần giao rồi, tìm hình chữ nhật nhỏ nhất bao phủ phần giao

• Sau đó, vẽ dấu cộng tại tâm hình chữ nhật bao phủ bằng phương thức mark.

**V. Lập trình trong môi trường windows**

**1. Vẽ hình**



Trong đó:

Đối tượng 1 màu viền là penBorder1

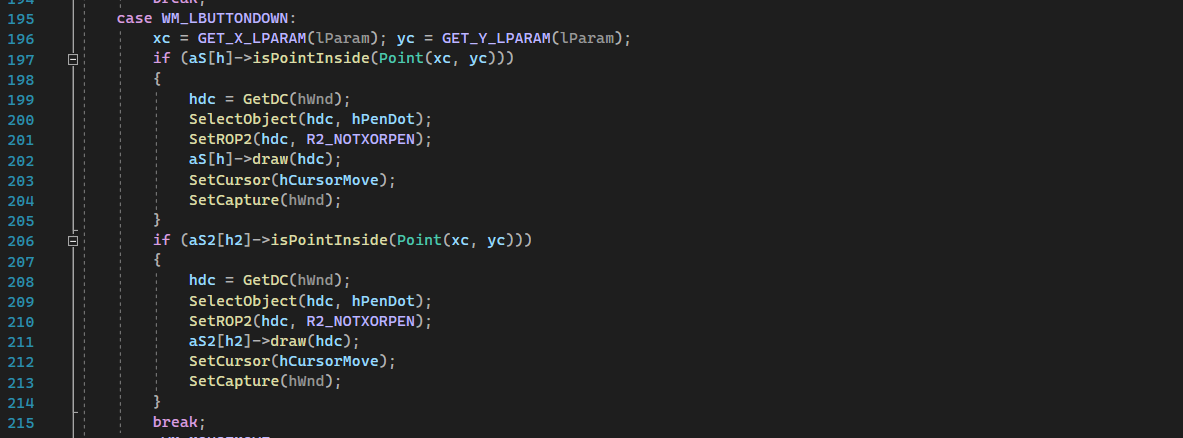
Đối tượng 2 màu viền là penBorder2

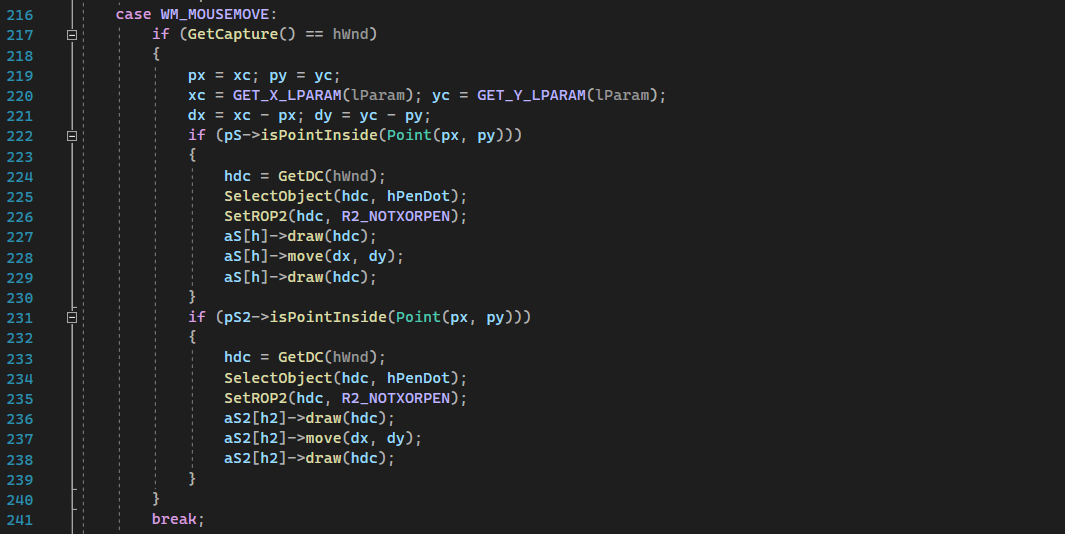
Tìm phần giao giữa 2 đối tượng sẽ xuất ra viền penBorder

**2. Các thao tác trong môi trường Windows**

**a. Thao tác di chuyển đối tượng bằng cách kéo thả chuột**

Khi kéo thả chuột thì vị trí một trong 2 hình sẽ thay đổi và hình ở vị trí ban đầu sẽ xóa và thay thế cho hình ở vị trí mới.

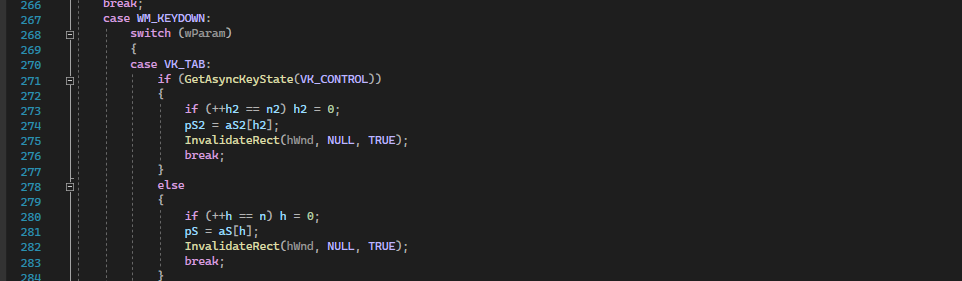




**b. Thao tác thay đổi đối tượng**

để thay đổi hình thứ 1(màu xanh dương) ta dùng phím Tab trên bàn phím

để thay đổi hình thứ 2( màu hồng) ta dung Tab + Ctrl



**c.Thao tác phóng to thu nhỏ**

Để phóng to ta sử dụng phím (+) và để thu nhỏ ta sử dụng phím (-)

