Viết một chương trình C/C++ như sau:

a) Có sẵn hai cấu trúc dữ liệu **point** (với hai trường **double x, y**) và **lineSegment** (với hai trường **point p, q**)

b) đọc file **input.txt** có cấu trúc:

dòng đầu là **N** biểu thị số lượng đa giác,

Dòng tiếp theo là số **M** biểu thị số đỉnh của đa giác thứ nhất. **M** dòng tiếp theo nữa là biểu thị tọa độ x, y của các đỉnh thuộc đa giác thứ nhất (x cách y bởi dấu cách - space)

Rồi tiếp tục lặp lại với các đa giác còn lại

Chẳng hạn các dòng dưới đây mô tả 2 đa giác, mỗi đa giác có 4 đỉnh.

2

4

2 6

2 0

0 0

0 6

4

5 0

5 6

7 6

7 0

Giả sử rằng

(i) các cạnh của đa giác luôn song song với một trong hai trục tọa độ.

(ii) nếu hai đỉnh có cùng tung độ y thì hoành độ của chúng chênh nhau một giá trị là bội số của 2

(iii) nếu hai đỉnh có cùng hoành độ x thì tung độ của chúng chênh nhau một giá trị là bội số của 6

(iv) các đỉnh của đa giác trong file đã được sắp xếp sao cho chiều di chuyển từ đỉnh ở dòng i (trong file) xuống dòng (i+M - 1) (trong file) là chiều kim đồng hồ

(v) N >= 1 và M >= 4

(vi) Các đa giác có thể là lồi hoặc lõm.

(vii) không có đỉnh nào xuất hiện hai lần trong số các đỉnh của một đa giác. Vẫn có thể có một đỉnh xuất hiện vài lần trong số các đỉnh của các đa giác khác nhau

Không cần phải viết mã kiểm tra xem input có thỏa mãn các điều kiện trên hay không

**Yêu cầu:**

Sau khi đọc xong file, chương trình sẽ có được hai **vector** như sau:

**vector<point> points** là danh sách các đỉnh của tất cả các đa giác

**vector<vector<lineSegment>>** **polygons** là tập hợp tất cả các **lineSegment** (cạnh) của tất cả các đa giác. Nhớ rằng **polygons.at(i)** là tập hợp tất cả các **lineSegment** của đa giác thứ i. Các **lineSegment** cũng phải được bố trí sao cho

**polygons.at(i).at(j-1).q = polygons.at(i).at(j).p**

**polygons.at(i).at(j).q = polygons.at(i).at(j+1).p**

chiều quay từ đỉnh p của **polygons.at(i).at(0)** đến đỉnh q của **polygons.at(i).at(M-2)** là chiều kim đồng hồ

c) Cho phép người dùng nhập vào một điểm **I(x, y)**, viết hàm kiểm tra xem điểm **I** này có phải là trọng tâm của một hình chữ nhật ABCD (rộng 2, dài 6) sao cho hình này:

(i) có ít nhất một cạnh của ABCD nằm trên cạnh của một và chỉ một đa giác (ở câu (b))

(ii) toàn bộ mọi điểm của hình chữ nhật đều thuộc đa giác

(iii) một đỉnh của hình chữ nhật ABCD nếu cùng hoành độ hoặc tung độ với một đỉnh của đa giác (có chứa ABCD) thì đều thỏa mãn điều kiện (ii), (iii) của câu (b).

Giả sử rằng đỉnh A là đỉnh cao nhất bên tay trái, đỉnh B là đỉnh cao nhất bên tay phải, đỉnh C là đỉnh thấp nhất bên tay phải và D là đỉnh thấp nhất bên tay trái.

d) Hỏi người dùng có muốn khoét đi hình chữ nhật có trọng tâm **I** hay không (biết **I** phải thỏa mãn các điều kiện trong câu (c)). Nếu có hãy:

(i) cập nhật lại **points** và in ra màn hình các đỉnh này

(ii) cập nhật lại **polygons** và in ra màn hình số đa giác và các đỉnh của mỗi đa giác

(iii) tạo ra file output.txt có định dạng như input.txt

Ở đây cần hiểu rằng việc khoét đi hình chữ nhật ABCD có trọng tâm I có thể khiến:

1. Đa giác chứa ABCD có số đỉnh tăng lên hoặc giảm đi hoặc không đổi
2. Số lượng đa giác có thể tăng lên hoặc giảm đi hoặc không đổi

Cố gắng làm xong trước 08/08, viết mã nguồn trong sáng, các hàm đều có comment mô tả tham số đầu vào và kết quả đầu ra. Các biến và hàm định nghĩa tên tường minh.

Yêu cầu code theo C/C++ chuẩn (tối đa C++17) chạy được trên Ubuntu Linux và Cygwin của Windows.

Mã nguồn cần được đưa vào sonarqube để kiểm định chất lượng

Khi thanh toán thì người làm có thể demo online qua Team hoặc Zoom, người làm nhập input theo yêu cầu.

Hoàn thành xong sẽ gửi file chạy a.out (hoặc a.exe) để nhận 50% qua chuyển khoản,

Nhận xong 50% đầu tiên khi đó sẽ gửi toàn bộ code và sẽ nhận tiếp 50% còn lại

Đơn giá 1 triệu đồng

Liên hệ: [nguyenthanh@soict.hust.edu.vn](mailto:nguyenthanh@soict.hust.edu.vn)