МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по учебной практике

Тема: Алгоритм A*

Студент гр. 9381 _____ Любимов В.А.

Студент гр. 9381 _____ Щеглов Д.А.

Студент гр. 9381 _____ Гурин С.Н.

Руководитель _____ Ефремов М.А

Санкт-Петербург

2021

ЗАДАНИЕ

НА УЧЕБНУЮ ПРАКТИКУ

Студент Любимов В.А., гр. 9381

Студент Щеглов Д.А., гр. 9381

Студент Гурин С.Н., гр. 9381

Тема практики: Алгоритм A*

Задание на практику:

Разработать визуализатор алгоритма A* на языке Java

Сроки прохождения практики: 01.07.2021 – 14.07.2021

Дата сдачи отчета: 14.07.2021

Дата защиты отчета:

Студент	_____	Любимов В.А.
Студент	_____	Щеглов Д.А.
Студент	_____	Гурин С.Н.
Руководитель	_____	Ефремов М.А.

АННОТАЦИЯ

Данная работа предусматривает создание GUI-приложения, позволяющее создать граф, а так же визуализировать на этом графе работу алгоритма A*.

Разработка данного приложения происходит на языке Java командой из 3-х человек. Роли каждого человека распределены в соответствии с

SUMMARY

This practical work provides for the creation of a GUI application that allows you to create a graph, as well as visualize the work of the A*algorithm on this graph.

The development of this application takes place in the Java language by a team of 3 people. The roles of each person are distributed in accordance with these tasks.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Целью данной практической работы является разработка GUI-приложения на языке Java. Данное приложение производит визуализацию работы алгоритма A*.

При разработке приложения планируется реализовать графичесский интерфейс для создания графа. Так же важной частью является реализация пошаговой визуализации работы алгоритма A*.

1. СПЕЦИФИКАЦИЯ ПРОГРАММЫ И ОРГАНИЗАЦИЯ РАБОТЫ

1.1. Диаграммы Use-Case и Классов

Диаграмма Use-Case:

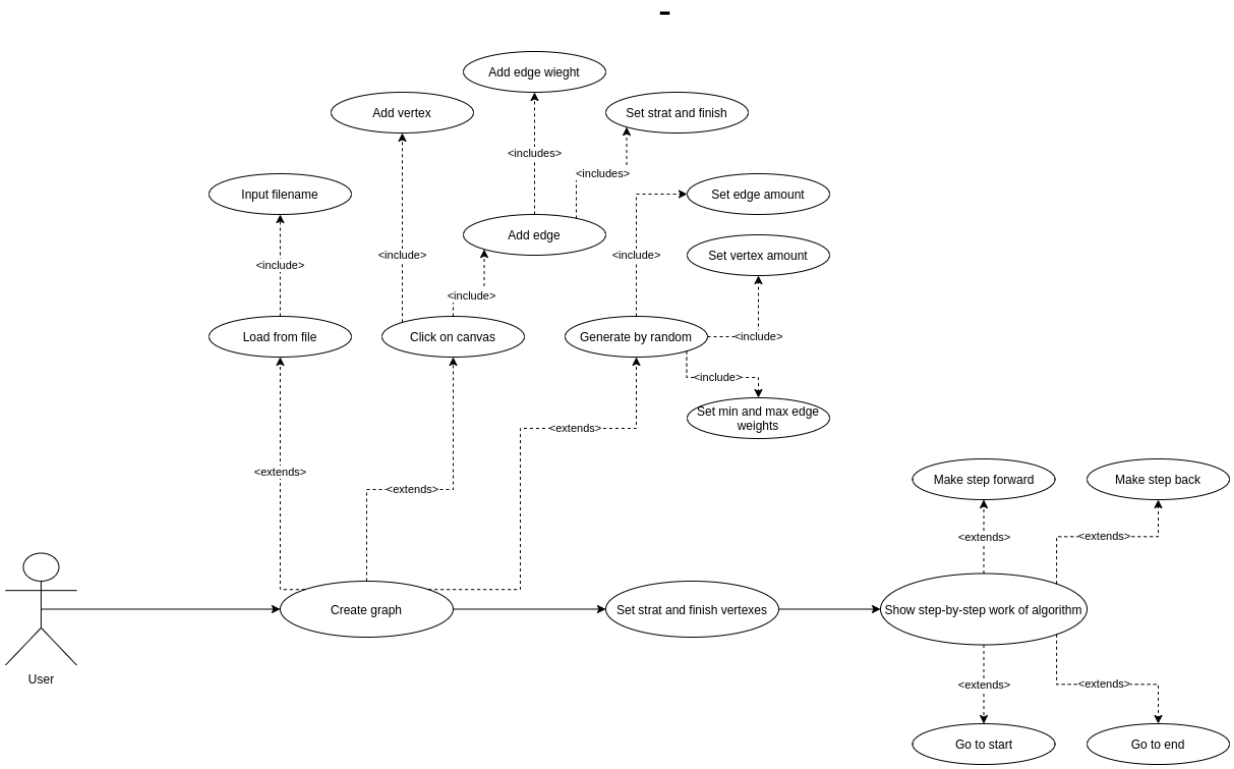

Рис. 1 Диаграмма Use-Case

Диаграмма Классов:

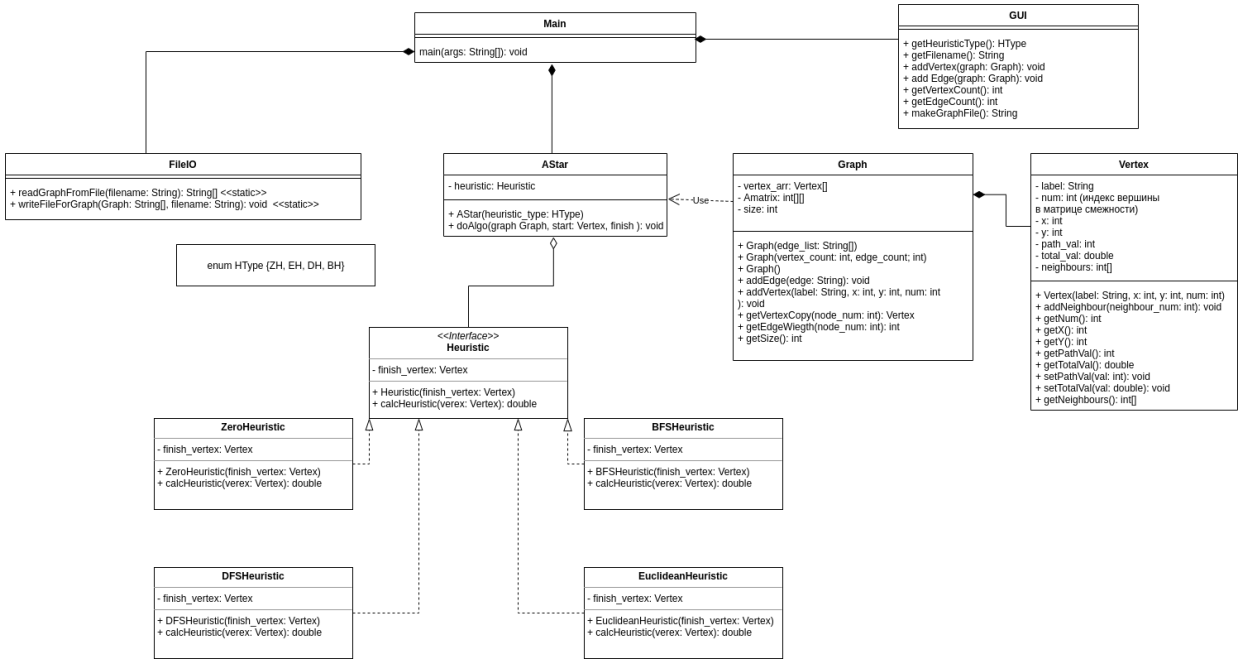

Рис. 2 Диаграмма Классов

1.2. План разработки

Задачи работы: изучение основ Java, создание приложения с графическим интерфейсом на Java (используя библиотеку Swing) для визуализации работы алгоритма, сборка (используя Maven) и тестирование программы.

План разработки:

1. 6 июля – создание прототипа, реализация алгоритма без демонстрации.
2. 8 июля – реализация демонстрации алгоритма в автоматическом режиме.
3. 10 июля – реализация демонстрации алгоритма в пошаговом режиме.
4. 12 июля – предоставление финальной версии программы, отчета и тестирования.

Распределение обязанностей:

Любимов В.А. – реализация алгоритма и классов бизнес-логики.

Щеглов Д.А. – реализация графического интерфейса, прототипа графического интерфейса и связи с бизнес-логики и интефейса.

Гурин С.Н. – написание отчета, сборка и тестирование программы.

2. ОПИСАНИЕ АЛГОРИТМА

2.1. Алгоритм A*

A* — это модификация алгоритма Дейкстры, оптимизированная для единственной конечной точки. Алгоритм Дейкстры может находить пути ко всем точкам, A* находит путь к одной точке. Он отдаёт приоритет путям, которые ведут ближе к цели.

A* пошагово просматривает все пути, ведущие от начальной вершины в конечную, пока не найдёт минимальный. Как и все информированные алгоритмы поиска, он просматривает сначала те маршруты, которые «кажутся» ведущими к цели. От жадного алгоритма, который тоже является алгоритмом поиска по первому лучшему совпадению, его отличает то, что при выборе вершины он учитывает, помимо прочего, весь пройденный до неё путь. Составляющая g(x) — это стоимость пути от начальной вершины, а не от предыдущей, как в жадном алгоритме.

В начале работы просматриваются узлы, смежные с начальным; выбирается тот из них, который имеет минимальное значение f(x), после чего этот узел раскрывается. На каждом этапе алгоритм оперирует с множеством путей из начальной точки до всех ещё не раскрытых (листовых) вершин графа — множеством частных решений, — которое размещается в очереди с приоритетом. Приоритет пути определяется по значению f(x) = g(x) + h(x). Алгоритм продолжает свою работу до тех пор, пока значение f(x) целевой вершины не окажется меньшим, чем любое значение в очереди, либо пока всё дерево не будет просмотрено. Из множества решений выбирается решение с наименьшей стоимостью.

Чем меньше эвристика h(x), тем больше приоритет, поэтому для реализации очереди можно использовать сортирующие деревья.

2.2. Реализация графа

В качестве эвристики берется |cur_x – finish_x| + |cur_y – finish_y|. Данный граф состоит из вершин хранящихся в динамическом массиве. Вершина этого графа является объект класса Vertex. У нее имеется имя, координаты, имя предыдущего элемента пути, вес пути до этой вершины, суммарную оценку (вес пути + эвристика) и список ребер исходящих из данной вершины.

Для A* в качестве следующей к рассмотрению вершины при равенстве суммарных оценок выбирается первая добавленная вершина в список вершин, которые необходимо рассмотреть.

A* выбрасывает исключение только, если в данном графе нет стартовой вершины. В остальных случаях будет выведен путь или сообщение об отсутствии пути.

ЗАКЛЮЧЕНИЕ

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

ПРИЛОЖЕНИЕ А
НАЗВАНИЕ ПРИЛОЖЕНИЯ