Maze Searching System Requirement Specification

James Roe

December 2022

Contents

1 Introduction

1.1 Purpose

The purpose of this system requirements specification document is to outline the requirements for the Maze Searching software to be created for the Introduction to programming mini project. The intended audience is the designer / coder James Roe, as well as the lecturer of aforementioned course Dr Bill Teahan. Other possible audiences include fellow course members of the author, and potentially more broadly others in the school of computer science.

1.2 Scope

Dr Bill Teahan requested that students carry out a mini project of a program that will search through a maze, and return possible paths of said maze. The functionality has been left largely up to the students with the required functionality as follows: the program must have an interface to operate it, The program must be able to display a number of mazes, the program must be able to solve these mazes. Proposed further functionality are as follows: A graphical interface to allow for greater interaction, animation of the maze being traversed as it is being searched, ability to choose from multiple search methods, a comparison of the efficiency of different search methods for a given maze, an option to import mazes into the program, a procedural maze generator. The overall goal of this program is to secure James Roe a good grade, but also to be a tool to help the user of the program to better understand search algorithms through visualisation of the algorithms at work.

1.3 Definitions, Acronyms, or Abbreviations

- BFS - Breadth first search. A search algorithm that traverses a graph by visiting each node connected to the current node before traversing each of the nodes linked to those.
- DFS - Depth first search
- Maze search - the process of finding a path from the start(entrance) of the maze to the end(exit/centre) of the maze. May also be referred to as searching the maze.

1.4 References

1.5 Overview

Section 2 of the specification outlines a general description of the program in terms of its functions, characteristics, and constraints as wells as its relative position when compared with other programs, and any assumptions made regarding the program. Section 3 gives an in depth look at the requirements of the program with regards to its function, that is to say what the program can do. Section 4 looks at non functional requirements. This is requirements that are necessary for the program which cannot be defined as functional, such as the necessity for the program to execute in a timely manner. Section 5 looks at the system architecture through the use of a class diagram. Section 6 provides a use case diagram for the sake of describing the functionality of the program. Finally we have the appendices.

2 General Description

2.1 Product Perspective

The program shall be standalone, and thus will not interface with other systems. It is created for educational purposes, both for the designer / programmer (James Roe) to better their programming skills but also for any user to help them visualise the search algorithms within the program in action. At the time of the creation of this document the responsibility of the program and its upkeep rests solely with the creator of the program.

2.2 Product Functions

The maze searching program allows the user to select mazes built into the program, and draws the selected maze within a graphical window that is part of the program. The user can then be select a search method which the program uses to search the maze. The program animates this process using lines that it draws as it traverses the maze. Upon finding the solution path it highlights this path, and erases all other paths. Upon completion of a search, the user can choose to see further information regarding the search and the maze, such as approximate time to complete search, number of rooms/cells traversed, percentage of the maze searched through, and Total number of rooms in the maze. The user has the option to import their own mazes into the program so that it can search through them.

2.3 User Characteristics

The likely users of the maze searching program shall be the creator James Roe, the client Dr Bill Teahan, and potentially others in the MSc Computing course whom are interested in viewing the program. All users of the program will have the same level of access to it, and so responsibility should be of the same level. The UI should be intuitive, and easy to use, as the training necessary to use the program should be none.

2.4 General Constraints

As mentioned in User Characteristics the UI will be intuitive enough such that no training is required by the user to make use of the program. The program will hold all the data it requires, but users may submit data for the program to draw mazes. This data will then be stored locally alongside the program.

2.5 Assumptions

The maze searching program shall be created in python, and as a result will require python to run. It shall, in particular require python version 3.10. In addition the maze's will be stored as json files within the program's directory, and so for the functionality of searching through a particular maze, the user will have to ensure that said maze's json file is there.

3 Functional Requirements

1. The program shall have a number of mazes for the user to select from.

2. The program shall be able to draw these prior mentioned mazes within a graphical window.

3. The program shall have a number of search methods for the user to select from.

4. Upon the user selecting a search method, The program shall traverse the maze using the search method and animate a line being drawn through the maze as it is traversed. The decision behind this functionality is that it will help the user to better understand how each of the search methods traverse the maze in order to find the path from entrance to exit, through a visual representation of the search method

5. The program shall allow the user to pause/play the animation of the maze search, as well as go through it step by step.

6. Upon completion of the search, the program shall offer further information about the maze and the search method. This information shall include the search method chosen, the name of the maze, the number of rooms of the maze, the time taken to search through the maze, the number of rooms traversed, the percentage of the rooms traversed, the size of the maze as a number of rooms. The idea here is that seeing the search method traverse the maze may mislead the user about the efficiency of each search method. This gives some ability to quantify a search method's efficacy.

7. The program shall allow the user to import mazes they have created themselves and thus give the user the ability to draw out the maze and search through it.

8. The program shall be able to show a comparison of different search algorithms on a maze.

9. The program shall be able to procedurally generate mazes of different sizes, and perform the previously mentioned operations on these mazes. The main reason for this function is a precursor to the next function on this list. By procedurally generating mazes to solve we gain a larger pool of data to draw from when attempting to work out the efficiency of the search algorithms.

10. The program shall be able to create a graph of the performance of the search methods compared with size of the mazes, to give the user an idea of the efficiency of the search methods as maze size increases.

4 Non Functional Requirements

1. The maze's that are drawn shall fit inside the graphical window they are drawn in such that the user can see the entirety of the maze
2. The mazes shall be searched in a timely manner. This is relative, as the search method chosen as well as the maze chosen shall affect the time taken to complete the search.
3. As mazes can be imported into the program it is important that the method for importing the mazes shall be easy to understand and to carry out.

5 System Architecture

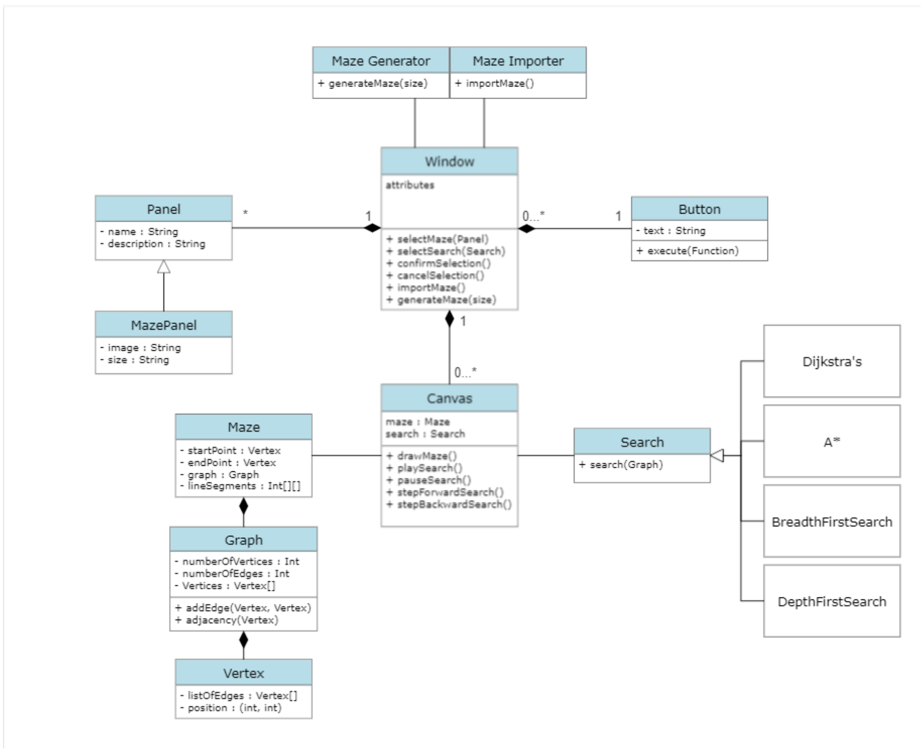
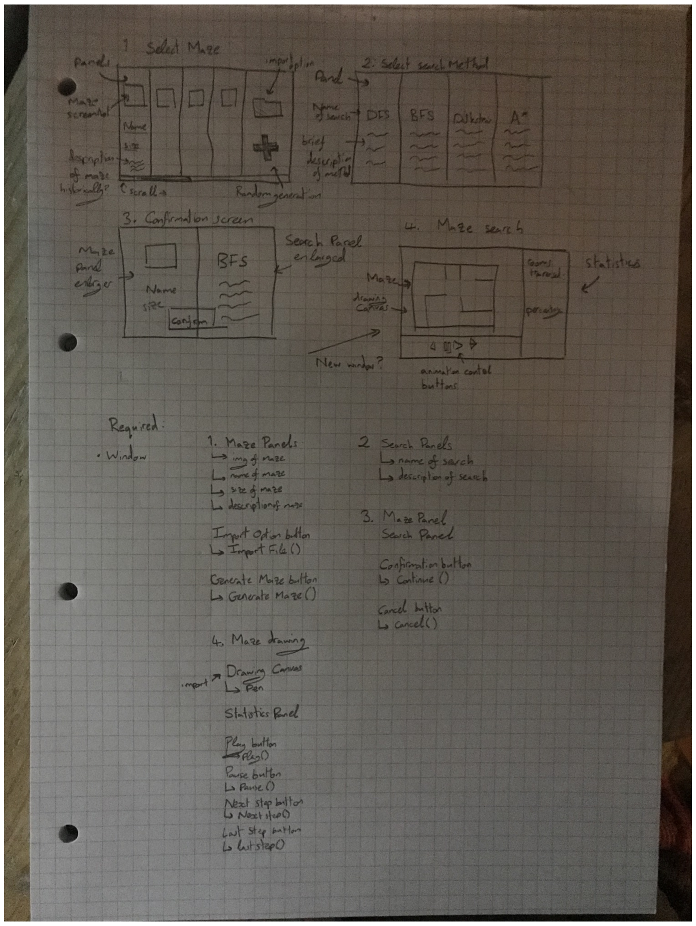


Figure 1: Class Diagram of Maze Searching Program



Required:

- Window

1. Maze Panels
   ↳ img of maze
   ↳ name of maze
   ↳ size of maze
   ↳ description of maze

   Import Option button
   ↳ Import File()

   Generate Maze button
   ↳ Generate Maze()

2 Search Panels
   ↳ name of search
   ↳ description of search

3. Maze Panel
   Search Panel

   Confirmation button
   ↳ Continue ()

   Cancel button
   ↳ cancel()

4. Maze drawing

   import ↗ Drawing Canvas
   ↳ Pen

   Statistics Panel

   Play button
   ↳ Play()
   Pause button
   ↳ Pause()
   Next step button
   ↳ Next step()
   Last step button
   ↳ last step()

Figure 2: Very rough sketch of design (not to be included in final report)

6 System Models

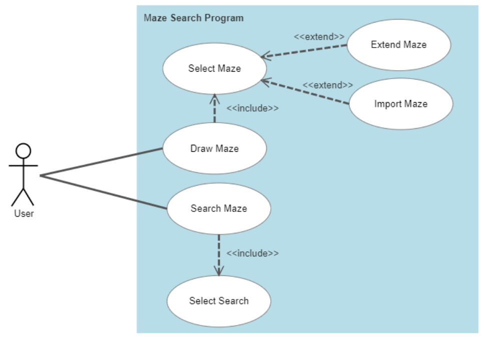


Figure 3: Use Case Diagram of Maze Searching Holiday