ФЕДЕРАЛЬНОЕ АГЕНСТВО СВЯЗИ

Ордена Трудового Красного Знамени

федеральное государственное бюджетное

образовательное учреждение высшего образования

«Московский Технический Университет Связи и Информатики» (МТУСИ)

Кафедра МКиИТ

Лабораторная работа по технологиям программирования №3

«Java-Я-звезда!»

Выполнил:

Студент 2 курса

Группы БСТ-1602

Абросимов Данил

Вариант №1

Москва, 2018

Оглавление

[1. Цель и задачи. 3](#_Toc507964498)

[1.1. Доработать класс location, реализовав проверку точек на идентичность и реализовав метод кучи. 3](#_Toc507964499)

[1.2. Доработать класс AStarState, реализовав все нужные для работы программы функции.. 3](#_Toc507964500)

[2. Анализ предметной области и выбор инструментария. 3](#_Toc507964501)

[3. Код программы и объяснение функций. 3](#_Toc507964502)

[3.1. Класс Location. 3](#_Toc507964503)

[3.2. Класс AppStarState. 4](#_Toc507964504)

[4. Заключение. 6](#_Toc507964505)

1. Цель и задачи.
   1. Доработать класс location, реализовав проверку точек на идентичность и реализовав метод кучи.
   2. Доработать класс AStarState, реализовав все нужные для работы программы функции..
2. Анализ предметной области и выбор инструментария.

В текущей работе я использовал бесплатно распространяемый пакет разработчика Java Development Kit. Однако, этот пакет не предусматривает интегрированную среду разработки, поэтому для написания непосредственно кода мною была выбрана программа Notepad++, так как она обеспечивает достаточно гибкий интерфейс и предоставляет возможность подсветки синтаксиса языка.

1. Код программы и объяснение функций.
   1. Класс Location.

public class Location

{

/\*\* X coordinate of this location. \*\*/

public int xCoord;

/\*\* Y coordinate of this location. \*\*/

public int yCoord;

/\*\* Creates a new location with the specified integer coordinates. \*\*/

public Location(int x, int y)

{

xCoord = x;

yCoord = y;

}

/\*\* Creates a new location with coordinates (0, 0). \*\*/

public Location()

{

this(0, 0);

}

public boolean equals(Object obj)

{

Location location = (Location) obj;

if (location == null) {

return false;

}

if (xCoord == location.xCoord && yCoord == location.yCoord)

{

return true;

}

return false;

}

public int hashCode()

{

int result = 0;

result += xCoord;

result += yCoord;

return result;

}

}

Добавлены функции equals, который возвращает результат сравнения дву переменных типа Location и hashCode, которая возвращает число из кучи, основанной на x, y координатах.

* 1. Класс AppStarState.

public class AStarState

{

/\*\* This is a reference to the map that the A\* algorithm is navigating. \*\*/

private Map2D map;

private HashMap<Location, Waypoint> openWaypoints;

private HashMap<Location, Waypoint> closedWaypoints;

/\*\*

\* Initialize a new state object for the A\* pathfinding algorithm to use.

\*\*/

public AStarState(Map2D map)

{

if (map == null)

throw new NullPointerException("map cannot be null");

this.map = map;

openWaypoints = new HashMap<Location, Waypoint>();

closedWaypoints = new HashMap<Location, Waypoint>();

}

/\*\* Returns the map that the A\* pathfinder is navigating. \*\*/

public Map2D getMap()

{

return map;

}

public Waypoint getMinOpenWaypoint()

{

Waypoint minWaypoint = null;

float min = 999999999;

float cost = 0;

for(Waypoint i : openWaypoints.values())

{

cost = i.getTotalCost();

if(min > cost)

{

min = cost;

minWaypoint = i;

}

}

return minWaypoint;

}

public boolean addOpenWaypoint(Waypoint newWP)

{

Waypoint point = openWaypoints.get(newWP.getLocation());

if(point == null || newWP.getPreviousCost() < point.getPreviousCost())

{

openWaypoints.put(newWP.getLocation(), newWP);

return true;

}

return false;

}

/\*\* Returns the current number of open waypoints. \*\*/

public int numOpenWaypoints()

{

return openWaypoints.size();

}

/\*\*

\* This method moves the waypoint at the specified location from the

\* open list to the closed list.

\*\*/

public void closeWaypoint(Location loc)

{

Waypoint point = openWaypoints.remove(loc);

if(point != null)

{

closedWaypoints.put(loc, point);

}

}

public boolean isLocationClosed(Location loc)

{

return closedWaypoints.containsKey(loc);

}

}

Добавлены переменные типа HashMap openWaypoints closedWaypoints.  
Фуникия getMinOpenWaypoint(), которая находит самую дешевую по цене точку из массива точек.

Функция numOpenWaypoints(), которая возвращает количество открытых точек.

Функция addOpenWaypoint(Waypoint newWP), которая создает открытую точку, если такой не было и заменяет «старую» открытую точку на «новую» открытую точку, если цента новой меньше цены старой точки.

Функция closeWaypoint(Location loc), которая перемещает открытую точку к набору закрытых точек.

Функция isLocationClosed(Location loc), которая возвращает true, если точка закрыта и false, если открыта

1. Заключение.

В ходе работы мы частично узнали как работает визуальная среда java, познакомились с типом HashMap и смогли дописать pathfinder.