**Problem Background**

The aim of this project is to design, develop and test a complex game that can test a user’s ability to think under pressure whilst sustaining accuracy in those decisions. I concluded that this was possible through a game revolving around mazes.

Mazes are known to be unique and challenging, from their start point to their end point, they are designed to test someone’s cognitive ability to recognise patterns and use specific memory techniques to solve the maze.

My initial research found me playing mazes games across the internet; while lots of games had promoted difficult and brain-teasing mazes to play, I found these very basic and generic. Lots of games were level based, the first level was generally very easy, a clear start point is shown, and an end goal is clearly indicated on the screen. They also use the directional keys on the keyboard to enable the user to solve the maze. As levels increase, the mazes kept getting harder, however, this became quickly repetitive. With ease, a user could complete 50 levels in under 10 minutes. Quickly becoming obvious, these games contained little to no competitiveness, no challenging mazes and lots of repetition through completing similar mazes over and over. On another hand, lots of these games had appealing user interfaces (UI) that I believe was essential to capture my initial attention.

In Summary, current maze games are:

* Too easy
* Repetitive
* Missing a competitive factor

**Objectives**

The overcome these problems, this project aims to address the following objectives:

* Create a Maze Solver Game that uses a Maze Generation Algorithm and uses an Algorithm to set that maze of a certain difficulty.
  + The maze generation will be able to generate completely random mazes which ensures that user will not experience the same maze twice.
* Create a Mechanism that allows the user to manually solve a Maze.
  + Will include a timer to increase the competitive factor of the game.
* Create a Maze Creation Game that allows the user to create their own maze with their own settings, starting point and end point.
  + An alternative mode can keep the user playing the game, avoiding repetition by replaying the same game over and over.
* Create a Shortest Path Algorithm that solves the maze created by the user.
  + Increases the competitive factor as the user will aim to create a maze that is difficult for the shortest path algorithm.
* Create an enjoyable, competitive game that keeps users playing and competing.

**Communication with End-User**

As my project can be played by anyone of any age or any background, I created an online form using Google Forms to grasp a stronger connection between my game and the potential end-users. I obtained a general idea about the game and what some of the end-users thought about it. The following data is collected from 9 different people (a mix of friends and family) and their data is kept anonymous (personal data was not collected for this quiz) for security reasons. However, a rough age demographic would be somewhere around 17-24.

Chart, pie chart

Description automatically generated

I wanted to get a general idea of what the potential users perceived as a ‘best’ feature in a maze game. Whilst most voted for the addition of a timer/clock, a third of the total responses were a level-based game. Despite my objections towards using a level-based game, I want to try create a more engaging game for multiple users and I believe using a level-based game isn’t the optimal approach for achieving this goal. Furthermore, lots of the responses in timer/clock suggest that this is an important feature in the game and will likely be included in the final solution.

Chart, pie chart

Description automatically generated

For this question I wanted to understand what the potential users thought of maze games and what cognitive skills users apply when completing them. A widely known, quintessential maze solving skill, Problem Solving, received the highest number of responses. I was interested to find that a third of responses though speed is the most used skill in solving mazes. This could suggest these users play competitively, with a mental challenge to solve mazes as quickly as possible, further implying the introduction of a timer/clock would be an excellent fit into the maze. Addressing the problem-solving responses, I believe I can achieve this through ensuring a rigorous verification process that will certify that a maze is up to a certain difficulty.

Chart, pie chart

Description automatically generated

A much-expected response for this question as this solidifies my initial points made that the game requires a distinctive feature that keeps the user engaged with the game for longer.

Chart, pie chart

Description automatically generated

I wanted to investigate the responses for this question as I was already intending to add a timer/clock to the game to increase the competitive nature of the maze. Two-thirds suggested that games require this aspect to be entertaining, and I fully support this response. Even for single player games (where the user typically plays a character in a story on their own and they progress through the story and the game simultaneously), there is a competitive aspect that is introduced. Whether this is through enemies or increasing difficult game-environment, the game becomes more challenging, and the player must overcome this challenge to progress. Through using the timer/clock I can achieve all of this from implementing this into the game, as the user will feel as they are the challenge they need to overcome (by beating their own time).

Chart, bubble chart

Description automatically generated

From my initial research, lots of the games were easy and simple, so I thought this was a necessary question to be in the quiz as this would determine how the game would need to be made. All responses, to no surprise, chose the challenging option for this game, confirming that there needed to be a sense of difficulty in this game.

**Conclusion**

The survey was very informative on many different levels for this game. It showed how the users wanted the game to be built, what features they wanted in the game, whether it is essential for the game to competitive and how difficult they wanted the game to be.

From the survey, the users wanted a competitive game. This would be achieved using a timer/clock to challenge the user to complete the maze as fast as they can. Furthermore, a dynamic maze game should be created, the users believe that problem-solving is the most used cognitive ability when solving mazes. This is achieved by ensuring that every game is unique and that no two mazes are the same (there is a finite permutation of mazes created but this number is astronomically large and will be very statistically unlikely to occur). Lots of the users felt that the game required a competitive edge to be entertaining, which is a main objective as games are intended to be entertaining, therefore, adding this competitive edge via a timer/clock mechanism should address this. Finally, the users wanted the game to be challenging which is important for sustaining the game long term as this will increase the re-playability of the game as users will enjoy this challenge for longer.

In conclusion, the survey provided a good basis for me to build this project upon by providing real end-user information about the game and the surrounding topics.

**Requirements**

|  |  |  |
| --- | --- | --- |
| *Requirements* | *Description* | *Priority* |
| The maze game contains two modes | Maze solving and maze creation. Maze solving involves the user being presented with an interactable maze and they must solve it  Maze creation involves the user creating a maze and either the user or the computer can solve it (user decision) | **High** |
| The maze game uses a GUI (graphical user interface) for a human computer interaction | The user will be presented with a graphical screen that the user interacts with. | **High** |
| The maze game uses menus to navigate around the game | The user can interact with main menu screens to play the game. | **High** |
| The menu (non-playable) screens don’t contain any interactive buttons. | The menu screens will contain images that contain the text that the user will interact with using keyboard inputs. The program contains no interactive buttons on screen. | **High** |
| The game will use a default colour theme. | Default theme will be white text for the menu screens, walls will be white lines/squares, start point is a blue square, character’s position is green square, end point is a red square. Win screens and timers are green text, while controls are likely to be images (white). All is displayed on a black screen as this is the best colour to improve eligibility of the text. | **Low** |
| The Maze generation algorithm is dependent on the user input. | The generation algorithm takes the user input as a parameter to meet the users maze requirements. | **High** |
| An algorithm is needed to be added to verify the difficulty of the maze | This will run in the background to check that a maze is of a certain difficulty before it is shown to the user | **High** |
| The user will be shown their time on the maze | If the user solves either the maze solving or the maze creator, their time is shown after they have solved the maze and during the solving process. | **High** |
| The user interacts with the application to navigate the maze. | The user will use the directional keys on the keyboard to navigate the maze. | **High** |
| The User is shown the controls and what each symbol means. | The game screen will be divided into two parts with approximately 5/6 of the screen devoted to the game and playable area and the other 1/6 will be information including controls, the timer, and the meaning of symbols used. | **High** |
| The user can exit the maze at any given time | Regardless of the game mode, the user does not need to finish the maze and can use the ESC key on the keyboard to return to that game mode’s main menu (can return to main menu from there). | **High** |
| The user can choose a grid size | The user will have to choose a grid size upon to build their maze. | **High** |
| The Maze creation uses mouse clicks to interact with the maze. | The user is presented with a grid filled with walls, using a left mouse click, they can remove a wall in the position their mouse is on the screen. They can click on blocks where walls do not exist to add one there. | **High** |
| The user must confirm the completion of their maze. | The user can click ‘c’ to confirm their maze and is asked to place a start point and end point where they must click ‘c’ to progress, respectively. | **High** |
| The user can navigate the maze if this option is chosen | This works exactly as the maze solving algorithms, the user is given the same controls and timer to solve the maze created by the user. | **High** |
| The shortest path algorithm is visualised. | The user can watch the computer solve the algorithm in real time, watching every decision it makes, whether it goes down the wrong path or solves it first time, the user will be able to see this. | **High** |
| After any mode has been completed, the user is presented with a ‘win’ screen | The ‘win’ screen shows the user how to replay that mode or possibly return to the main menu | **High** |

**User Based Functions**

For the Maze Creation, a requirement is that the user can interact with the walls and remove/add them. The user will interact with a grid of squares conjoined in a user defined rows and columns grid. Interacting with the maze will involve the user using a left mouse click to remove a wall and a left mouse click again to add a wall. Every wall in the grid is its separate node and object, therefore, I can easily obtain information about where the node is, its neighbours, whether it is a wall or not or even the colour of the wall. The user, after they have completed their maze, will use the keyboard input ‘c’ to confirm their maze, and then will be asked to choose a start point using left mouse click (works the same when interacting with walls). Confirming using ‘c’, the user now must choose a node where the end/exit is to be, likewise, this works exactly like choosing the start point. After a verification process involving that a start and end point exists in the maze, both points are not on a wall, and that the start point is not on the end point, the maze is classed as complete so that the shortest path algorithm can be implemented onto it.

Manual navigation of the maze can be implemented by using the directional keys on the keyboard. When a user presses one of these keys, assuming there is no wall in between them and the next position, the user can travel to that position.

In the Maze Solving section, a requirement is that the user chooses a difficulty. Whether this be Easy, Medium or Hard, the maze should adapt to this change and this change is applied when the user plays the game. Generally, the bigger the maze the harder it is, this is because there are more decision points in the maze that could lead to a dead end earlier and unless the player knows the maze, the player will not know what path is correct. Likewise, if the maze is small, it is easy to find a path between the two points as the whole maze can be seen and analysed quickly. Furthermore, the position of the goal must be within a certain range in the bottom section of the maze. For example, if the maze is a 30x30 grid, then an ideal range for the goal should be between 25 and 30 for both the row and the column, in the bottom right of the grid. This should increase the overall difficulty of the maze regardless of whether the maze is easy, medium or hard as the end-users’ survey suggested that the game should be challenging (see Communication with End-Users). To add another touch, an algorithm should be implemented to find the maximum nodes required to solve the maze, if this value is above 90% for example, then the maze is of that difficulty chosen, if not, the algorithm will have to keep repeating until a maze is created that fits that criteria.

I believe if all these features are implemented, I can distinguish difficulties of mazes evidently and create a maze that is of a challenging status.

**Proposed Solution**

Aiming to fix these problems in this specific gaming genre, I propose to create a multi-purpose maze game. Using a maze generation algorithm, this allows me to create millions of permutations of mazes at once. The ability to create completely random mazes that are always unique can prevent users from finding maze games repetitive and uninteresting. Upon achieving this maze, the user can solve the maze and navigate between the walls finding a path from the start to the end (both defined), using the directional keys. To add a competitive nature, I intend to include an on-screen timer, counts in seconds and milliseconds, that makes the user feel as if they must act quickly and accurately to solve the maze. I believe this would work better than using a level-based system as this becomes very tedious and the user can try to get a faster time on the mazes. Using a simple difficulty-based system of easy, medium and hard, mixing this with a new maze for every difficulty, the user becomes quickly aware of achieving a goal of beating times to complete a maze over a slow, progressively difficult maze system.

To further broaden the horizon of my game, I decided to involve a creation aspect to the game. The user is presented with a grid of known size and can interact with the grid using mouse left click processes. The user utilises this mechanism to ‘fill’ a wall in or ‘remove’ a wall. By placing a start point and an end point using the same mechanism, the user confirms the completion of their maze and can advance further. The user can choose to see a shortest path algorithm solve their maze. I chose to include the shortest path algorithm to my game due to the practicality. Whilst this is a game, the shortest path algorithm can be used in a variety of different scenarios such as routing or finding the shortest path between 2 points. Future uses of this algorithm are endless hence why I choose to add this element to my game. Expanding on the entertainment aspect, the user can also give the computer to another player and let them solve the maze the user creates. As this is difficult to generally complete, verifications such as a route between the 2 points must be defined/exist (can be confirmed through using a graph traversal algorithm). Exactly the way the user will solve a computer-generated maze, the user will now solve a user-created maze. This includes all the same key presses and the onscreen timer. Furthermore, the user can play the same maze again, increasing a competitive aspect to this section of the game.

The project is primarily based on the user’s entertainment of the underlying algorithms and could be further expanded to store data, however, in further updates the game will be built to support this function of exporting data to databases like MySQL, although currently, the focus will be to create the game in the base format with potential to expand.

**Maze Solver Flowchart:**

This is an initial flowchart of the main processes a user will experience when the program runs, and the user chooses the maze solver option. Using only keyboard inputs, the user can navigate through the menu, choosing a difficulty by picking a number that corresponds to the difficulty chosen. The computer uses this information to generate a maze based off this difficulty. If the user cannot complete the maze, they can choose to exit the maze and return to the Maze Solver Menu. Although, if the user can complete the maze, a win screen is shown, showing the user the time taken for them to complete the maze. Now the user can choose to return to the Main Menu or the Maze Solver Menu to retry a new maze.

**Diagram

Description automatically generated**

**Maze Creator Flowchart:**

For the second aspect of my game, the user is presented with a menu screen that allows the user to choose a grid size. Using keyboard inputs, like Maze Solver, the user chooses a grid size that they want to draw their maze on. The grid is generated, and the user interacts using left mouse clicks to remove or add a wall. The user can confirm their maze and subsequently enters the starting point and ending point both also using left mouse click. Through the completion of these steps, the user is given a choice to let another player solve the maze or let the computer solve the maze. If the computer is chosen, the shortest path algorithm is implemented into this maze to find the shortest path between the user defined start point and end point. However, if the user decides that the player will solve it, this will work the exact same way that Maze Solver will work, using the directional keys on the keyboard. The user will also be timed on how fast they can solve the maze. The user is presented with a win screen that shows the time taken to complete the maze. Regardless of the decision made (computer or user), the user can return to the Main Menu or try to create another challenging maze.

**Diagram

Description automatically generated**

**Initial Concept / User Interface**

An essential part to any programming project that involves a human-computer interaction, is the user interface, or more specifically, how the user interacts with the software, how the user sees the software and how the user can navigate the software.

**Interacting with the software**

The user interacts with the software using keyboard presses and mouse clicks. Through some initial testing, I found that using buttons using pygame’s inbuilt button functions is not as optimal as I thought; this could’ve been due to several issues including the computer I was using. Regardless, I decided to use keyboard presses instead to navigate the game, if the user wants to solve a maze on easy difficulty for example, they can press ‘1’ on the keyboard and this will generate that maze.

**UI Graphics**

The game will use text that stands out to the user and will be easy navigate. Menu screens will involve a basic design involving a logo/title area followed by the menu options underneath this (see Initial Concepts for User Interfaces). I aim to use Procreate (a software on iPad) to create designs for the user interface. Menu Screens will be designed in Procreate and in-game hints like controls will be also created in procreate. All other text-based screens, like win screens and in-game features like the timer and the control hints, will use pygame’s own graphics.

**Main Menu**

Maze Game

2 – Create

1 – Solve

This is an initial design for the Main Menu, the text is likely to change, and colours are subject to change but as for an initial design/layout, I think this represents the simplicity of the game and is neat. Alternatively, I could choose to add a better background like that of a maze however this can be decided later.

**Maze Solving Menu**

Solve

2 – Medium

1 – Easy

3 – Hard

ESC – Return to Main Menu

This design follows on from the Main Menu design, using the same keypresses design. Like the main menu, I plan to change the text with a possibility of changing the background.

**Maze Creation Menu**

Create

2 – 20x20

1 – 10x10

3 – 30x30

ESC – Return to Main Menu

Maze creation uses the same design as the maze solving menu. Layout is subject to change; however, this is unlikely. The options for the grids are used instead of the difficulty of the maze as this is more relevant information for the user.

**Maze Solving (Self-Constructed)**

**A picture containing diagram

Description automatically generated**

This is the possible maze solving design, the walls are lines created using pygame’s line graphics. The section on the side of the screen is to be dedicated to things like the timer and in-game hints and controls. The blue square is the starting square, red square is the goal, and the green is the position of the player. I believe this initial design works well with the colours and layout and will likely be kept as the final product.

**Maze Creation (Self-Constructed)**

*A picture containing text, building

Description automatically generated*

The design for the Maze creation involves a grid where the user can click on any of the squares (walls) to remove them. Like the maze solving, there is a tab on the side where I will add the in-game information like the controls and hints. This design is also very simple too which I believe is perfect for this part of the game as the user could find this confusing.