**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по учебной практике**

Тема: Алгоритм Флойда-Уоршелла

|  |  |  |
| --- | --- | --- |
| Студент гр. 5303 |  | Нигай А.С. |
| Студентка гр. 5303 |  | Матвеева А.И. |
| Студентка гр. 5303 |  | Табунникова Н.Р. |
| Руководитель |  | Фирсов М.А. |

Санкт-Петербург

2017

**ЗАДАНИЕ**

**на учебную практику**

|  |  |  |
| --- | --- | --- |
| Студент Нигай А.С. группы 5303 | | |
| Студентка Матвеева А.И. группы 5303 | | |
| Студентка Табунникова Н.Р. группы 5303  Тема практики: Алгоритм Флойда-Уоршелла | | |
| Задание на практику:  Командная итеративная разработка визуализатора алгоритма на Java с графическим интерфейсом.  Алгоритм: Флойда-Уоршелла. | | |
| Сроки прохождения практики: 21.06.2017 – 04.07.2017 | | |
| Дата сдачи отчета: 30.06.2017 | | |
| Дата защиты отчета: 30.06.2017 | | |
|  | | |
| Студент |  | Нигай А.С. |
| Студентка |  | Матвеева А.И. |
| Студентка |  | Табунникова Н.Р. |
| Руководитель |  | Фирсов М.А. |

**Аннотация**

Целью работы было создание визуализатора алгоритма Флойда-Уоршелла на Java с графическим интерфейсом. В результате была написана программа, выполняющая следующие задачи:

* Отображение графа по введенной матрице смежности / списку инцидентности;
* Визуализация работы алгоритма Флойда – Уоршелла;
* Отображение кратчайшего пути между двумя заданными вершинами;
* Генерация нового графа;
* Сохранение матриц смежности и инцидентности в файл и чтение матрицы инцидентности из него.

Программа выполняет проверки на корректность входных данных и существование в графе циклов отрицательного веса.

**содержание**

|  |  |  |
| --- | --- | --- |
|  | Введение | 5 |
| 1. | Требования к программе | 6 |
| 1.1. | Исходные требования к программе | 6 |
| 1.2. | Уточнение требований после сдачи прототипа | 6 |
| 1.3. | Уточнение требований после сдачи 1-ой версии | 7 |
| 2. | План разработки и распределение ролей в бригаде | 8 |
| 2.1. | План разработки | 8 |
| 2.2. | Распределение ролей в бригаде | 8 |
| 3. | Особенности реализации | 9 |
| 3.1. | Использованные структуры данных | 9 |
| 3.2. | Основные методы | 9 |
| 4. | Тестирование | 12 |
|  | Заключение | 14 |
|  | Список использованных источников | 15 |
|  | Приложение А. Исходный код программы | 16 |

**введение**

Целью работы было создание визуализатора алгоритма Флойда-Уоршелла на Java с графическим интерфейсом.

Алгоритм Флойда-Уоршелла — алгоритм для нахождения кратчайших расстояний между всеми вершинами взвешенного [графа](http://ru.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D1%84_(%D0%BC%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)) без циклов с отрицательными весами. Алгоритм легко модифицировать таким образом, чтобы он возвращал не только длину кратчайшего пути, но и сам путь.

На вход программе подаётся взвешенный ориентированный [граф](https://neerc.ifmo.ru/wiki/index.php?title=%D0%9E%D1%81%D0%BD%D0%BE%D0%B2%D0%BD%D1%8B%D0%B5_%D0%BE%D0%BF%D1%80%D0%B5%D0%B4%D0%B5%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F:_%D0%B3%D1%80%D0%B0%D1%84,_%D1%80%D0%B5%D0%B1%D1%80%D0%BE,_%D0%B2%D0%B5%D1%80%D1%88%D0%B8%D0%BD%D0%B0,_%D1%81%D1%82%D0%B5%D0%BF%D0%B5%D0%BD%D1%8C,_%D0%BF%D0%B5%D1%82%D0%BB%D1%8F,_%D0%BF%D1%83%D1%82%D1%8C,_%D1%86%D0%B8%D0%BA%D0%BB) G(V, E), заданный в виде матрицы смежности — двумерного массива d[][] размера n×n, в котором каждый элемент задаёт вес ребра между соответствующими вершинами. Нужно, чтобы выполнялось d[i][i]=0 для любых i.

Требуется найти матрицу кратчайших расстояний l, в которой элемент lij  либо равен длине кратчайшего пути из i в j, либо равен 0, если вершина j не достижима из i. Алгоритм Флойда-Уоршелла последовательно вычисляет все значения  {\displaystyle d\_{ij}^{k},} {\displaystyle \forall i,\;j} для k от 1 до n. Полученные значения {\displaystyle d\_{ij}^{n}} являются длинами кратчайших путей между вершинами i, j. На каждом шаге алгоритм генерирует [матрицу](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%82%D1%80%D0%B8%D1%86%D0%B0_(%D0%BC%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)) W, . Матрица W содержит длины кратчайших путей между всеми вершинами графа. Перед работой алгоритма матрица W заполняется длинами рёбер графа (или запредельно большим M, если ребра нет). Алгоритм работает за O(n3) времени и использует O(n2) памяти.

Если в графе есть циклы отрицательного веса, то формально алгоритм Флойда-Уоршелла неприменим к такому графу. На самом же деле, для тех пар вершин i и j, между которыми нельзя зайти в цикл отрицательного веса, алгоритм отработает корректно. Для тех же пар вершин, ответа для которых не существует (по причине наличия отрицательного цикла на пути между ними), алгоритм Флойда найдёт в качестве ответа какое-то число (возможно, сильно отрицательное, но не обязательно).

Написанная программа пошагово визуализирует работу алгоритма Флойда-Уоршелла на графе. При работе алгоритма подсвечиваются проверяемые на данной итерации вершины и пути, меняется матрица кратчайших путей. Кратчайший путь между двумя заданными вершинами подсвечивается на графе и в матрице кратчайших путей, а его длина выводится на экран.

**1. требования к программе**

* 1. **Исходные Требования к программе**
     1. **Требования к вводу исходных данных**

Матрица смежности должна вводиться в виде, удобном пользователю, а именно в виде таблицы. Также программа должна иметь возможность ввода матрицы смежности из выбранного пользователем файла. В файле матрица смежности указывается как последовательность чисел, где количество вершин определяется автоматически. 0 означает, что ребро отсутствует, а любое другое целое число является весом ребра. Файл выбирается пользователем.

Программа должна определять неверное количество входных элементов при вводе из файла, а также обнаруживать циклы отрицательного веса.

* + 1. **Требования к генерации графа**

Метод генерации графа должен по введённому количеству вершин и максимальному весу сгенерировать удобный пользователю граф (максимальное и минимальное количество входящих и исходящих ребер должно быть ограниченно, чтобы граф был достаточно «равномерным»).

* + 1. **Требования к визуализации**

Программа должна выполнять следующие действия:

* построить граф по матрице смежности;
* посчитать матрицу кратчайших путей для заданного графа;
* найти кратчайший путь между двумя вершинами заданного графа;

Матрица смежности должна вводиться в виде, удобном пользователю, а именно в виде таблицы. Граф строится в поле для визуализации работы алгоритма. Пользователь по желанию может ввести вместо матрицы список инцидентности. Вид представления графа пользователь должен указать.

Матрица кратчайших путей должна появиться в соответствующем поле в виде таблицы. При работе алгоритма должны подсвечиваться проверяемые на данной итерации вершины и пути и меняться матрица кратчайших путей в удобном для пользователя виде. При нахождении кратчайшего пути между двумя вершинами, вершины должны выбираться на графе интерактивно. Кратчайший путь должен подсветиться на графе, а его длина появится в соответствующем поле.

* 1. **Уточнение требований после сдачи прототипа**

Выбор файла, из которого программа читает матрицу смежности, должен осуществляться с помощью диалогового окна выбора файла. Колонки таблицы для ввода матрицы смежности не должны меняться местами. Первая колонка с номерами рядов (вершин) должна выделяться. После выбора файла матрица должна сразу вводится в таблицу.

* 1. **Уточнение требований после сдачи 1-ой версии**

Программа должна проверять количество элементов при вводе матрицы из файла (программе для корректной работы нужна строго квадратная матрица) и выводить пользователю сообщение о неправильном числе элементов. Метод генерации графа должен строить удобный для пользователя граф, с более равномерным распределением ребер.

Метод поиска кратчайшего пути между двумя вершинами должен вернуть кратчайший из найденных на момент вызова путь (а не кратчайший в принципе, если он существует).

**2. План разработки и распределение ролей в бригаде**

**2.1. План разработки**

26.06.2017 – разработка метода, строящего матрицу кратчайших путей с помощью алгоритма Флойда-Уоршелла. Разработка визуализации графа. Ввод матрицы смежности из файла и с помощью таблицы.

28.06.2017 – разработка метода генерации графа, разработка визуализации работы алгоритма. Разработка метода поиска кратчайшего пути между двумя вершинами.

30.06.2017 – разработка способа задачи графа списком инцидентности. Исправление метода генерации графа. Синхронизация таблицы для ввода матрицы смежности с матрицей кратчайших путей и отображением графа.

Разработка метода обнаружения циклов отрицательного веса. Проверка количества элементов при вводе матрицы смежности из файла.

**2.2. Распределение ролей в бригаде**

Нигай А.С. – разработка визуализации графа, визуализации работы алгоритма. Разработка метода обнаружения циклов отрицательного веса. Проверка количества элементов при вводе матрицы смежности из файла.

Матвеева А.И. – разработка метода, строящего матрицу кратчайших путей с помощью алгоритма Флойда-Уоршелла. Разработка метода генерации графа. Разработка метода поиска кратчайшего пути между двумя вершинами. Разработка способа задачи графа списком инцидентности.

Табунникова Н.Р. – разработка ввода матрицы смежности с помощью таблицы. Разработка удобного пользователю внешнего вида таблиц ввода и вывода. Синхронизация таблицы для ввода матрицы смежности с матрицей кратчайших путей и отображением графа.

**3. Особенности реализации**

**3.1. Использованные структуры данных**

Класс Alg - отвечает за выполнение алгоритма. Содержит матрицу кратчайших путей и матрицу нахождения пути, счетчики для пошаговой работы.

Класс CustomTableCellRenderer реализует необходимую отрисовку ячеек таблиц.

Класс Gen – описывает форму ввода данных для генерации графа.

Класс Graph – описывет граф. Содержит массив для хранения матрицы смежности, число элементов.

Класс GraphPanel – содержит метод отрисовки графа. Отображает граф в зависимости от текущего состояния работы алгоритма.

Класс TableHelper – содержит метод отрисовки таблицы.

Класс TableModelBase – базовый для TableModdelGraph и TableModelResalt.

Класс TableModelInc - описывает представление списка инцидентности в виде таблицы.

Класс TableModdelGraph – описывает представление матрицы смежности в виде таблицы.

Класс TableModelResalt – описывает представление матрицы кратчайших путей в виде таблицы.

**3.2. Основные методы**

Класс Alg включает в себя методы:

* · public static void Init(String s) – инициализация (чтение из строки) – определяет количество вершин, инициализирует массивы.
* · public static void AlgF() - алгоритм Флойда-Уоршелла, строит матрицу кратчайших путей, матрицу нахождения пути.
* · public static boolean check() - поиск циклов отрицательного веса, возвращает true, если таких не найдено.
* · public static void start() – начало работы алгоритма, инициализирует массивы.
* · public static void oneStep() – один шаг алгоритма, равносилен одной итерации алгоритма Флойда-Уршелла.
* · public static void finish() – действия после окончания работы алгоритма – обнуляет счётчики, редактирует массив кратчайших путей.
* · public static boolean done() – алгоритм завершил работу? Возращает true, если да.
* · public static ArrayList way(int n1, int n2) - возвращает кратчайший путь между двумя указанными вершинами.

Класс CustomTableCellRenderer реализует необходимую отрисовку ячеек таблиц

* · public CustomTableCellRenderer() – конструктор
* · public Component getTableCellRendererComponent(JTable table, Object value, boolean isSelected, boolean hasFocus, int row, int column) –получение компонента для отрисовки ячеек

Класс Gen

* · private void butOkActionPerformed(java.awt.event.ActionEvent evt) – генерация графа по нажатию кнопки, проверяет тип и диапазон введенных данных, выводит сообщение, если данные некорректны.

Класс Graph

* · public static void Arr\_in(String s) – заполнение матрицы смежности.
* · public static void getN(String s) – получение количества вершин.
* · public static int[][] gen(int n, int w) – генерация графа.
* · protected static int rnd(int max) – генерация случайного числа, в диапазоне от 0 до max.

Класс GraphPanel

* · private void formMouseClicked(java.awt.event.MouseEvent evt) – выделяет вершину, если это вторая выделенная вершина показывает кратчайший путь между ними.
* · private double[][] getVertexLoc() – вычисление координат вершин.
* · public void paint(Graphics g) – отрисовать граф, выделить проверяемые на данном шаге работы алгоритма вершины и рёбра.
* · private void obr(double x1, double y1, double x2, double y2,int w, Graphics2D g) – отображает направление и вес ребра.

TableHelper

* · public static void SetUpTable(JTable table, TableModel model) – выставить параметры таблицы (размер, заполнение ячеек и пр.).

TableModelBase

* · public Class<?> getColumnClass(int columnIndex) – получить класс колонок
* · public int getColumnCount() – получить количество колонок
* · public String getColumnName(int columnIndex) – получить имя колонки
* · public int getRowCount() – получить количество рядов
* · public abstract Object getValueAt(int rowIndex, int columnIndex) – получить значение в ячейке
* · public boolean isCellEditable(int rowIndex, int columnIndex) – ячейка редактируема?
* · public void setValueAt(Object value, int rowIndex, int columnIndex) – выставить значение ячейки
* TableModelGraph
* · public Object getValueAt(int rowIndex, int columnIndex) – получить значение в ячейке
* · public boolean isCellEditable(int rowIndex, int columnIndex) – ячейка редактируема?
* · public void setValueAt(Object value, int rowIndex, int columnIndex) – выставить значение в ячейке

TableModelInc

* · public int getRowCount() – получить количество строк
* · public int getColumnCount() – получить количество колонок
* · public String getColumnName(int columnIndex) - получить имя колонки
* · public Class<?> getColumnClass(int columnIndex) – получить класс колонки
* · public boolean isCellEditable(int rowIndex, int columnIndex) – ячейка редактируема?
* · public Object getValueAt(int rowIndex, int columnIndex) – получить значение в ячейке
* · public void setValueAt(Object aValue, int rowIndex, int columnIndex) – выставить значение в ячейке
* TableModelResult
* · public Object getValueAt(int rowIndex, int columnIndex) – получить значение ячейки

**4. Тестирование**

|  |  |  |
| --- | --- | --- |
| Действие | Что должно было произойти | Что произошло |
| Ввод отрицательного числа вершин | Вывод сообщения о выходе за пределы установленного диапазона числа вершин |  |
| Ввод буквы в поле для ввода количества вершин | Вывод сообщения о неправильности введённых данных. |  |
| Попытка ввести матрицу смежности из файла, в котором не записана квадратная матрица. | Вывод сообщения о неправильном числе элементов в выбранном файле. |  |
| Ввод отрицательного числа вершин в поле ввода числа вершин генерируемого графа. | Вывод сообщения о выходе за пределы установленного диапазона. |  |
| Ввод матрицы смежности с циклом отрицательного веса. | Вывод сообщения о присутствии циклов отрицательного веса. |  |

|  |  |  |
| --- | --- | --- |
| Поиск пути между двумя вершинами путь между которыми ешё не найден (но существует) | Путь не будет показываться. |  |
| Поиск пути между двумя вершинами путь между которыми уже найден | Путь подсветится, если он существует. |  |
| Поиск кратчайшего пути из вершины в эту же вершину | Путь подсветится, если он существует. |  |

**заключение**

В результате выполнения практической работы был написан визуализатор алгоритм Флойда-Уоршалла на языке Java с графическим интерфейсом.

Программа согласно спецификации должна была выполнять следующие действия:

* По введенной матрице смежности/ матрице инцидентности/ списку инцидентности отображать граф, работу алгоритма Флойда – Уоршелла;
* Генерировать новый граф;
* Иметь и при надобности выводить справку для пользователя.

В результате была написана программа, выполняющая следующие задачи:

* Отображение графа по введенной матрице смежности / списку инцидентности;
* Визуализация работы алгоритма Флойда – Уоршелла;
* Отображение кратчайшего пути между двумя заданными вершинами;
* Генерация нового графа;
* Сохранение матриц смежности и инцидентности в файл и чтение матрицы инцидентности из него.
* Вывод справки для пользователя.

Программа выполняет проверки на корректность входных данных и существование в графе циклов отрицательного веса.

Матрица смежности и матрица кратчайших путей выводятся в виде, удобном пользователю, а именно в виде таблицы. При работе алгоритма подсвечиваются проверяемые на данной итерации вершины и пути, меняется матрица кратчайших путей. При нахождении кратчайшего пути между двумя вершинами, вершины выбираются на графе интерактивно. Кратчайший путь подсвечивается на графе, а его длина появляется в соответствующем поле.

Все поставленные задачи были выполнены.

**список использованных источников**

1. Сьерра К., Бейтс Б. Изучаем Java, 2012
2. Васильев А.Н. Java Объектно-ориентированное программирование для магистров и бакалавров. СПб.: Питер, 2011
3. https://stepik.org/lesson/Примитивные-типы-12759/step/1?course=Java-Базовый-курс&unit=3107

**приложение А**

**исходный Код программы**

**Файл Alg.java:**

package pract;

import java.util.ArrayList;

/\*\*

\*

\* @author Asus-PC

\*/

public class Alg extends Graph{

public static int[][] next; //матрица нахождения путей

public static int[][] d; //матрица кратчайших путей

public static int iVertex = -1; // индекс вершины

public static int iFrom = -1; // из какой вершины

public static int iTo = -1; // в какую вершину

public static final int INF = 10000; // большое число

public static void Init(String s)

{

s = fix(s);

getN(s);

Arr\_in(s);

next = new int[N][N];

d = new int[N][N];

for (int i = 0; i < N; i++) {

for (int j = 0; j < N; j++) {

if (L[i][j] != 0) {

next[i][j] = j;

d[i][j] = L[i][j];

}

}

}

}

public static void AlgF() //собственно алгоритм Флойда Уоршалла

{

start();

for (int i = 0; i < N; i++) {

for (int u = 0; u < N; u++) {

for (int v = 0; v < N; v++) {

if (d[u][i] + d[i][v] < d[u][v]) {

d[u][v] = d[u][i] + d[i][v];

next[u][v] = next[u][i];

}

}

}

}

finish();

}

public static boolean check() //ищем циклы отрицательного веса

{

boolean c=true;

start();

Alg.AlgF();

for (int i=0; i<N; ++i)

for (int j=0; j<N; ++j)

for (int t=0; t<N; ++t)

if (d[i][t] < INF/2 && d[t][t] < 0 && d[t][j] < INF/2)

c = false;

finish();

return c;

}

// Начало работы алгоритма

public static void start() {

next = new int[N][N];

d = new int[N][N];

for (int i = 0; i < N; i++) {

for (int j = 0; j < N; j++) {

if (L[i][j] == 0) {

d[i][j] = INF;

} else {

d[i][j] = L[i][j];

next[i][j] = j;

}

}

}

iVertex = 0;

iFrom = 0;

iTo = 0;

}

// Один шаг работы алгоритма

public static void oneStep() {

if (d[iFrom][iVertex] + d[iVertex][iTo] < d[iFrom][iTo]) {

d[iFrom][iTo] = d[iFrom][iVertex] + d[iVertex][iTo];

next[iFrom][iTo] = next[iFrom][iVertex];

}

int imax = N - 1;

if (iTo < imax) {

iTo++;

} else if (iFrom < imax) {

iTo = 0;

iFrom++;

} else if (iVertex < imax) {

iFrom = 0;

iTo = 0;

iVertex++;

}

}

// Действия после окончания работы алгоритма

public static void finish() {

for (int i = 0; i < N; i++) {

for (int j = 0; j < N; j++) {

if (d[i][j] >= INF)

d[i][j]=0;

}

}

iVertex = -1;

iFrom = -1;

iTo = -1;

}

// Алгоритм закончил работу?

public static boolean done() {

int imax = N - 1;

return iFrom == imax && iTo == imax && iVertex == imax;

}

public static ArrayList way(int n1, int n2) //возвращает кратчайший путь

{

ArrayList<Integer> u = new ArrayList<>();

int S=0;

if (d[n1][n2]!=0)

{

if (n1==n2)

{

for (int i=0;i<N;++i)

{

u = new ArrayList<>();

S=0;

if (L[n1][i]!=0)

{

if (d[i][n2] != 0) {

int c = i;

u.add(n1);

u.add(c);

S=d[c][n2]+d[n1][c];

while (c != n2) {

c = next[c][n2];

u.add(c);

}

if (S==d[n1][n2])

return u;

}

}

}

}

else

if (d[n1][n2] != 0) {

int c = n1;

while (c != n2) {

c = next[c][n2];

u.add(c);

}

}}

return u;

}

}

**Файл CustomTableCellRenderer.java:**

package pract;

import java.awt.Color;

import java.awt.Component;

import javax.swing.JTable;

import javax.swing.table.DefaultTableCellRenderer;

public class CustomTableCellRenderer extends DefaultTableCellRenderer {

private final Color firstColumnColor = new Color(0xdd, 0xdd, 0xdd);

private final Color firstColumnOddRowColor = new Color(0xd0, 0xd0, 0xd0);

private final Color oddRowColor = new Color(0xf0, 0xf0, 0xf0);

private final Color highlightColor = new Color(0xff, 0xcc, 0xcc);

public CustomTableCellRenderer() {

super();

super.setHorizontalAlignment(javax.swing.JLabel.CENTER);

}

@Override

public Component getTableCellRendererComponent(JTable table, Object value,

boolean isSelected, boolean hasFocus, int row, int column) {

Component result = super.getTableCellRendererComponent(table, value, isSelected, hasFocus, row, column);

if (column == 0)

super.setBackground(row % 2 == 1 ? firstColumnOddRowColor : firstColumnColor);

else if (row == Alg.iFrom && (column - 1) == Alg.iTo)

super.setBackground(highlightColor);

else

super.setBackground(row % 2 == 1 ? oddRowColor : Color.white);

if (table.getColumnCount()-1>table.getRowCount())

{

if (column % 2 == 0 && column!=0)

super.setBackground(firstColumnColor);

}

super.setForeground(Color.black);

return result;

}

}

**Файл Gen.java:**

package pract;

import java.awt.event.WindowEvent;

import java.awt.event.WindowListener;

/\*\*

\*

\* @author Asus-PC

\*/

public class Gen extends javax.swing.JFrame implements WindowListener{

public int I=0;

/\*\*

\* Creates new form Gen

\*/

public Gen() {

initComponents();

}

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

jLabel3 = new javax.swing.JLabel();

jTextField3 = new javax.swing.JTextField();

jLabel1 = new javax.swing.JLabel();

jTextField1 = new javax.swing.JTextField();

butOk = new javax.swing.JButton();

jButton2 = new javax.swing.JButton();

setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE\_ON\_CLOSE);

setTitle("Генерировать граф");

jLabel3.setText("Введите максимальный вес рёбер");

jTextField3.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jTextField3ActionPerformed(evt);

}

});

jLabel1.setText("Введите количество вершин генерируемого графа");

jTextField1.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jTextField1ActionPerformed(evt);

}

});

butOk.setText("OK");

butOk.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

butOkActionPerformed(evt);

}

});

jButton2.setText("Отмена");

jButton2.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton2ActionPerformed(evt);

}

});

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addComponent(jLabel1, javax.swing.GroupLayout.DEFAULT\_SIZE, 326, Short.MAX\_VALUE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jTextField1, javax.swing.GroupLayout.PREFERRED\_SIZE, 40, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(layout.createSequentialGroup()

.addComponent(jLabel3, javax.swing.GroupLayout.DEFAULT\_SIZE, 326, Short.MAX\_VALUE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jTextField3, javax.swing.GroupLayout.PREFERRED\_SIZE, 40, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(layout.createSequentialGroup()

.addComponent(butOk)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(jButton2, javax.swing.GroupLayout.PREFERRED\_SIZE, 92, javax.swing.GroupLayout.PREFERRED\_SIZE)))

.addContainerGap())

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel1)

.addComponent(jTextField1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED\_SIZE, 20, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jTextField3, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(butOk)

.addComponent(jButton2))

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

);

pack();

}// </editor-fold>

private void jTextField3ActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

}

private void jTextField1ActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

}

private void butOkActionPerformed(java.awt.event.ActionEvent evt) {

int n, w;

String s;

try {

s = jTextField1.getText();

n = Integer.parseInt(s);

if (n < 2 || n > 20) {

javax.swing.JOptionPane.showMessageDialog(this, "Количество вершин графа вне диапазона 2...20!");

return;

}

} catch (NumberFormatException ex) {

javax.swing.JOptionPane.showMessageDialog(this, "Неверно задано количество вершин графа!");

return;

}

try {

s = jTextField3.getText();

w = Integer.parseInt(s);

if (w < 1 || w > 10) {

javax.swing.JOptionPane.showMessageDialog(this, "Вес ребра вне диапазона 1...10!");

return;

}

} catch (NumberFormatException ex) {

javax.swing.JOptionPane.showMessageDialog(this, "Неверно задан вес ребра!");

return;

}

Alg.N = n;

Alg.L = new int[n][n];

Alg.L = Graph.gen(n,w);

Alg.d = new int[n][n];

for (int i = 0; i < n; i++)

System.arraycopy(Alg.L[i], 0, Alg.d[i], 0, n);

I=1;

dispose();

}

private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {

I=0;

dispose();

}

// Variables declaration - do not modify

private javax.swing.JButton butOk;

private javax.swing.JButton jButton2;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel3;

private javax.swing.JTextField jTextField1;

private javax.swing.JTextField jTextField3;

// End of variables declaration

@Override

public void windowOpened(WindowEvent e) {

throw new UnsupportedOperationException("Not supported yet."); //To change body of generated methods, choose Tools | Templates.

}

@Override

public void windowClosing(WindowEvent e) {

throw new UnsupportedOperationException("Not supported yet."); //To change body of generated methods, choose Tools | Templates.

}

@Override

public void windowClosed(WindowEvent e) {

throw new UnsupportedOperationException("Not supported yet."); //To change body of generated methods, choose Tools | Templates.

}

@Override

public void windowIconified(WindowEvent e) {

throw new UnsupportedOperationException("Not supported yet."); //To change body of generated methods, choose Tools | Templates.

}

@Override

public void windowDeiconified(WindowEvent e) {

throw new UnsupportedOperationException("Not supported yet."); //To change body of generated methods, choose Tools | Templates.

}

@Override

public void windowActivated(WindowEvent e) {

throw new UnsupportedOperationException("Not supported yet."); //To change body of generated methods, choose Tools | Templates.

}

@Override

public void windowDeactivated(WindowEvent e) {

throw new UnsupportedOperationException("Not supported yet."); //To change body of generated methods, choose Tools | Templates.

}

}

**Файл Graph.java:**

package pract;

/\*\*

\*

\* @author Asus-PC

\*/

public class Graph {

static public int N = 0;//кол-во вершин

static public int[][] L;//матрица смежности

public static void Arr\_in(String s) {

L = new int[N][N];

String x = "";

s+=" ";

int j = 0, k = 0;

for (int i = 0; i < s.length(); i++)

{

if (!s.substring(i, i+1).equals(" "))

{

if (i==0) {

x = s.substring(i, i + 1);

} else {

if (s.substring(i-1, i).equals(" ")) {

x=s.substring(i,i+1);

} else {

x+=s.substring(i,i+1);

}

if (s.substring(i+1, i+2).equals(" ")) {

try{

int u = Integer.parseInt(x);

L[k][j] = u;

if (j == N - 1) {

j = 0;

++k;

} else {

++j;

}}catch (NumberFormatException ex){}

}

}

}

}

}

public static void getN(String s) {

N = 0;

s = fix(s);

for (int i = 1; i < s.length(); i++) {

String s1, s2;

s1 = s.substring(i, i + 1);

s2 = s.substring(i - 1, i);

if (s1.equals(" ") && !s2.equals(" ")) {

++N;

}

}

double n = Math.sqrt(N);

//N = (int) n;

int f = (int) n;

if (N!=f\*f)

{

N=0;

javax.swing.JOptionPane.showMessageDialog(null, "Неправильное количество элементов");

//return;

}

else

N=f;

}

public static String fix(String s) {

String S = " ";

for (int i=0;i<s.length();++i)

{

int n;

try {

n = Integer.parseInt(s.substring(i,i+1));

S+=n+"";

} catch (NumberFormatException ex) {

S+=" ";

}

}

S += " ";

return S;

}

public static int[][] gen(int n, int w) {

int[][] D = new int[n][n];

int[] b = new int[n];

int e = 0;

for (int i = 0; i < n; i++) {

int u = rnd(100);

int k = 0;

if (n > 3) {

k = 2;

} else if (n == 2) {

k = 1;

}

for (int j = 0; j < k; j++) {

int y = 0;

while (y == 0) {

int r = rnd(n - 1);

if (i == r) {

int z = 0;

for (int l = 0; l < n; l++) {

if ((b[l] <= e) && l != i) {

z = 1;

}

}

if (z == 0) {

++e;

}

}

if (r != i && b[r] <= e) {

D[i][r] = rnd(w);

++b[r];

y = 1;

}

}

}

}

int n1;

int n2;

for (int i = 0; i < n; i++) {

n1 = 0;

n2 = 0;

for (int j = 0; j < n; j++) {

if (D[i][j] == 0) {

++n1;

}

if (D[j][i] == 0) {

++n2;

}

}

if (n1 == n - 1) {

int c = rnd(n - 1);

while (c == i) {

c = rnd(n - 1);

}

D[i][c] = rnd(w);

}

if (n2 == n - 1) {

int c = rnd(n - 1);

while (c == i) {

c = rnd(n - 1);

}

D[c][i] = rnd(w);

}

}

return D;

}

protected static int rnd(int max) {

if (max != 0)

return (int)(Math.random() \* ++max);

return 0;

}

}

**Файл GraphPanel.java:**

package pract;

import java.awt.\*;

import java.util.ArrayList;

public class GraphPanel extends javax.swing.JPanel {

private static int c = 0;

private static int first = -1;

private static int sec = -1;

/\*\*

\* Creates new form GraphPanel

\*/

public GraphPanel() {

initComponents();

}

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

setBackground(new java.awt.Color(255, 255, 255));

setBorder(javax.swing.BorderFactory.createLineBorder(new java.awt.Color(153, 153, 153)));

addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseClicked(java.awt.event.MouseEvent evt) {

formMouseClicked(evt);

}

});

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(this);

this.setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGap(0, 398, Short.MAX\_VALUE)

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGap(0, 298, Short.MAX\_VALUE)

);

}// </editor-fold>

private void formMouseClicked(java.awt.event.MouseEvent evt) {

int x, y;

x = evt.getX();

y = evt.getY();

double F[][] = getVertexLoc();

for (int i = 0; i < Alg.N; ++i) {

if (Math.pow((x - F[i][0]), 2) + Math.pow((y - F[i][1]), 2) < 100) {

++c;

if (c == 1) {

first = i;

} else {

sec = i;

}

i = Alg.N;

}

else

{

}

}

this.repaint();

}

private double[][] getVertexLoc()

{

int N = Alg.N;

double[][] F = new double[N][2];

double PI = 3.14159;

double angle = 0;

if (N != 0) {

angle = 360 / N;

}

angle = angle \* PI / 180;

double r = 80;

double x = 275;

double y = 275;

double x0 = x, y0 = y;

y = 35;

double oldx, oldy;

for (int i = 0; i < N; ++i) {

F[i][0] = x;

F[i][1] = y;

oldx = x;

oldy = y;

x = (oldx - x0) \* Math.cos(angle) + (oldy - y0) \* Math.sin(angle) + x0;

y = -(oldx - x0) \* Math.sin(angle) + (oldy - y0) \* Math.cos(angle) + y0;

}

return F;

}

@Override

public void paint(Graphics g) {

super.paint(g);

Graphics2D gfx = (Graphics2D) g;

gfx.setStroke(new BasicStroke(1));

int N = Alg.N;

double F[][] = getVertexLoc();

if (c == 1) {

g.setColor(Color.red);

gfx.setStroke(new BasicStroke(3));

g.drawOval((int) F[first][0] - 18, (int) F[first][1] - 18, 36, 36);

} else {

if (c == 2) {

if (GraphView.I == 1) {

GraphView.I = 2;

GraphView.t.stop();

}

g.setColor(Color.red);

gfx.setStroke(new BasicStroke(3));

g.drawOval((int) F[first][0] - 18, (int) F[first][1] - 18, 36, 36);

g.drawOval((int) F[sec][0] - 18, (int) F[sec][1] - 18, 36, 36);

c = 0;

if (Alg.d[first][sec] != Alg.INF && Alg.d[first][sec] != 0) {

int Sum=0;

int t1, t2;

t1 = first;

ArrayList<Integer> w = Alg.way(first, sec);

for (int i = 0; i < w.size(); ++i) {

t2 = w.get(i);

g.setColor(Color.red);

gfx.setStroke(new BasicStroke(3));

g.drawLine((int) F[t1][0], (int) F[t1][1], (int) F[t2][0], (int) F[t2][1]);

Sum+=Alg.d[t1][t2];

t1 = t2;

}

if (first==sec)

Sum=Alg.d[first][sec];

g.setColor(Color.black);

g.drawString("Сумм. вес пути = "+Sum +".", 5, 15);

g.setColor(Color.red);

}

else

{

g.setColor(Color.black);

g.drawString("Путь не найден.", 5, 15);

g.setColor(Color.red);

}

}

}

gfx.setStroke(new BasicStroke(1));

if (GraphView.I != 0 && GraphView.I != 3) {

gfx.setStroke(new BasicStroke(5));

if (Alg.d[Alg.iFrom][Alg.iTo] == Alg.INF) {

g.setColor(Color.yellow);

} else {

g.setColor(Color.orange);

}

g.drawLine((int) F[Alg.iFrom][0], (int) F[Alg.iFrom][1], (int) F[Alg.iTo][0], (int) F[Alg.iTo][1]);

if (Alg.d[Alg.iFrom][Alg.iVertex] == Alg.INF) {

g.setColor(Color.cyan);

} else {

g.setColor(new Color(102, 153, 255));

}

g.drawLine((int) F[Alg.iFrom][0], (int) F[Alg.iFrom][1], (int) F[Alg.iVertex][0], (int) F[Alg.iVertex][1]);

if (Alg.d[Alg.iVertex][Alg.iTo] == Alg.INF) {

g.setColor(Color.cyan);

} else {

g.setColor(new Color(102, 153, 255));

}

g.drawLine((int) F[Alg.iVertex][0], (int) F[Alg.iVertex][1], (int) F[Alg.iTo][0], (int) F[Alg.iTo][1]);

gfx.setStroke(new BasicStroke(1));

}

for (int i = 0; i < N; i++) {

for (int j = 0; j < N; j++) {

if (Alg.L[i][j] != 0) {

g.setColor(Color.red);

g.drawLine((int) F[i][0], (int) F[i][1], (int) F[j][0], (int) F[j][1]);

gfx.setStroke(new BasicStroke(3));

obr(F[i][0], F[i][1], F[j][0], F[j][1], Alg.L[i][j], gfx);

gfx.setStroke(new BasicStroke(1));

}

}

}

gfx.setStroke(new BasicStroke(3));

for (int i = 0; i < N; i++) {

for (int j = 0; j < N; j++) {

if (Alg.L[i][j] != 0) {

obr(F[i][0], F[i][1], F[j][0], F[j][1], Alg.L[i][j], gfx);

}

}

}

gfx.setStroke(new BasicStroke(1));

for (int i = 0; i < N; i++) {

g.setColor(Color.green);

g.fillOval((int) (F[i][0] - 10), (int) F[i][1] - 10, 20, 20);

g.setColor(Color.black);

g.drawOval((int) (F[i][0] - 10), (int) F[i][1] - 10, 20, 20);

String s = i + "";

g.drawString(s, (int) (F[i][0]) - 5, (int) F[i][1] + 5);

}

gfx.setStroke(new BasicStroke(3));

g.setColor(Color.red);

if (Alg.iVertex != -1) {

g.drawOval((int) F[Alg.iVertex][0] - 13, (int) F[Alg.iVertex][1] - 13, 26, 26);

}

g.setColor(Color.blue);

if (Alg.iFrom != -1) {

g.drawOval((int) F[Alg.iFrom][0] - 15, (int) F[Alg.iFrom][1] - 15, 30, 30);

}

g.setColor(Color.orange);

if (Alg.iTo != -1) {

g.drawOval((int) F[Alg.iTo][0] - 13, (int) F[Alg.iTo][1] - 13, 26, 26);

}

gfx.setStroke(new BasicStroke(1));

g.setColor(Color.red);

}

private void obr(double x1, double y1, double x2, double y2,int w, Graphics2D g)

{

double x0, y0, x, y, xn, yn, x3, y3, x5,y5;

double PI = 3.14159;

double a = 45;

a = a \* PI / 180;

x0 = (x1 + x2) / 2;

y0 = (y1 + y2) / 2;

double k;

double b;

if (x1 != x2) {

k = (y2 - y1) / (x2 - x1);

b = y1 - k \* x1;

x = 10 \* Math.cos(Math.atan(k));

x5 = 5 \* Math.cos(Math.atan(k));

x = Math.abs(x);

x5 = Math.abs(x5);

x3 = 20 \* Math.cos(Math.atan(k));

x3 = Math.abs(x3);

if (x1 > x2) {

x0 = x5+x0;

x = x + x0;

x3 = x3 + x0;

} else {

x0=x0-x5;

x = x0 - x;

x3 = -x3 + x0;

}

y0 =k\*x0+b;

y = k \* x + b;

y3 = k \* x3 + b;

} else {

x = x0;

x3 = x0;

if (y1 > y2) {

y0+=5;

y = y0 + 10;

y3 = y0 + 20;

} else {

y0-=5;

y = y0 - 10;

y3 = y0 - 20;

}

}

xn = x0 - Math.sin(a) \* (y - y0) + Math.cos(a) \* (x - x0);

yn = y0 + Math.cos(a) \* (y - y0) + Math.sin(a) \* (x - x0);

g.drawLine((int) x0, (int) y0, (int) xn, (int) yn);

a = -45;

a = a \* PI / 180;

xn = x0 - Math.sin(a) \* (y - y0) + Math.cos(a) \* (x - x0);

yn = y0 + Math.cos(a) \* (y - y0) + Math.sin(a) \* (x - x0);

g.drawLine((int) x0, (int) y0, (int) xn, (int) yn);

g.setStroke(new BasicStroke(1));

g.setColor(Color.white);

g.fillOval((int) (x3 - 8), (int) (y3 - 8), 16, 16);

g.setColor(Color.red);

String S = w + "";

g.drawString(S, (int) (x3 - 5), (int) (y3 + 5));

g.drawOval((int) (x3 - 8), (int) (y3 - 8), 16, 16);

g.setStroke(new BasicStroke(3));

}

// Variables declaration - do not modify

// End of variables declaration

}

**Файл GraphView.java:**

package pract;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.io.File;

import java.io.FileNotFoundException;

import java.io.FileOutputStream;

import java.io.PrintWriter;

import javax.swing.Timer;

/\*\*

\*

\* @author Asus-PC

\*/

public class GraphView extends javax.swing.JFrame implements ActionListener{

public static Timer t;

public static int I=0;

/\*\*

\* Creates new form GraphView

\*/

public GraphView() {

initComponents();

}

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

jFileChooser1 = new javax.swing.JFileChooser();

jScrollPane1 = new javax.swing.JScrollPane();

jTable1 = new javax.swing.JTable();

NextButton = new javax.swing.JButton();

GoButton = new javax.swing.JButton();

PauseButton = new javax.swing.JButton();

jLabel1 = new javax.swing.JLabel();

graphPanel1 = new pract.GraphPanel();

SkipButton = new javax.swing.JButton();

AgainButton = new javax.swing.JButton();

jMenuBar1 = new javax.swing.JMenuBar();

jMenu1 = new javax.swing.JMenu();

jMenuItem1 = new javax.swing.JMenuItem();

jFileChooser1.setDialogType(javax.swing.JFileChooser.SAVE\_DIALOG);

jFileChooser1.setCurrentDirectory(null);

setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE\_ON\_CLOSE);

setTitle("Граф и матрица кратчайших путей");

setResizable(false);

addWindowListener(new java.awt.event.WindowAdapter() {

public void windowClosing(java.awt.event.WindowEvent evt) {

formWindowClosing(evt);

}

public void windowOpened(java.awt.event.WindowEvent evt) {

formWindowOpened(evt);

}

});

jTable1.setModel(new javax.swing.table.DefaultTableModel(

new Object [][] {

{null, null, null, null},

{null, null, null, null},

{null, null, null, null},

{null, null, null, null}

},

new String [] {

"Title 1", "Title 2", "Title 3", "Title 4"

}

));

jScrollPane1.setViewportView(jTable1);

NextButton.setText("Следующий");

NextButton.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

NextButtonActionPerformed(evt);

}

});

GoButton.setText("Пуск");

GoButton.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseClicked(java.awt.event.MouseEvent evt) {

GoButtonMouseClicked(evt);

}

});

GoButton.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

GoButtonActionPerformed(evt);

}

});

PauseButton.setText("Пауза");

PauseButton.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

PauseButtonActionPerformed(evt);

}

});

javax.swing.GroupLayout graphPanel1Layout = new javax.swing.GroupLayout(graphPanel1);

graphPanel1.setLayout(graphPanel1Layout);

graphPanel1Layout.setHorizontalGroup(

graphPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGap(0, 550, Short.MAX\_VALUE)

);

graphPanel1Layout.setVerticalGroup(

graphPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGap(0, 549, Short.MAX\_VALUE)

);

SkipButton.setText("Пропустить");

SkipButton.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

SkipButtonActionPerformed(evt);

}

});

AgainButton.setText("Заново");

AgainButton.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

AgainButtonActionPerformed(evt);

}

});

jMenu1.setText("Файл");

jMenuItem1.setText("Сохранить в файл");

jMenuItem1.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jMenuItem1ActionPerformed(evt);

}

});

jMenu1.add(jMenuItem1);

jMenuBar1.add(jMenu1);

setJMenuBar(jMenuBar1);

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addComponent(graphPanel1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(NextButton)

.addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED\_SIZE, 199, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING, false)

.addComponent(GoButton, javax.swing.GroupLayout.PREFERRED\_SIZE, 63, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(PauseButton))

.addGap(93, 93, 93)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(SkipButton, javax.swing.GroupLayout.Alignment.TRAILING)

.addComponent(AgainButton, javax.swing.GroupLayout.Alignment.TRAILING))

.addContainerGap())))

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(graphPanel1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGroup(layout.createSequentialGroup()

.addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(NextButton)

.addComponent(GoButton)

.addComponent(SkipButton))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED\_SIZE, 13, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(PauseButton)

.addComponent(AgainButton)))))

.addContainerGap(17, Short.MAX\_VALUE))

);

pack();

}// </editor-fold>

private void formWindowOpened(java.awt.event.WindowEvent evt) {

TableHelper.SetUpTable(jTable1, new TableModelResult());

Alg.start();

}

private void NextButtonActionPerformed(java.awt.event.ActionEvent evt) {

if (I == 1) {

I = 2;

t.stop();

}

if (I == 0) {

I = 2;

Alg.start();

t = new Timer(500, this);

t.start();

t.stop();

}

if (I != 3) {

Alg.oneStep();

}

if (Alg.done()) {

t.stop();

Alg.finish();

I = 3;

//jLabel1.setBackground(Color.green);

jLabel1.setText("Алгоритм завершил работу");

}

jTable1.repaint();

graphPanel1.repaint();

}

private void GoButtonActionPerformed(java.awt.event.ActionEvent evt) {

if (I == 0) {

I = 1;

Alg.start();

t = new Timer(500, this);

t.start();

}

if (I == 2) {

I = 1;

t.start();

}

if (I == 1) {

jLabel1.setText("Алгоритм работает...");

}

}

private void PauseButtonActionPerformed(java.awt.event.ActionEvent evt) {

if (I == 1) {

t.stop();

I = 2;

}

if (I == 2) {

jLabel1.setText("Алгоритм приостановлен");

}

}

private void formWindowClosing(java.awt.event.WindowEvent evt) {

if (I == 1) {

t.stop();

}

I = 0;

Alg.finish();

}

private void GoButtonMouseClicked(java.awt.event.MouseEvent evt) {

// TODO add your handling code here:

}

private void jMenuItem1ActionPerformed(java.awt.event.ActionEvent evt) {

if (jFileChooser1.showSaveDialog(this) == javax.swing.JFileChooser.APPROVE\_OPTION) {

File file = jFileChooser1.getSelectedFile();

try {

FileOutputStream fs = new FileOutputStream(file);

try (PrintWriter writer = new PrintWriter(fs)) {

for (int i = 0; i < Alg.N; i++) {

for (int j = 0; j < Alg.N; j++)

writer.print(Alg.d[i][j] + " ");

writer.println();

}

}

} catch (FileNotFoundException ex) {

javax.swing.JOptionPane.showMessageDialog(this, ex.getMessage());

}

}

}

private void SkipButtonActionPerformed(java.awt.event.ActionEvent evt) {

if (I == 1) {

t.stop();

}

if (!Alg.done()) {

Alg.AlgF();

Alg.finish();

I = 3;

jLabel1.setText("Алгоритм завершил работу");

}

jTable1.repaint();

graphPanel1.repaint();

}

private void AgainButtonActionPerformed(java.awt.event.ActionEvent evt) {

if (I == 1) {

t.stop();

}

I = 0;

Alg.start();

jTable1.repaint();

graphPanel1.repaint();

}

public static void main(String args[]) {

/\* Set the Nimbus look and feel \*/

//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">

/\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.

\* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html

\*/

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(GraphView.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(GraphView.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(GraphView.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(GraphView.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

new GraphView().setVisible(true);

}

});

}

// Variables declaration - do not modify

private javax.swing.JButton AgainButton;

private javax.swing.JButton GoButton;

private javax.swing.JButton NextButton;

private javax.swing.JButton PauseButton;

private javax.swing.JButton SkipButton;

private pract.GraphPanel graphPanel1;

private javax.swing.JFileChooser jFileChooser1;

private javax.swing.JLabel jLabel1;

private javax.swing.JMenu jMenu1;

private javax.swing.JMenuBar jMenuBar1;

private javax.swing.JMenuItem jMenuItem1;

private javax.swing.JScrollPane jScrollPane1;

private javax.swing.JTable jTable1;

// End of variables declaration

@Override

public void actionPerformed(ActionEvent e) {

Alg.oneStep();

if (Alg.done()) {

t.stop();

Alg.finish();

I = 0;

jLabel1.setText("Алгоритм завершил работу");

}

jTable1.repaint();

graphPanel1.repaint();

}

public void updateView() {

if (I == 1) {

t.stop();

}

I = 0;

Alg.finish();

jLabel1.setText("");

TableHelper.SetUpTable(jTable1, new TableModelResult());

Alg.start();

jTable1.repaint();

graphPanel1.repaint();

}

}

**Файл Help.java:**

package pract;

import javax.swing.DefaultComboBoxModel;

/\*\*

\*

\* @author Asus-PC

\*/

public class Help extends javax.swing.JFrame {

private String[] items = {

"Работа с главным окном",

"Работа с окном графа",

"Генерация",

"Список инцидентности"

};

/\*\*

\* Creates new form Help

\*/

public Help() {

initComponents();

}

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

jComboBox1 = new javax.swing.JComboBox<>();

jScrollPane2 = new javax.swing.JScrollPane();

jTextArea1 = new javax.swing.JTextArea();

setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE\_ON\_CLOSE);

addWindowListener(new java.awt.event.WindowAdapter() {

public void windowOpened(java.awt.event.WindowEvent evt) {

formWindowOpened(evt);

}

});

jComboBox1.setModel(new javax.swing.DefaultComboBoxModel<>(new String[] { "Item 1", "Item 2", "Item 3", "Item 4" }));

jComboBox1.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jComboBox1ActionPerformed(evt);

}

});

jTextArea1.setEditable(false);

jTextArea1.setBackground(new java.awt.Color(240, 240, 240));

jTextArea1.setColumns(20);

jTextArea1.setRows(5);

jTextArea1.setBorder(null);

jScrollPane2.setViewportView(jTextArea1);

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING, false)

.addComponent(jScrollPane2, javax.swing.GroupLayout.DEFAULT\_SIZE, 367, Short.MAX\_VALUE)

.addComponent(jComboBox1, 0, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addComponent(jComboBox1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jScrollPane2, javax.swing.GroupLayout.DEFAULT\_SIZE, 438, Short.MAX\_VALUE)

.addContainerGap())

);

pack();

}// </editor-fold>

private void formWindowOpened(java.awt.event.WindowEvent evt) {

// TODO add your handling code here:

DefaultComboBoxModel<String> cbModel = new DefaultComboBoxModel<String>();

for (int i = 0; i < items.length; i++)

cbModel.addElement((String)items[i]);

jComboBox1.setModel(cbModel);

if (((String)jComboBox1.getSelectedItem()).equals(items[0]))

{

String t=items[0];

t+=":\nДля изменения количества вершин в графе, введите нужное\nкол-во вершин в соответствующее текстовое поле.\n\nДля изменения/добавления/удаления ребра измените число\n"

+ "в ячейке [i][j], где i - номер вершины откуда идёт ребро, j - но-\nмер вершины куда идёт ребро.\n\n"+"Чтобы открыть окно с графом нажмите на кнопку 'Показать\nграф'.\n\n";

t+="Чтобы ввести матрицу смежности из файла нажмите 'Файл'->\n'Прочитать из файла' и выберите нужный.\n\n"+"Чтобы сохранить матрицу смежности в файл нажмите 'Файл'\n->'Сохранить в файл"

+ "' и выберите нужный.\n\n";

t+="Чтобы Сгенерировать новый граф нажмите 'Граф'->'Сгенери-\nровать...' и введите требуемые характеристики генерируемого\nграфа. Подробнее в п.Генерация\n\n";

t+="Чтобы задать граф списком инцидентности нажмите 'Граф'\n->'Задать граф списком инцидентности'. Подробнее в \nп.Список инцидентности\n\n";

jTextArea1.setText(t);

}

//jComboBox1.setModel();

//jComboBox1.setModel();

}

private void jComboBox1ActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

if (((String)jComboBox1.getSelectedItem()).equals(items[0]))

{

String t=items[0];

t+=":\nДля изменения количества вершин в графе, введите нужное\nкол-во вершин в соответствующее текстовое поле.\n\nДля изменения/добавления/удаления ребра измените число\n"

+ "в ячейке [i][j], где i - номер вершины откуда идёт ребро, j - но-\nмер вершины куда идёт ребро.\n\n"+"Чтобы открыть окно с графом нажмите на кнопку 'Показать\nграф'.\n\n";

t+="Чтобы ввести матрицу смежности из файла нажмите 'Файл'->\n'Прочитать из файла' и выберите нужный.\n\n"+"Чтобы сохранить матрицу смежности в файл нажмите 'Файл'\n->'Сохранить в файл"

+ "' и выберите нужный.\n\n";

t+="Чтобы Сгенерировать новый граф нажмите 'Граф'->'Сгенери-\nровать...' и введите требуемые характеристики генерируемого\nграфа. Подробнее в п.Генерация\n\n";

t+="Чтобы задать граф списком инцидентности нажмите 'Граф'\n->'Задать граф списком инцидентности'. Подробнее в \nп.Список инцидентности\n\n";

jTextArea1.setText(t);

} else {

if (((String)jComboBox1.getSelectedItem()).equals(items[1]))

{

String t=items[1];

t+=":\nЧтобы сохранить матрицу кратчайших путей на любом шаге в \nфайл нажмите 'Файл'->'Сохранить в файл"

+ "' и выберите нужный.\n\n";

t+="Управление визуализацией алгоритма:\n'Пуск' - начать или продолжить работу алгоритма;\n'Пауза' - остановить работу алгоритма;";

t+="\n'Пропустить' - пропустить визуализацию алгоритма и получить \nматрицу кратчайших путей;\n'Заново' - обнулить работу алгоритма;";

t+="\n'Следующий' - перейти к следующему шагу алгоритма;\n";

t+="\nПри работе алгоритма сравниваются пути из синей вершины в \nоранжевую через красную вершину. Если ребра между двумя \nвершинами нет, то синюю и оранжевую при визуализации \nсоединяет желтая линия, а пары вершин синяя-красная и \nкрасная-оранжевая - голубая.\n";

t+="Если ребро есть, то желтая линия становится оранжевой, а \nголубая - темно-синей.\n";

t+="Если сумма весов синих ребер меньше веса оранжевого, \nто эта сумма записывается в матрицу кратчайших путей как \nкратчайший путь между синей и оранжевой вершинами.\n";

jTextArea1.setText(t);

}

else

if (((String)jComboBox1.getSelectedItem()).equals(items[2]))

{

String t=items[2];

t+=":\nЧтобы Сгенерировать новый граф нажмите 'Граф'->'Сгенерир-\nовать...' и введите требуемые характеристики генерируемого\nграфа.\n\n";

t+="При генерации графа нужно указать количество вершин графа\nи максимальный вес ребер в соответствующих полях и нажать \nкнопку 'ОК'.\n";

t+="Количество вершин должно быть указано в пределах от 2 до \n20, а вес - от 1 до 100. Ребра с отрицательным весом не \nгенерируются.\n\n";

t+="При нажатии кнопки 'Отмена' окно закроется и граф не будет \nсгенерирован.\n\n";

jTextArea1.setText(t);

}

else

if (((String)jComboBox1.getSelectedItem()).equals(items[3]))

{

String t=items[3];

t+=":\nЧтобы задать граф списком инцидентности нажмите 'Граф'\n->'Задать списком инцидентности'.\n\n";

t+="В открывшемся окне введите количество вершин графа в \nсоответствующем поле, заполните таблицу и нажмите кнопку \n'ОК'.\n";

t+="Столбцы i и j таблицы заполняются автоматически, где i - вер-\nшина, из которой ребро исходит, а j - вершина, в которую ребро\nвходит.\n";

t+="Столбцы w(вес ребра) заполняются пользователем. Для до-\nбавления ребра введите в ячейку целое число отличное от\nнуля.\n\n";

t+="При нажатии кнопки 'Отмена' окно закроется и список \nинцидентности не будет задан.\n\n";

jTextArea1.setText(t);

}

}

}

// Variables declaration - do not modify

private javax.swing.JComboBox<String> jComboBox1;

private javax.swing.JScrollPane jScrollPane2;

private javax.swing.JTextArea jTextArea1;

// End of variables declaration

}

**Файл MainWindow.java:**

package pract;

import java.awt.event.WindowEvent;

import java.awt.event.WindowListener;

import java.io.\*;

import java.util.Scanner;

/\*\*

\*

\* @author Asus-PC

\*/

public class MainWindow extends javax.swing.JFrame {

/\*\*

\* Creates new form MainWindow

\*/

public static String filename="E:/testGraph.txt";

private GraphView graphView = null;

public MainWindow() {

initComponents();

}

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

jFileChooser1 = new javax.swing.JFileChooser();

jMenuBar2 = new javax.swing.JMenuBar();

jMenu4 = new javax.swing.JMenu();

jMenu5 = new javax.swing.JMenu();

butShowGraph = new javax.swing.JButton();

jLabel1 = new javax.swing.JLabel();

jScrollPane1 = new javax.swing.JScrollPane();

jTable1 = new javax.swing.JTable();

jTextField1 = new javax.swing.JTextField();

jLabel3 = new javax.swing.JLabel();

jMenuBar1 = new javax.swing.JMenuBar();

jMenu1 = new javax.swing.JMenu();

mnuOpenFile = new javax.swing.JMenuItem();

mnuSaveToFile = new javax.swing.JMenuItem();

jSeparator1 = new javax.swing.JPopupMenu.Separator();

jMenu2 = new javax.swing.JMenu();

mnuGenerateGraph = new javax.swing.JMenuItem();

jMenuItem1 = new javax.swing.JMenuItem();

jMenu3 = new javax.swing.JMenu();

jFileChooser1.setCurrentDirectory(null);

jFileChooser1.setDialogTitle("");

jMenu4.setText("File");

jMenuBar2.add(jMenu4);

jMenu5.setText("Edit");

jMenuBar2.add(jMenu5);

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

setTitle("Алгоритм Флойда-Уоршелла");

addWindowListener(new java.awt.event.WindowAdapter() {

public void windowOpened(java.awt.event.WindowEvent evt) {

formWindowOpened(evt);

}

});

butShowGraph.setText("Показать граф");

butShowGraph.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

butShowGraphActionPerformed(evt);

}

});

jLabel1.setText(" Введите матрицу смежности");

jTable1.setModel(new javax.swing.table.DefaultTableModel(

new Object [][] {

{null, null, null, null},

{null, null, null, null},

{null, null, null, null},

{null, null, null, null}

},

new String [] {

"Title 1", "Title 2", "Title 3", "Title 4"

}

));

jScrollPane1.setViewportView(jTable1);

jTextField1.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jTextField1ActionPerformed(evt);

}

});

jLabel3.setText("Введите число вершин");

jMenu1.setText("Файл");

mnuOpenFile.setText("Прочитать из файла...");

mnuOpenFile.setActionCommand("Открыть");

mnuOpenFile.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

mnuOpenFileActionPerformed(evt);

}

});

jMenu1.add(mnuOpenFile);

mnuSaveToFile.setText("Сохранить в файл...");

mnuSaveToFile.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

mnuSaveFileActionPerformed(evt);

}

});

jMenu1.add(mnuSaveToFile);

jMenu1.add(jSeparator1);

jMenuBar1.add(jMenu1);

jMenu2.setText("Граф");

mnuGenerateGraph.setText("Сгенерировать...");

mnuGenerateGraph.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

mnuGenerateGraphActionPerformed(evt);

}

});

jMenu2.add(mnuGenerateGraph);

jMenuItem1.setText("Задать списком инцидентности");

jMenuItem1.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jMenuItem1ActionPerformed(evt);

}

});

jMenu2.add(jMenuItem1);

jMenuBar1.add(jMenu2);

jMenu3.setText("Справка");

jMenu3.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseClicked(java.awt.event.MouseEvent evt) {

jMenu3MouseClicked(evt);

}

});

jMenu3.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jMenu3ActionPerformed(evt);

}

});

jMenuBar1.add(jMenu3);

setJMenuBar(jMenuBar1);

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addComponent(jTextField1, javax.swing.GroupLayout.PREFERRED\_SIZE, 88, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(0, 0, Short.MAX\_VALUE))

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()

.addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED\_SIZE, 200, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(butShowGraph))

.addComponent(jLabel1, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, 0, Short.MAX\_VALUE))

.addContainerGap())

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel3)

.addComponent(butShowGraph))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jTextField1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(11, 11, 11)

.addComponent(jLabel1)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jScrollPane1, javax.swing.GroupLayout.DEFAULT\_SIZE, 284, Short.MAX\_VALUE)

.addContainerGap())

);

pack();

}// </editor-fold>

private void butShowGraphActionPerformed(java.awt.event.ActionEvent evt) {

if (!Alg.check())

{if (javax.swing.JOptionPane.showConfirmDialog(this, "В графе есть циклы отрицательного веса.\nПродолжить?")==0)

showGraphView();}

else

showGraphView();

}

private void showGraphView() {

graphView = new GraphView();

graphView.setVisible(true);

butShowGraph.setEnabled(false);

graphView.addWindowListener(new java.awt.event.WindowAdapter() {

@Override

public void windowClosed(WindowEvent e) {

graphView = null;

butShowGraph.setEnabled(true);

}

});

}

public void updateGraphView() {

if (graphView != null)

graphView.updateView();

}

public void updateTable() {

TableHelper.SetUpTable(jTable1, new TableModelGraph(this));

}

private void mnuOpenFileActionPerformed(java.awt.event.ActionEvent evt) {

jFileChooser1.setDialogTitle("Открыть файл");

jFileChooser1.setPreferredSize(new java.awt.Dimension(700, 600));

if (jFileChooser1.showOpenDialog(this) == javax.swing.JFileChooser.APPROVE\_OPTION) {

File file = jFileChooser1.getSelectedFile();

try {

FileInputStream fs = new FileInputStream(file);

try (Scanner scanner = new Scanner(fs)) {

String s = scanner.useDelimiter("\\A").next();

Alg.Init(s);

TableHelper.SetUpTable(jTable1, new TableModelGraph(this));

jTextField1.setText(Alg.N + "");

updateGraphView();

}

} catch (FileNotFoundException ex) {

javax.swing.JOptionPane.showMessageDialog(this, ex.getMessage());

}

}

}

private void jTextField1ActionPerformed(java.awt.event.ActionEvent evt) {

String s = jTextField1.getText();

String S = "";

int l = s.length();

String digits = "0123456789";

for (int i = 0; i < l; i++) {

char ch = s.charAt(i);

if (digits.indexOf(ch) >= 0) {

S += ch;

}

}

s=S;

try {

int n = Integer.parseInt(s);

if (n < 2 || n > 20) {

javax.swing.JOptionPane.showMessageDialog(this, "Количество вершин графа вне диапазона 2...20!");

return;

}

s = "";

for (int i = 0; i < n \* n; ++i) {

s += "0 ";

}

}

catch (NumberFormatException ex) {

javax.swing.JOptionPane.showMessageDialog(this, "Неверно задано количество вершин графа!");

return;

}

Alg.Init(s);

TableHelper.SetUpTable(jTable1, new TableModelGraph(this));

updateGraphView();

}

private void formWindowOpened(java.awt.event.WindowEvent evt) {

String s = "";

for (int i = 0; i < 100; ++i)

s += "0 ";

Alg.Init(s);

TableHelper.SetUpTable(jTable1, new TableModelGraph(this));

}

private void mnuGenerateGraphActionPerformed(java.awt.event.ActionEvent evt) {

Gen g = new Gen();

g.setVisible(true);

MainWindow owner = this;

g.addWindowListener(new java.awt.event.WindowAdapter() {

@Override

public void windowClosed(WindowEvent e) {

if (g.I == 1) {

TableHelper.SetUpTable(jTable1, new TableModelGraph(owner));

jTable1.repaint();

if (graphView == null)

showGraphView();

else

updateGraphView();

}

}

});

}

private void mnuSaveFileActionPerformed(java.awt.event.ActionEvent evt) {

jFileChooser1.setDialogTitle("Сохранить файл");

jFileChooser1.setPreferredSize(new java.awt.Dimension(600, 400));

if (jFileChooser1.showSaveDialog(this) == javax.swing.JFileChooser.APPROVE\_OPTION) {

File file = jFileChooser1.getSelectedFile();

try {

FileOutputStream fs = new FileOutputStream(file);

try (PrintWriter writer = new PrintWriter(fs)) {

for (int i = 0; i < Alg.N; i++) {

for (int j = 0; j < Alg.N; j++)

writer.print(Alg.L[i][j] + " ");

writer.println();

}

}

} catch (FileNotFoundException ex) {

javax.swing.JOptionPane.showMessageDialog(this, ex.getMessage());

}

}

}

private void jMenuItem1ActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

MainWindow owner = this;

SpInc p = new SpInc(this);

p.setVisible(true);

p.addWindowListener(new java.awt.event.WindowAdapter() {

@Override

public void windowClosed(WindowEvent e) {

if (p.done()) {

TableHelper.SetUpTable(jTable1, new TableModelGraph(owner));

jTable1.repaint();

}

}

});

}

private void jMenu3ActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

new Help().setVisible(true);

}

private void jMenu3MouseClicked(java.awt.event.MouseEvent evt) {

// TODO add your handling code here:

new Help().setVisible(true);

}

/\*\*

\* @param args the command line arguments

\*/

public static void main(String args[]) {

/\* Set the Nimbus look and feel \*/

//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">

/\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.

\* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html

\*/

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(MainWindow.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(MainWindow.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(MainWindow.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(MainWindow.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable() {

@Override

public void run() {

new MainWindow().setVisible(true);

}

});

}

// Variables declaration - do not modify

private javax.swing.JButton butShowGraph;

private javax.swing.JFileChooser jFileChooser1;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel3;

private javax.swing.JMenu jMenu1;

private javax.swing.JMenu jMenu2;

private javax.swing.JMenu jMenu3;

private javax.swing.JMenu jMenu4;

private javax.swing.JMenu jMenu5;

private javax.swing.JMenuBar jMenuBar1;

private javax.swing.JMenuBar jMenuBar2;

private javax.swing.JMenuItem jMenuItem1;

private javax.swing.JScrollPane jScrollPane1;

private javax.swing.JPopupMenu.Separator jSeparator1;

private javax.swing.JTable jTable1;

private javax.swing.JTextField jTextField1;

private javax.swing.JMenuItem mnuGenerateGraph;

private javax.swing.JMenuItem mnuOpenFile;

private javax.swing.JMenuItem mnuSaveToFile;

// End of variables declaration

}

**Файл SpInc.java:**

package pract;

import java.awt.event.WindowEvent;

import java.awt.event.WindowListener;

/\*\*

\*

\* @author Asus-PC

\*/

public class SpInc extends javax.swing.JFrame implements WindowListener {

private final MainWindow owner;

private int I=1;

/\*\*

\* Creates new form SpInc

\*/

public SpInc(MainWindow owner) {

super();

initComponents();

this.owner = owner;

}

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

jLabel1 = new javax.swing.JLabel();

jButton1 = new javax.swing.JButton();

jTextField1 = new javax.swing.JTextField();

jScrollPane1 = new javax.swing.JScrollPane();

jTable1 = new javax.swing.JTable();

jButton2 = new javax.swing.JButton();

jLabel2 = new javax.swing.JLabel();

setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE\_ON\_CLOSE);

addWindowListener(new java.awt.event.WindowAdapter() {

public void windowOpened(java.awt.event.WindowEvent evt) {

formWindowOpened(evt);

}

});

jLabel1.setText("Введите число вершин");

jButton1.setText("Отмена");

jButton1.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton1ActionPerformed(evt);

}

});

jTextField1.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jTextField1ActionPerformed(evt);

}

});

jTable1.setModel(new javax.swing.table.DefaultTableModel(

new Object [][] {

{null, null, null, null},

{null, null, null, null},

{null, null, null, null},

{null, null, null, null}

},

new String [] {

"Title 1", "Title 2", "Title 3", "Title 4"

}

));

jScrollPane1.setViewportView(jTable1);

jButton2.setText("Задать граф");

jButton2.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton2ActionPerformed(evt);

}

});

jLabel2.setText("от вершины i, к j, весом w");

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()

.addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)

.addComponent(jScrollPane1, javax.swing.GroupLayout.DEFAULT\_SIZE, 580, Short.MAX\_VALUE)

.addGroup(javax.swing.GroupLayout.Alignment.LEADING, layout.createSequentialGroup()

.addComponent(jLabel1)

.addGap(18, 18, 18)

.addComponent(jTextField1, javax.swing.GroupLayout.PREFERRED\_SIZE, 47, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(18, 18, 18)

.addComponent(jLabel2, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

.addGroup(layout.createSequentialGroup()

.addComponent(jButton2)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(jButton1)))

.addContainerGap())

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()

.addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jTextField1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jLabel1)

.addComponent(jLabel2))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, 374, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jButton2)

.addComponent(jButton1))

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

);

pack();

}// </editor-fold>

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

I=0;

dispose();

}

private void jTextField1ActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

I=1;

int n=10;

try {

String s = jTextField1.getText();

n = Integer.parseInt(s);

if (n < 2 || n > 20) {

javax.swing.JOptionPane.showMessageDialog(this, "Количество вершин графа: вне диапазона 2...20");

return;

}

} catch (NumberFormatException ex) {

javax.swing.JOptionPane.showMessageDialog(this, "Неверно задано количество вершин графа");

return;

}

Alg.N=n;

Alg.L=new int[Alg.N][Alg.N];

TableHelper.SetUpTable(jTable1, new TableModelInc(this.owner));

//TableHelper.SetUpTable(jTable1, new TableModelGraph(owner));

}

private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

I=3;

dispose();

}

private void formWindowOpened(java.awt.event.WindowEvent evt) {

// TODO add your handling code here:

TableHelper.SetUpTable(jTable1, new TableModelInc(this.owner));

}

public boolean done()

{

if (I==1 || I==3)

return true;

else

return false;

}

public boolean shouldShow()

{

return I==3;

}

// Variables declaration - do not modify

private javax.swing.JButton jButton1;

private javax.swing.JButton jButton2;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel2;

private javax.swing.JScrollPane jScrollPane1;

private javax.swing.JTable jTable1;

private javax.swing.JTextField jTextField1;

// End of variables declaration

@Override

public void windowOpened(WindowEvent e) {

}

@Override

public void windowClosing(WindowEvent e) {

}

@Override

public void windowClosed(WindowEvent e) {

}

@Override

public void windowIconified(WindowEvent e) {

}

@Override

public void windowDeiconified(WindowEvent e) {

}

@Override

public void windowActivated(WindowEvent e) {

}

@Override

public void windowDeactivated(WindowEvent e) {

}

}

**Файл TableHelper.java:**

package pract;

import javax.swing.JTable;

import javax.swing.table.TableColumnModel;

import javax.swing.table.TableModel;

public class TableHelper {

private static final int CELL\_SIZE = 24;

public static void SetUpTable(JTable table, TableModel model) {

CustomTableCellRenderer renderer = new CustomTableCellRenderer();

table.setDefaultRenderer(Integer.class, renderer);

table.setModel(model);

table.getTableHeader().setReorderingAllowed(false);

table.setAutoResizeMode(JTable.AUTO\_RESIZE\_OFF);

table.setRowHeight(CELL\_SIZE);

TableColumnModel columnModel = table.getColumnModel();

int columnCount = columnModel.getColumnCount();

for (int i = 0; i < columnCount; i++) {

columnModel.getColumn(i).setPreferredWidth(CELL\_SIZE);

columnModel.getColumn(i).setWidth(CELL\_SIZE);

}

}

}

**Файл TableModelBase.java:**

package pract;

import java.util.HashSet;

import javax.swing.event.TableModelListener;

import javax.swing.table.TableModel;

public abstract class TableModelBase implements TableModel {

private HashSet<TableModelListener> listeners = new HashSet<>();

protected TableModelBase() {

}

@Override

public void addTableModelListener(TableModelListener listener) {

listeners.add(listener);

}

@Override

public void removeTableModelListener(TableModelListener listener) {

listeners.remove(listener);

}

@Override

public Class<?> getColumnClass(int columnIndex) {

return Integer.class;

}

@Override

public int getColumnCount() {

return Alg.N + 1;

}

@Override

public String getColumnName(int columnIndex) {

if (columnIndex == 0)

return "";

Integer id = columnIndex-1;

return id.toString();

}

@Override

public int getRowCount() {

return Alg.N;

}

@Override

public abstract Object getValueAt(int rowIndex, int columnIndex);

@Override

public boolean isCellEditable(int rowIndex, int columnIndex) {

return false;

}

@Override

public void setValueAt(Object value, int rowIndex, int columnIndex) {

}

}

**Файл TableModelGraph.java:**

package pract;

public class TableModelGraph extends TableModelBase {

private final MainWindow owner;

public TableModelGraph(MainWindow owner) {

super();

this.owner = owner;

}

@Override

public Object getValueAt(int rowIndex, int columnIndex) {

if (columnIndex == 0)

return rowIndex ;

return Alg.L[rowIndex][columnIndex - 1];

}

@Override

public boolean isCellEditable(int rowIndex, int columnIndex) {

return columnIndex > 0;

}

@Override

public void setValueAt(Object value, int rowIndex, int columnIndex) {

if (columnIndex == 0)

return;

Alg.L[rowIndex][columnIndex - 1] = Integer.valueOf(value.toString());

this.owner.updateGraphView();

}

}

**Файл TableModelInc.java:**

package pract;

import java.util.HashSet;

import javax.swing.event.TableModelListener;

import javax.swing.table.TableModel;

/\*\*

\*

\* @author Asus-PC

\*/

public class TableModelInc implements TableModel {

private HashSet<TableModelListener> listeners = new HashSet<>();

private final MainWindow owner;

TableModelInc(MainWindow owner) {

super();

this.owner = owner;

}

@Override

public int getRowCount() {

return Alg.N;

}

@Override

public int getColumnCount() {

return Alg.N\*2+1;

}

@Override

public String getColumnName(int columnIndex) {

if (columnIndex == 0)

return "i";

if (columnIndex%2==0)

return "w";

else

return "j";

}

@Override

public Class<?> getColumnClass(int columnIndex) {

return Integer.class;

}

@Override

public boolean isCellEditable(int rowIndex, int columnIndex) {

if (columnIndex==0)

return false;

else

return columnIndex%2==0;

}

@Override

public Object getValueAt(int rowIndex, int columnIndex) {

if (columnIndex == 0)

return rowIndex ;

if (columnIndex%2==0)

return Alg.L[rowIndex][(columnIndex-1)/2];

else

return (columnIndex-1)/2;

}

@Override

public void setValueAt(Object aValue, int rowIndex, int columnIndex) {

Alg.L[rowIndex][(columnIndex-1)/2]=Integer.valueOf(aValue.toString());

this.owner.updateGraphView();

this.owner.updateTable();

}

@Override

public void addTableModelListener(TableModelListener l) {

listeners.add(l);

}

@Override

public void removeTableModelListener(TableModelListener l) {

listeners.remove(l);

}

}

**Файл TableModelResult.java:**

package pract;

public class TableModelResult extends TableModelBase {

public TableModelResult() {

super();

}

@Override

public Object getValueAt(int rowIndex, int columnIndex) {

if (columnIndex == 0)

return rowIndex;

if (Alg.d[rowIndex][columnIndex - 1] == Alg.INF)

return 0;

return Alg.d[rowIndex][columnIndex - 1];

}

}