**Sprawozdanie**

**Opis problemu**

Celem zadania było zaimplementowanie algorytmu wyznaczania położenia punktu (odszukiwania wielokąta zawierającego dany wierzchołek) na płaszczyźnie 2D przy podziale planarnym odcinków.

By algorytm wykonywał się efektywnie konieczne było skorzystanie z odpowiednich struktur danych.

Struktury jakie wykorzystaliśmy to mapa trapezowa T, reprezentowana przez strukturę powiązań sąsiedzkich konkretnych elementów (trapezów) oraz graf przeszukiwań. Zastosowany graf posiadał 3 rodzaje węzłów, NodeVertex, NodeSegment oraz NodeArea, reprezentujące odpowiednio wierzchołek, odcinek oraz ścianę (trapez). Każdy węzeł w grafie posiadał dowiązania do odpowiadającego mu elementu na mapie trapezowej. Węzły NodeVertex oraz NodeSegment zawsze posiadały dokładnie dwójkę dzieci (wierzchołki wewnętrzne), natomiast węzły NodeArea nie posiadały dzieci, będąc zawsze liściami.

Istotną częścią zadania było poprawne identyfikowanie sąsiadów każdego trapezu, ze względu na zastosowanie położenia ogólnego odcinków każdy trapez mógł zawierać maksymalnie 4 sąsiadów. Mimo tak niewielkiej ilości sąsiadów istniało dużo konfiguracji ich występowania, co należało odpowiednio rozważyć przy dodawaniu kolejnych odcinków.

**Testy**

Aby zweryfikować poprawność działania algorytmu przeprowadzono szereg testów weryfikujących działanie programu. Testy te miały szereg zadań, m.in. weryfikację poprawności tworzenia nowych trapezów, weryfikację poprawności tworzenia i edytowania mapy trapezowej oraz grafu przeszukiwania. Testy były wprowadzane przez interfejs graficzny użytkownika, zawierały różne konfiguracje odcinków o różnym stopniu złożoności. Poniżej znajduje się kilka przeprowadzonych testów oraz wynik (wizualizacja) jaki otrzymano.

TUTAJ BĘDZIE SPOKOJNIE

**Wyniki oraz wnioski**

Algorytm dokonuje poprawnego określenia pozycji zadanego punktu w podziale planarnym, co zweryfikowano z pomocą dołączonej do algorytmu wizualizacji kolejnych etapów działania algorytmu (zarówno tworzenia samej mapy trapezowej jak i wyszukiwania na niej zadanych wierzchołków). Sama implementacja również wpisuje się w założenia co do użytej pamięci oraz złożoności obliczeniowej algorytmu (odpowiednio O(n) oraz O(nlogn) dla tworzenia mapy trapezowej oraz O(logn) dla lokalizacji punktu).