Contents

[Project Description 4](#_Toc194523204)

[Modifications 5](#_Toc194523205)

[Collaboration 6](#_Toc194523206)

[Student 1: Diana Ali Silva 6](#_Toc194523207)

[Diana’s Screenshots 6](#_Toc194523208)

[Student 2: Carlos David Urra Cabello 7](#_Toc194523209)

[Carlos’s Screenshots 7](#_Toc194523210)

[Student 3: Georgii Taisaev 9](#_Toc194523211)

[Georgii’s Screenshots 9](#_Toc194523212)

[Student 4: Muslim Muradov 10](#_Toc194523213)

[Muslim’s Screenshots 10](#_Toc194523214)

[Planned Improvements for Next Version 11](#_Toc194523215)

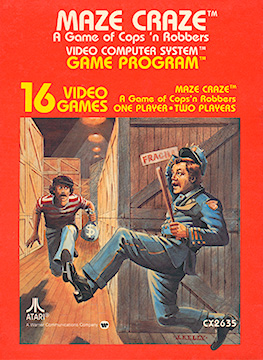
[Link to our repository: 12](#_Toc194523216)

[References 13](#_Toc194523217)



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Student name: | Student 1: Diana Ali Silva  Student 2: Carlos David Urra Cabello  Student 3: Georgii Taisaev  Student 4: Muslim Muradov | | | | | |
| Student number: | Student 1: 3123965  Student 2: 3125350  Student 3: 3119655  Student 4: 3128794 | | | | | |
| Faculty: | Computing Science | | | | | |
| Course: | BSCH/BSCO/EXCH | | | Stage/year: | 2 | |
| Subject: | Software Development 2 | | | | | |
| Study Mode: | Full time | Icon  Description automatically generated |  | Part-time |  |  |
| Lecturer Name: | Gemma Deery | | | | | |
| Assignment Title: | Review 1 | | | | | |
| Date due: | 03 April 2025 | | |  | | |
| Date submitted: | 03 April 2025 | | |  | | |
| **Plagiarism disclaimer:**  *I understand that plagiarism is a serious offence and have read and understood the college policy on plagiarism. I also understand that I may receive a mark of zero if I have not identified and properly attributed sources which have been used, referred to, or have in any way influenced the preparation of this assignment, or if I have knowingly allowed others to plagiarise my work in this way.*  *I hereby certify that this assignment is my own work, based on my personal study and/or research, and that I have acknowledged all material and sources used in its preparation. I also certify that the assignment has not previously been submitted for assessment and that I have not copied in part or whole or otherwise plagiarised the work of anyone else, including other students.*  **Signed: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** | | | | | | |
| **Please note:** Students **MUST** retain a hard / soft copy of **ALL** assignments as well as a receipt issued and signed by a member of Faculty as proof of submission. | | | | | | |

# Project Description

A box with a game

AI-generated content may be incorrect.The 2D Multiplayer Maze Game is a Java application inspired by the classic arcade games *Amazing Maze (1976)* and *Maze Craze (1980)*. The game challenges two players to race through a randomly generated maze to reach the exit first. Each maze is built using a recursive backtracking algorithm to ensure it is always solvable and offers a fresh layout in every round.[1,2]

*Maze Craze (1980).*

*Amazing Maze (1976).*

A screenshot of a computer

AI-generated content may be incorrect.Originally developed as a console-based game using ASCII characters, the project has now started transitioning into a graphical version using Java Swing. This GUI aims to enhance the player experience with better visuals and interactive elements. Local multiplayer is a key feature, allowing both players to compete using the same keyboard. Sprite integration and GUI components are currently being developed, and JUnit testing will be implemented to ensure the game runs reliably as more features are added.

*Console Output of the project.*

# Modifications

Several changes and improvements have been made since we started the project. The general structure of the game, including the folder organization, the maze generation algorithm, and the basic game logic, was set up at the beginning. These parts allowed us to test the game early and gave a strong base to build on.

Diana created a Java Swing window that opens when running the class. For now, it only shows a gray screen, but it’s the first step toward moving the game from the console to a graphical interface. Later, this window will be connected to the main game to make the experience smoother and more interactive.[4]

Georgii added some basic sprites to the maze. At the moment, these are shown as the letter 'x' in the console, but they already help make the game more challenging. These placeholders will be replaced by proper graphics once the Swing version is ready.

Muslim worked on the local multiplayer feature. Right now, both players use the same keyboard but cannot move at the same time, since the game runs in the console. However, we plan to support simultaneous gameplay once the game is fully integrated with Java Swing.

# Collaboration

## Student 1: Diana Ali Silva

During the development of the maze game, I spent time researching the best way to implement the Graphical User Interface (GUI). After discussing options with my lecturer, Tracey Cassels, we agreed that Java Swing would be the right choice for this stage. It is simpler to work with than JavaFX and better suited for where we are in the learning process. I have since built a basic game window with Swing, setting up the title, icon, size and background colour to create a clean starting point.

However, in attempting to integrate the maze game logic into the GUI, I faced a number of issues with Git. Specifically, some of my commits were present locally but not reflected in the remote repository. To synchronise the local and remote branches, the commits had to be merged together. After addressing these merge conflicts, I encountered another issue in which Git stated that there were no changes to commit despite changes being made to the code. This has momentarily slowed development because I cannot submit changes until the commit history is fully synced.

Moving forward, my immediate plan is to address these issues to ensure a smooth workflow. This involves resolving any discrepancies between local and remote branches, and making sure future commits are properly synchronised. Once these issues are resolved, the next step will be to fully integrate the maze game into the GUI. This includes connecting the game logic to the visual components and ensuring smooth interaction between the two. Additionally, I will review the code for any errors or inefficiencies, particularly in how the game renders with the GUI integration.

The goal is to have a functional, visually cohesive maze game with a properly working GUI in the next phase of development.

### Diana’s Screenshots

A screen shot of a computer

AI-generated content may be incorrect.

*Base code for the GUI .*

**A screenshot of a computer program

AI-generated content may be incorrect.**

*Code in attempt to integrate game logic.*

**A screenshot of a computer

AI-generated content may be incorrect.**

*Simple GUI window with gray background, logo and generic game name.*

**A screen shot of a computer program

AI-generated content may be incorrect.**

*Unable to commit due to git not recognising any changes made.*

****

*Merge pull request to synch local and remote commits.*

## Student 2: Carlos David Urra Cabello

At the start of the project, I created a GitHub repository to help the team collaborate more easily and work remotely. This also made it easier to track changes and manage versions.

I developed the **random maze generator algorithm** using recursive backtracking, which creates a new and solvable maze each time the game runs. I also implemented basic game logic, including single-player movement and a message that appears when the player reaches the exit.[3]

Lastly, I organized the folder structure to follow the layout from the example shared on Moodle. This helped make the project easier to manage and prepare for future updates like the GUI and multiplayer features.

### Carlos’s Screenshots

A close up of a text

AI-generated content may be incorrect.

*Github repository*

A black and white maze

AI-generated content may be incorrect.

*randomly generated maze*

*A screenshot of a computer game

AI-generated content may be incorrect.*

*Exit message shown when the player successfully reaches the end of the maze.*

*A screenshot of a computer program

AI-generated content may be incorrect.*

*Organized folder structure based on the example provided on Moodle.*

## Student 3: Georgii Taisaev

### Georgii’s Screenshots

## Student 4: Muslim Muradov

### Muslim’s Screenshots

# Planned Improvements for Next Version

For the next stage of the project, our goal is to move from a console-based experience to a more complete and interactive graphical game. One of the main updates will be to fully integrate the Java Swing interface with the game logic, allowing the maze and player movement to be displayed in a window with basic visual elements.[4]

We plan to replace the current console output with a 2D view of the maze, where walls, paths, and players are drawn using simple graphics or placeholder sprites. This will improve the overall user experience and make the game easier to understand and play.

Another important update is to support simultaneous local multiplayer. Right now, players take turns moving through the maze, but in the next version, both players should be able to move at the same time using different keys on the same keyboard. This will make the gameplay more dynamic and competitive.

Other improvements may include adding a simple menu or game start screen, using Swing buttons or key bindings, and possibly updating the placeholder sprites to better represent obstacles or player characters.

These updates will bring the project closer to a fully playable and visually interactive game.

### Link to our repository:

https://github.com/Carlos-0620/Software\_Development\_Project.git

## References

1. Midway Manufacturing Co. (1976). Amazing Maze [Arcade Game]. International Arcade Museum. Retrieved from https://www.arcade-museum.com/Videogame/amazing-maze
2. Atari. (1980). Maze Craze [Atari 2600 Game]. Atari Official Website. Retrieved from <https://atari.com/pages/mazecraze>
3. Buck, J. (2015). Mazes for Programmers: Code Your Own Twisty Little Passages. The Pragmatic Bookshelf. ISBN: 978-1-68050-055-4
4. GeeksforGeeks. (n.d.). Introduction to Java Swing. Retrieved from https://www.geeksforgeeks.org/introduction-to-java-swing/