# Zaawansowane Techniki Programowania Zad 3 Solid

## Klasa Solver

Klasa będąca rozszerzeniem HttpServlet

## Metody:

- collectDataFromDB

Opis :

Funkcja nawiązuje połączenie z bazą danych, a następnie pobiera wszystkie punkty z bazy danych i zapisuje w liście.

Dane wejściowe:

- String tableName - nazwa tabeli z punktami

Dane wyjściowe:

- List<float[]> - wszystkie punkty z bazy danych

- doGet

Opis :

Metoda obsługuje zapytanie get. Pobiera z parametrów zapytania parametr t, odpowiadający za nazwę tabeli w bazie danych Funkcja wywołuje główny algorytm i pobiera wynik końcowy który następnie jest wysyłany jako odpowiedź na zapytanie.

Dane wejściowe:

- HttpServletRequest request - zmienna przechowująca parametry zapytania GET
- HttpServletResponse response - zmienna odpowiedzialna za odpowiedź z servletu

Dane wyjściowe:

- void

# Klasa Point

Klasa zawierająca informacje o punkcie.

## Pola:

- float x -współrzędna x
- float y - współrzędna y
- float z - współrzędna z

## Metody:

- Point(float x, float y, float z)
- getX
- getY
- getZ

# Klasa Block

Klasa będąca implementacja interfejsu IBlockRemote

## Metody:

- calculateLateralSurface

Opis :

Funkcja główna wywołująca funkcje odpowiedzialne za obliczenie obwodu podstawy i wysokości figury. Następnie zwraca pole boczne figury.

Dane wejściowe:

- List<float[]> points - wszystkie punkty z bazy danych

Dane wyjściowe:

- float - pole boczne figury

- calculateHeight

Opis :

Metoda wylicza średnia arytmetyczna współczynników z punktów która stanowi wysokość figury.

Dane wejściowe:

- List<Point> points - lista wszystkich punktów

Dane wyjściowe:

- float - wysokość figury

- clockOrientation

Opis :

Metoda sprawdza orientacje trzech punktów czy jest zgoda lub nie z wskazówkami zegara lub czy punkty leżą w jednej linii.

Dane wejściowe:

- Point p
- Point q
- Point r

Dane wyjściowe:

- int - orientacji

- getHullPoints

Opis :

Metoda wyszukuje za pomocą algorytmu Jarvisa punkty leżące na brzegu otoczki

Dane wejściowe:

- List<Point> points - lista wszystkich punktów

Dane wyjściowe:

- Vector<Points> - punkty będące otoczka podstawy

- calculatePolygonCircuit

Opis :
Funkcja oblicza obwód podstawy

Dane wejściowe:
- Vector<Points> points - lista punktówy budujach wielokat będącego podstawą figury

Dane wyjściowe:
- float - obwód podstawy

## Interface IBlockRemote
Interfejs modułu Block

## Metody:
- calculateLateralSurface

## Interface IDSManagerRemote
Interfejs dostarczający informacje o bazie danych

## Metody:
- getDS

# Opis algorytmu

W celu wyliczenia pola bocznego figury nalezy pomnozyc wysokość figury razy obwód jej podstawy. Wysokość została obliczona jako średnia arytmetyczna współczynników “z” punktów.

W celu wyznaczenia podstawy figury został zastosowany algorytm Jarvisa . Na początku algorytmy wyznaczany jest punkt z największym współczynnikiem x-

jest on dodawany do listy jako punkt startowy. Następnie w pętli sprawdzane są wszystkie pozostałe punkty z którymi poprzedni punkt z listy tworzy prosta, tak aby wszystkie pozostałe punkty leżały po lewej stronie od prostej powstałej miedzy tymi dwoma punktami. W tym celu dla dla dwóch punktów tworzących prosta za pomocą wzoru :

```
public int clockOrientation(Point p, Point q, Point r) {
   float temp = (q.getY() - p.getY()) * (r.getX() - q.getX()) -
(q.getX() - p.getX()) * (r.getY() - q.getY());

   if (temp == 0) return 0;
   return (temp > 0) ? 1 : 2;
}
```

s

Sprawdzane jest czy punkt r leży po lewej stronie(wynik dodatni) od prostej lub po prawej(wynik ujemny) - jeżeli wynik wynosi 0 to wszystkie 3 punkty leżą w jednej prostej. Jeżeli położenie punktu r jest niezgodne z zasadą wskazówek zegara to pozostaje podmiana punktu q na r.  Dla kombinacji tych dwóch punktów następuje dalsze przeszukiwanie - ostatni punkt q zostaje dodany do listy i cała operacja się powtarza aż wrócimy z powrotem do punktu startowego.

Gdy wszystkie punkty tworzące wielokąt podstawy znajdują się w liście następuje obliczenie obwodu podstawy jako odległości między kolejno ułożonymi punktami w liście.

Na koniec obwód podstawy zostanie przemnożony przez wysokość figury co daje pole powierzchni bocznej.