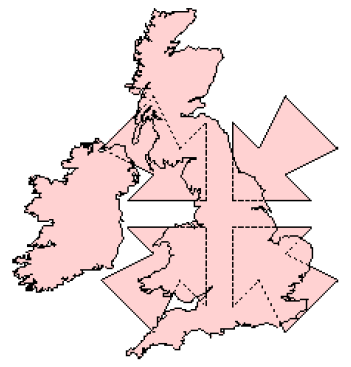
Algorithmic and C Programming - LO27

Project: 2D Polygon Library



###### Summary

[I. Main goal of the project: 3](#_Toc408164592)

[II. Explanations: 3](#_Toc408164593)

[1. Data structure: 3](#_Toc408164594)

[Type Point 3](#_Toc408164595)

[Type Polygon 3](#_Toc408164596)

[Type PolyList 4](#_Toc408164597)

[2. Mains functions : 4](#_Toc408164598)

[Polygon unionPolygons(Polygon poly1, Polygon poly2) 4](#_Toc408164599)

[Polygon intersectionPolygons(Polygon p1, Polygon p2) 4](#_Toc408164600)

[PolyList exclusiveORPolygons(Polygon p1, Polygon p2) 4](#_Toc408164601)

[PolyList differencePolygons(Polygon p1, Polygon p2) 4](#_Toc408164602)

[Polygon convexHullPolygon(Polygon poly) 5](#_Toc408164603)

[III. Difficulties and optimisations: 5](#_Toc408164604)

[1. Difficulties: 5](#_Toc408164605)

[2. Optimisations: 5](#_Toc408164606)

[IV. Conclusion: 5](#_Toc408164607)

# Main goal of the project:

The main goal of this project was to provide a library of functions for manipulating 2D-polygons.We had to provide a program that allows a user to interactively test every single functions provided by this library.

The library written in C contains the types and functions that we are going to describe all along this report.

# Explanations:

## Data structure:

### Type Point

The type Point represents a 2D-Point (x,y) in a real two dimensional space. x and y are doubles that represent the coordinates of the point.

### Type Polygon

The type Polygon represents a general polygon as a circular doubly linked list of Points. This list is designed as a datatype that maintains an access to the first point and to the number of points of the polygon. The considered polygons can be convex or concave but they can’t contain any hole.

### Type PolyList

The type PolyList represents a group of polygons as a linked list of datatype Polygon. This list is designed to always maintain an access to the next polygon. This kind of list is used in the function exclusiveORPolygons that return several polygons or in the main program.

## Mains functions :

In this part we are going to explain to you the running of the mains functions of our library. We only explain the complex parts of the algorithms of these functions that is to say without the trivial cases.

### Polygon unionPolygons(Polygon poly1, Polygon poly2)

Trivial cases:

General case:

### Polygon intersectionPolygons(Polygon p1, Polygon p2)

The behavior of intersectionPolygons is really close to unionPolygon, only some small differences occurs in the trivial cases or in the traversal of the two polygons.

### PolyList exclusiveORPolygons(Polygon p1, Polygon p2)

Even if this function seems hard to to, we realised that with two differents call of the function differencePolygons, the result was the same as excepted.

### PolyList differencePolygons(Polygon p1, Polygon p2)

Trivial cases:

General case:

### Polygon convexHullPolygon(Polygon poly)

Trivial cases:

General case:

# Difficulties and optimisations:

## Difficulties:

## Optimisations:

With a dedicated function for exclusiveORPolygon, we probably could save some computation and be more efficient.

# Conclusion:

Apports du projet

* Plus de rigueur dans nos programmes
* L'importance des commentaires pour que celui qui est avec nous comprenne aussi
* Programmation modulaire : réutilisation de fonctions dans plusieurs sous-programmes
* …