Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования

Пермский национальный исследовательский политехнический университет

Электротехнический факультет

Кафедра информационных технологий и автоматизированных систем

**ОТЧЕТ**

**Тема:** Лабораторная работа «задача коммивояжера»

Семестр: 2

Выполнил студент ИВТ-23-2б:

Синицын Владислав Сергеевич

(дата, подпись)

Проверила:

Полякова Ольга Андреевна

(дата, подпись)

Пермь 2024

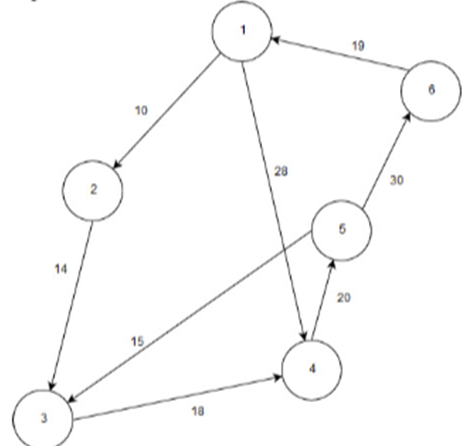
**Задание**

-задача коммивояжера

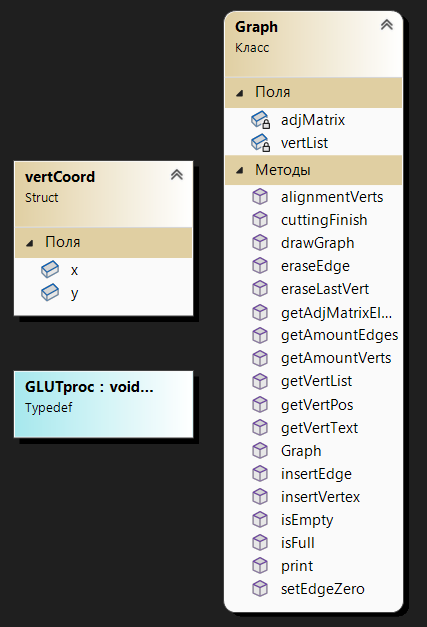
- интерфейса

- текстового сопровождения

- вертикальная печать

****

**UML-Диаграмма**



**Программное решение**

**Main.cpp**

#include "Graph\_.h"

#include <Windows.h>

#include <GL\glew.h>

#include <GL\freeglut.h>

#include <iostream>

int main(int argc, char\* argv[]) {

setlocale(LC\_ALL, "Russian");

makeGraph();

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_DEPTH | GLUT\_DOUBLE | GLUT\_RGBA);

glutInitWindowSize(WinW, WinH);

glutCreateWindow("Графы");

glClearColor(0.8, 0.8, 0.8, 0.8);

WinW = glutGet(GLUT\_WINDOW\_WIDTH);

WinH = glutGet(GLUT\_WINDOW\_HEIGHT);

glutDisplayFunc(display);

glutReshapeFunc(reshape);

glutMotionFunc(mouseMove);

glutPassiveMotionFunc(mouseMove);

glutMouseFunc(mouseClick);

glutMainLoop();

return 0;

}

**Graph.h**

#ifndef GRAPH\_GRAPH\_H

#define GRAPH\_GRAPH\_H

#include <Windows.h>

#include <iostream>

#include <vector>

#include <iomanip>

#include <string>

using namespace std;

const int maxSize = 20; // Максимальное количество вершин

extern int WinW; // Ширина окна

extern int WinH; // Высота окна

// Структура координат вершни

struct vertCoord {

int x, y;

};

class Graph

{

private:

vector<int> vertList; // Вектор вершин

int adjMatrix[maxSize][maxSize]; // Матрица смежности

public:

Graph(); // Конструктор

bool isEmpty(); // Проверка на пустоту

bool isFull(); // Проверка на полноту

int getVertText(int i); // Получить значение вершины

vector<int> getVertList(); // Получить вектор вершин

int getAdjMatrixElem(int i, int j); // Получить элемент матрицы смежности

int getAmountVerts(); // Получить количество вершин графа

int getVertPos(const int& vertex); // Получить позицию вершины в векторе вершин

int getAmountEdges(); // Получить количество ребер графа

void setEdgeZero(int i, int j); // Занулить элемент матрицы (для Коммивояжера)

void insertVertex(const int& vertex); // Добавить вершину (в конец вектора вершин)

void insertEdge(const int& vertex1, const int& vertex2, int weight); // Добавить ребро весом weight между vertex1 и vertex2

void print(); // Вывод матрицы смежности в консоль

void eraseLastVert(); // Удалить последнюю вершину в векторе вершин

void eraseEdge(const int& vertex1, const int& vertex2); // Удалить ребро между vertex1 и vertex2

void drawGraph(); // Нарисовать граф в OpenGL

void cuttingFinish(); // Закончили резать

void alignmentVerts(); // Выравниваем вершины

};

extern Graph graph; // Создается объект класса Graph

/\*

// Функции для решения задачи Коммивояжера

\*/

void printMatrix(int\*\* matrix); // Вывод текущей матрицы

bool checkSalesman(int\*\* matrix); // Проверка: можно ли решить задачу Коммивояжера

int\*\* Change\_Matrix(); // Создается матрица, заменяя 0 на -1

int\* searchMinElem(int\* line, int n); // Поиск минимального элемента в строке матрицы

int\*\* reductMatrix(int\*\* oldmatrix); // Редукция матрицы

int\*\* highZero(int\*\* oldmatrix); // Нуль с наибольшим потенциалом

void printResult(); // Вывод результата решения Коммивояжера

/\*

// Рисование кнопок OpenGL

\*/

void drawButton1(); // Кнопка "Make new graph"

void drawButton2(); // Кнопка "Salesman Task"

void drawButton3(); // Кнопка "Print Matrix In Console"

void drawButton4(); // Кнопка "Add Vertex"

void drawButton5(); // Кнопка "Delete Vertex"

void drawButton6(); // Кнопка "Add Edge"

void drawButton7(); // Кнопка "Delete Edge"

void drawButton11(); // Кнопка "Draw path"

void drawInformation(); // Значение радиуса в окне OpenGL

/\*

// Рисование элементов графа OpenGL

\*/

void drawCircle(int x, int y, int R, bool aim); // Рисование круга (aim - навелись на кружок)

void drawText(int nom, int x1, int y1); // Рисование веса ребра

void drawLine(int text, int x0, int y0, int x1, int y1, int start, int end); // Рисование ребра

void drawVertex(int n); // Рисование вершины

void drawPath(int n); // Рисование пути Коммивояжера

void drawCreateEdge(); // Рисуем линию для путей

void drawCutting(); // Рисуем линию для "отрезания"

void makeGraph(); // Создание графа

void setCoords(int i, int n); // Установка координат для вершин графа

/\*

// Функции - проверка

\*/

int checkCircle(int x, int y); // Проверка курсора на кружке

void checkButton(int x, int y); // Проверка курсора на кнопке

/\*

// Функции отслеживания поведения мыши

\*/

void mouseMove(int x, int y); // Отслеживание движения курсора

void mouseClick(int button, int state, int x, int y); // Отслеживание нажатия клавиш на мыши

/\*

// Изображение

\*/

void reshape(int w, int h); // Изменение размера экрана

void display(); // Создание изображения

/\*

// Вспомогательные

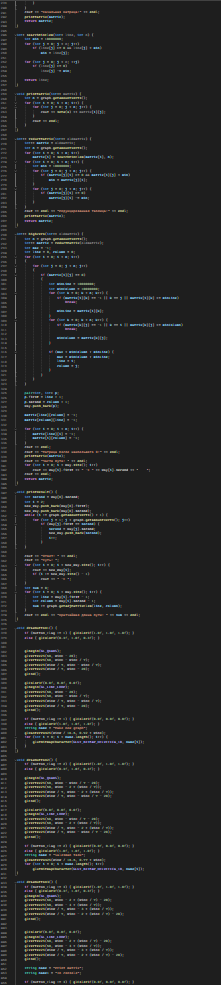
\*/

bool intersect(vertCoord a1, vertCoord a2, vertCoord b1, vertCoord b2); // Проверка на пересечение двух отрезков

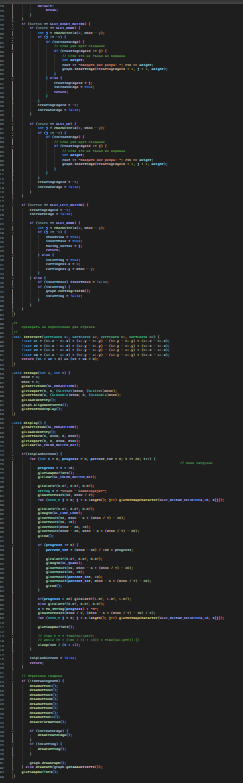
#endif // GRAPH\_GRAPH\_H

**Graph.cpp**

****

****

**…**

****