**Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования «Московский технический университет связи и информатики»**

**Лабораторная работа № 3**

Выполнил студент

Группы «БСТ1601»

Спивак Александр

Москва 2018

Оглавление

[Цель работы: 3](#_Toc507356670)

[Задание на разработку проекта: 3](#_Toc507356671)

[Анализ предметной области и выбор инструментария: 3](#_Toc507356672)

[Объяснение функций: 3](#_Toc507356673)

[Вывод: 3](#_Toc507356674)

# 

# Цель работы:

Изучение алгоритма, который постепенно создает путь от источника до места назначения, всегда используя "до сих пор лучший путь ", чтобы сделать следующий шаг.

# Задание на разработку проекта:

1.Location:

обеспечение реализации метода equals ()

обеспечение реализации метода hashcode()

2.AStarState:

Добавить две (нестатические) области класса AStarState с этим типом

реализовать следующие методы типа AStarState:

public int numOpenWaypoints()

public Waypoint getMinOpenWaypoint()

public boolean addOpenWaypoint(Waypoint newWP)

public boolean isLocationClosed(Location loc)

public void closeWaypoint(Location loc)

3.Компиляция и выполнение программы

# Анализ предметной области и выбор инструментария:

NetBeans IDE 8.2

# Объяснение функций:

* Map2D.java - представляет собой карту, по которой алгоритм двигается, в том числе проходимы ли клетки.
* Location.java - этот тип представляет собой координаты конкретной ячейки на карте.
* Waypoint.java - представляет отдельные точки в созданный путь.
* AStarPathfinder.java - этот тип реализует алгоритм поиска пути как статический метод.
* AStarState.java - этот тип хранит набор открытых точек и закрытых точек и обеспечивает базовые операции, необходимые для функционирования алгоритма поиска.
* AStarApp.java - простое Swing - приложение, которое обеспечивает редактируемый вид 2D карты, и запускает поиск пути по запросу.
* JMapCell.java - это Swing - компонент, который используется для отображения состояния клеток на карте.

# 

# Вывод:

Каждая итерация алгоритма довольно проста: найти наименее дорогостоящий пункт из набора открытых точек, сделать шаг в любом направлении от этой точки для создания новой открытой точки, а затем переместить путевую точку из открытого набора в закрытый набор. Это повторяется до тех пор, пока текущая точка не достигнет пункта назначения! Если во время этого процесса алгоритм открытых точек заканчивается, то нет пути от начальной точки до пункта назначения. Такая обработка в первую очередь зависит от расположения точек, поэтому он очень полезен для хранения путевых точек как отображение от места до соответствующих точек.