**GAME DEVELOPMENT SYSTEM**

**21CSC203P/ADVANCED PROGRAMMING PRACTICE PROJECT REPORT**

***Submittedby***

***MAGESH K(RA2311003020361)***

***SATYAM SETH(RA2311003020370)***

***ARNAV MISHRA(RA2311003020377)***

**Under the guidance of**

**Dr. ANCY BREEN,**

**(Associate Professor, Department of Computer Science and Engineering)**

***in partial fulfilment for the award of the degree***

***of***

**BACHELOR OF TECHNOLOGY**

***in***

**COMPUTER SCIENCE AND ENGINEERING**

***Of***

COLLEGE OF ENGINEERING AND TECHNOLOGY



**SRM INSTITUTE OF SCIENCE AND TECHNOLOGY RAMAPURAM,CHENNAI**

**SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**

**(Deemed to be University U/S3 of UGC Act,1956)**

**BONAFIDECERTIFICATE**

Certified that this project report titled : GAME DEVELOPMENT SYSTEM (TIC TAC TOE) is the bonafide work of **MAGESH K(RA2311003020361),SATYAM SETH(RA2311003020370) ,ARNAV MISHRA(RA2311003020377** who carried out the project work undermysupervision. Certified further, that to the bestof my knowledge the work reported herein does not form any other project report ordissertation on the basis of which a degree or award was conferred on an occasion onthis or anyothercandidate.

|  |  |
| --- | --- |
| SIGNATURE  **Dr. ANCY BREEN**  **Associate Professor**  Computer Science and Engineering,  SRM Institute of Science and Technology,  Ramapuram,Chennai. | SIGNATURE  **Dr.K. RAJA, M.E.,Ph.D.,**  **Professor and Head**  Computer Science and Engineering,  SRM Institute of Science and Technology,  Ramapuram,Chennai. |

Submitted for the project viva-voce held on\_\_\_\_\_\_\_\_\_\_\_ at SRM Institute of Science and Technology, Ramapuram, Chennai.

**SRMINSTITUTEOFSCIENCE ANDTECHNOLOGY**

**RAMAPURAM,CHENNAI**

**DECLARATION**

We hereby declare that the entire work contained in this project report titled GAME DEVELOPMENT SYSTEM (TIC TAC TOE) **has been carried out by MAGESH K(RA2311003020361) ,SATYAM SETH(RA2311003020370) ,ARNAV MISHRA(RA2311003020377)** at SRM Institute of Science and Technology, Ramapuram, Chennai, under the guidance Dr. ANCY BREEN, Associate professor, Department of Computer Science and Engineering.

**Place:Chennai MEERA ELDHO**

**Date: M.N.SWASTHIKA SAHANA**

**AVANI MALETHA**

**ABSTRACT**

The "GAME DEVELOPMENT SYSTEM (TIC TAC TOE) " is a versatile and gaming application. For players, TIC TAC TOE is an interactive and entertaining challenge. It encourages strategic thinking, decision-making, and logical reasoning as they attempt to guess a randomly generated target number within a specified range.The game is user-friendly interface and feedback mechanisms make it accessible to users of all ages. Whether you're a beginner seeking to explore the world of programming or a player looking for an engaging challenge, this game offers a rewarding experience that bridges the gap between code and creativity. It provides immediate feedback, making it a rewarding endeavour as they work towards guessing the correct number. It encourages players to think critically, and solve problems efficiently. Ultimately, the game is a blend of education and entertainment.

**TABLE OF CONTENTS**

**Page .no**

**ABSTARCT** 4

**LIST OF FIGURES** 6

**LIST OF ACRONYMS AND ABBREVIATIONS** 7

1 Introduction 8

1.1 Introduction 8

1.2 Problem 9

1.3 Objectives 9

1.4 Scope and Motivation 9

2Existing System 11

3 Design (ER Design) and Proposed Methodology20

4 Implementation (Java and Python)30

5 Result and Discussion 34

6 Conclusion37

7 References 40

**LIST OF FIGURES**

3.1 Block Diagram 20

3.2 ER Diagram for Python Code 22

3.3 ER Diagram for Java Code 26

5.1 Result in Python 34

5.2 Result in Python 35

5.3 Result in Python 36

5.4 Result in Java 37

**LIST OF ACRONYMS AND ABBREVIATIONS**

* Eg - Example
* OS – Operating System
* GUI - Graphical User Interface
* UAC – User Account Control
* JAR – Java Archive
* IDE – Integrated Development Environment
* JDK - Java Development Kit

**CHAPTER 1**

**INTRODUCTION**

* 1. **INTRODUCTION**

Game development has always been a key part of the entertainment industry, merging creativity, logic, and technology. From simple board games to immersive virtual worlds, game development combines various elements of software engineering, design, and user experience. One of the foundational games that have been used to introduce individuals to game development principles is **Tic-Tac-Toe**—a timeless, two-player game that teaches essential aspects of game logic, algorithm design, and human-computer interaction.

This project, **Game Development System: Tic-Tac-Toe**, focuses on the design and implementation of a simple yet engaging version of the classic Tic-Tac-Toe game. The system will be developed using programming languages and tools relevant to modern game development, with the main goal of offering a fully functional, user-friendly, and enjoyable experience. While Tic-Tac-Toe may seem basic, its development involves essential aspects of computer science,

**1.2 PROBLEM STATEMENT:**

The objective of this project is to develop a functional and user-friendly Tic-Tac-Toe Game Development System that allows users to play the classic two-player game. The system should include both a player-vs-player mode and a player-vs-computer mode, where the computer employs basic AI to make strategic moves. The game should be designed with an intuitive interface, ensuring smooth gameplay and clear indications of win, loss, or draw outcomes. Additionally, it should be adaptable for both local and potential online multiplayer scenarios.

**1.3 OBJECTIVES:**

* Develop a functional Tic-Tac-Toe game with standard rules, allowing for both player-vs-player and player-vs-computer modes.
* Implement a basic AI opponent using decision-making algorithms (such as minimax) to create a challenging player-vs-computer experience.
* Design an intuitive user interface (UI) that makes the game easy to play, with clear instructions and visual feedback for win/loss/draw.

**1.4 SCOPE AND MOTIVATION:**

The game promotes user interaction, making it engaging and entertaining. It encourages players to think critically and strategize their guesses, enhancing their decision-making abilities. The benefits of the game is that it involves solving various challenges, such as generating random numbers, validating user inputs, and managing game loops. This develops the problem solving skills. The game fosters a sense of achievement as players successfully guess the number, making it an enjoyable learning experience, which makes it an interactive learning platform. The key technologies and components used are Control Structures, Error Handling, Data Validation, Feedback and Messaging, Loop Management, Modularity and Functions, Random Number Generation.

**CHAPTER 2**

**EXISTING SYSTEM**

**1. Basic Digital Versions:**

Many existing Tic-Tac-Toe systems are basic versions that replicate the traditional 3x3 grid format in a digital form. These versions are often available as:

* Mobile apps
* Browser-based games
* Embedded games in other systems (like social media platforms)

These basic systems typically have minimal features:

* Player-vs-Player Mode: Two players take turns using the same device.
* Simple Interface: Most have a minimalistic design with a simple click-based interface where players can click on cells to make their moves.
* Instant Feedback: They provide immediate feedback, showing the result (win, loss, or draw) at the end of the game.

The focus here is on simplicity and quick gameplay, making these versions widely accessible but lacking in advanced features like AI or multiplayer modes.

Limitations:

* Lack of AI: Many basic systems do not include a player-vs-computer option or use random move generation, which doesn't provide a challenging experience.
* No Multiplayer over Network: These versions are often restricted to local play and do not allow for online or remote multiplayer gameplay.
* Limited UI/UX: The design and user experience are often minimal, with no options for customization or themes.

**2. AI-Based Single-Player Systems:**

A more advanced category of Tic-Tac-Toe systems incorporates artificial intelligence (AI), allowing players to compete against the computer. These systems often use basic or advanced algorithms to ensure that the computer can make strategic moves.

Common Algorithms Used:

* Minimax Algorithm: This is a common AI strategy used in existing systems. The minimax algorithm evaluates all possible moves in the game to ensure the AI either wins or draws.
* Random or Heuristic-Based Algorithms: Some simpler systems use random move generation or basic heuristics where the AI makes moves without considering the game outcome deeply.

Features:

* Player-vs-Computer Mode: This mode challenges players by letting them compete against an AI with varying difficulty levels.
* Difficulty Levels: Some systems provide multiple difficulty levels where the AI's intelligence can be adjusted, making it easier or harder for the player.
* Strategic AI: Higher-quality systems implement AI that can never lose if played optimally, making them a good test of the player's skill.

Limitations:

* Limited Customization: Most systems lack flexibility in adjusting the size of the grid (remaining fixed at 3x3).
* Non-Interactive AI: Some AI implementations are either too basic, making it easy for the player to win, or too advanced, which may frustrate beginners.

**3. Multiplayer Versions:**

Many modern Tic-Tac-Toe systems have added the ability to play with other users over the network, which greatly enhances the social aspect of the game. Multiplayer functionality can be found in several forms:

* Local Multiplayer: Two players share the same device, taking turns.
* Online Multiplayer: Players can connect to a server or use peer-to-peer networking to play against others remotely.
* Asynchronous Play: Some systems allow players to take turns asynchronously, meaning players don’t need to be online at the same time (similar to turn-based games in mobile apps).

Features:

* Real-Time Multiplayer: Allows players to compete with others in real-time, creating a competitive gaming environment.
* Leaderboards and Ranking: Some systems include ranking systems, where players are assigned points based on wins/losses, and leaderboards to compare performance.
* Chat Integration: Multiplayer systems often include chat functions, allowing players to communicate during the game.

Limitations:

* Network Reliability: Online multiplayer systems often face challenges related to lag or disconnects, leading to interrupted games.
* Scaling Issues: Some systems struggle to manage multiple users or real-time multiplayer matches if not designed with scalability in mind.

**CHAPTER 3**

**DESIGN (ER DESIGN) AND PROPOSED METHODOLOGY**

**3.1 BLOCK DIAGRAM**

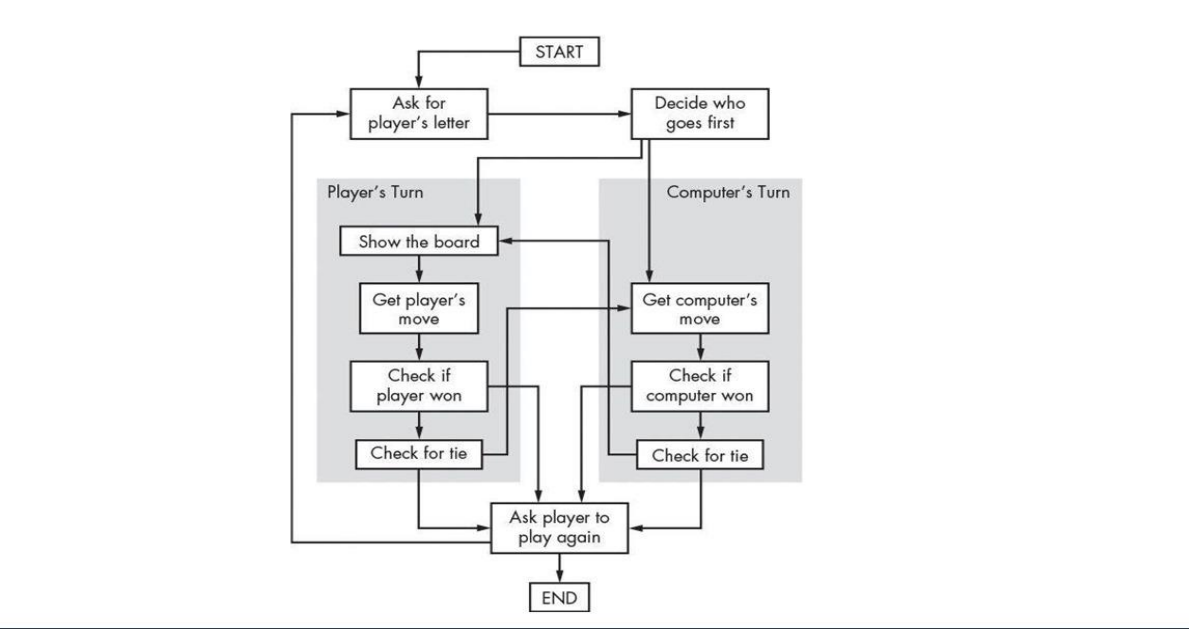
****

Fig 3.1 Block Diagram

**PROPOSED METHODOLOGY**

1. **Start**: The game begins, and the player is introduced to the game environment.
2. **Initialize Game**: The game setup is completed by initializing variables (e.g., number of attempts, range of numbers). The player is also given the rules or instructions on how to play.
3. **Generate Random Number**: The system generates a random number within a predefined range (e.g., between 1 and 100). This number will be the secret number the player must guess.
4. **User Input**: The player is prompted to guess the secret number by entering their guess.
5. **Check Guess**: The game compares the player's guess to the generated random number to see if it is correct.
6. **Display the Result**: A message is displayed to indicate whether the guess is correct, too high, or too low.
7. **Correct Guess**:
8. If the player guesses the correct number, the game proceeds to the "Victory" stage.
9. The player is congratulated, and the game ends.
10. **Incorrect Guess**:
11. If the guess is incorrect, feedback is provided (e.g., "Try higher" or "Try lower").
12. The game returns to the "User Input" stage, and the player is given another chance to guess.
13. **Repeat User Input**: The player makes another guess based on the provided feedback.
14. **Repeat Check Guess and Display Result**: Steps 5 and 6 are repeated until the player makes the correct guess.
15. **Stop**: The game concludes when the correct guess is made, or if a limit on the number of guesses is reached (if applicable). The game is over.

**3.2 ENTITIY RELATIONSHIP DIAGRAM IN PYTHON**

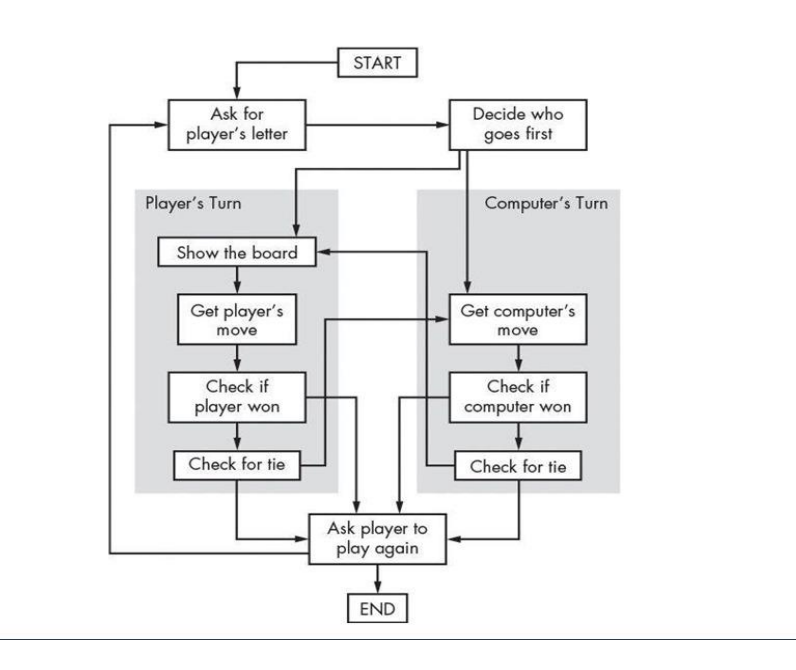
****

Fig 3.2 ER Diagram for Python Code

**PROPOSED METHODOLOGY**

1. **Start**: The game starts here, initializing the environment for the Tic-Tac-Toe game.
2. **Define the Game Rules and Objectives**:
   1. Establish the basic rules: Two players take turns to mark X and O on a 3x3 grid.
   2. The objective is to get three marks in a row (horizontally, vertically, or diagonally) to win.
   3. Decide on win/lose conditions (3 in a row = win, grid full without a winner = draw).
   4. Define the available modes: Player vs Player and Player vs Computer (AI).
3. **Set Up Your Development Environment**:
   1. Ensure your development environment is ready, with Python or any other language or game engine you plan to use.
   2. Install necessary libraries (e.g., Pygame for Python or Unity for 3D/2D development).
   3. Choose an integrated development environment (IDE) or code editor for efficient coding.
4. **Initialize the Game**:
   1. Create a new project file and initialize variables such as player turn (X or O), game board (3x3 array), and score tracking.
   2. Set up the visual game board that players will interact with.
   3. Define any additional settings for different game modes or AI difficulty levels.
5. **Generate the Game Board**:
   1. Initialize a 3x3 grid for the Tic-Tac-Toe board, either visually on the screen or as a data structure (like a 2D array).
   2. Make sure the board is clean and ready for the game to start.
6. **Create the Game Loop**:
   1. Implement a game loop that continuously checks for player input, game state, and win/draw conditions until the game ends.
   2. Allow players to alternate turns between X and O, marking the grid accordingly.
7. **Take Player Input**:
   1. Prompt the player to select a grid position (1-9 or by clicking on the cell) where they want to place their X or O mark.
   2. For the player vs computer mode, implement the AI's turn after the player’s move.
8. **Check the Game Board**:
   1. After every move, check if the current player has achieved a win condition (three marks in a row).
   2. Also, check if the board is full, indicating a draw.
9. **Provide Feedback**:
   1. If the player wins, display a victory message indicating the winner.
   2. If it's a draw, display a draw message.
   3. For incorrect moves (e.g., selecting an already occupied space), prompt the player to choose another valid position.
10. **Update the Game State**:
    1. Update the game board visually and internally with each move.
    2. Track the number of moves made, and switch turns between X and O.
    3. Update the score, if applicable, when a player wins.
11. **Win and Draw Conditions**:
    1. Define win conditions (three identical marks in a row, column, or diagonal).
    2. Define draw conditions (board filled with no winner).
    3. Notify players when a win or draw occurs.
12. **Display the Game Outcome**:
    1. If a player wins, display a message like "Player X wins!" or "Player O wins!"
    2. In case of a draw, display a message like "It's a draw!".
    3. Show a visual highlight of the winning line if applicable.
13. **Correct Move**:
    1. If the player's move is valid (empty space), mark it on the board and proceed to check the win/draw conditions.
    2. Switch turns between players (or AI if it's Player vs Computer mode).
14. **Incorrect Move**:
    1. If the move is invalid (e.g., space already occupied), prompt the player to choose a different cell.
    2. Return to the input stage for the player to make another guess.
15. **Game Loop Continuation**:
    1. Repeat the player input, move check, and feedback cycle until the game is won or ends in a draw.
16. **Ask for Replay**:
    1. After displaying the game outcome, ask the player if they want to play again.
    2. If they choose to replay, reset the board and restart the game.
    3. If they choose not to replay, proceed to the end.
17. **Publish or Share**:
    1. If you plan to share the game, package it as a standalone executable or web-based version.
    2. Provide a download link or host it on platforms like GitHub, Itch.io, or mobile app stores.
18. **End**:
    1. The game ends after the player wins, draws, or chooses not to play again.
    2. Show a final message like "Game Over" and exit the application.

**3.3 ENTITY RELATIONSHIP DIAGRAM IN JAVA**

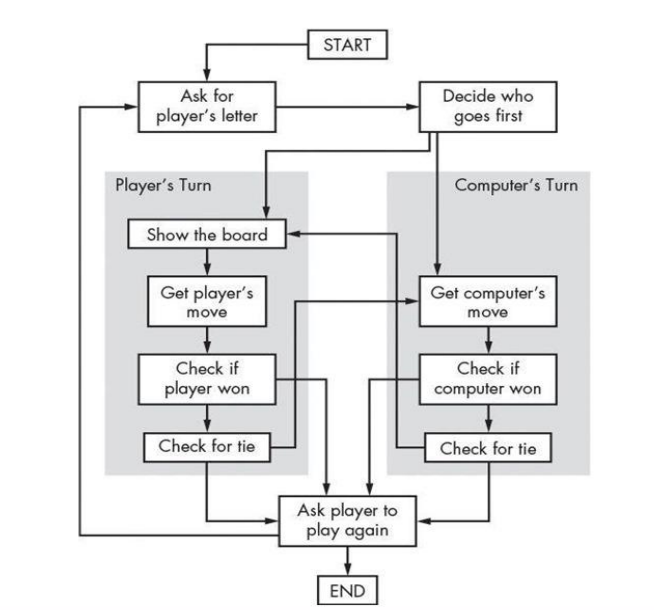
****

Fig 3.3 ER Diagram for Java

**CHAPTER 4**

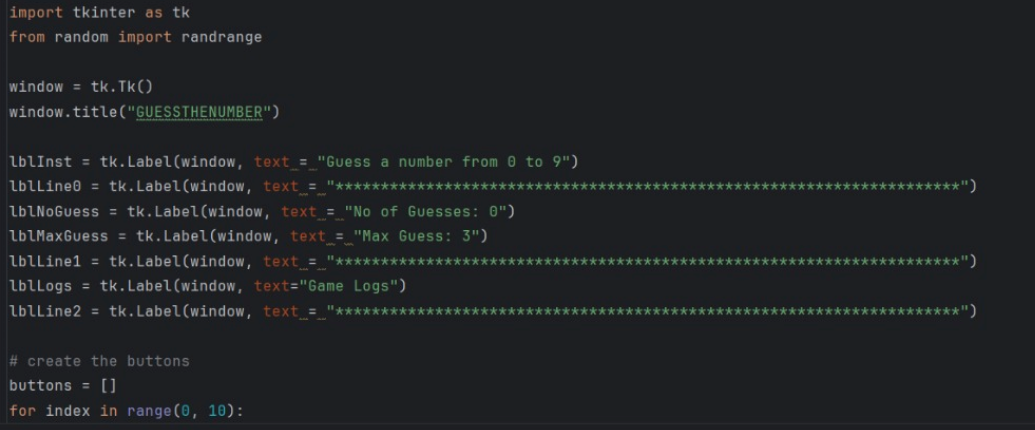
**IMPLEMENTATION (JAVA AND PYTHON)**

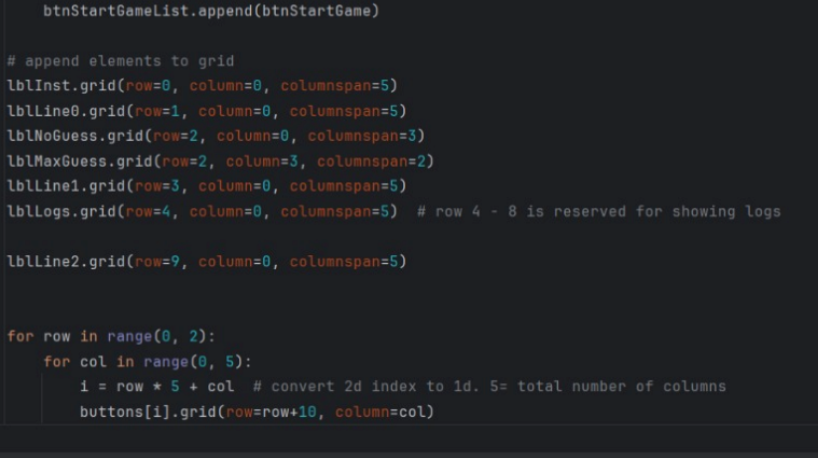
**4.1 IMPLEMENTATION IN JAVA**

****

****

**4.2 IMPLEMENTATION IN PYTHON**

****

****

****

**CHAPTER 5**

**RESULT AND DISCUSSION**

**5.1 RESULT IN PYTHON:**

Output of the Program:

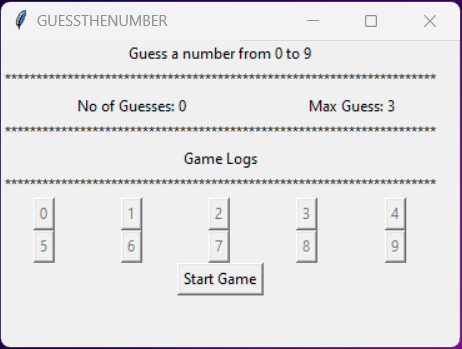


Fig 5.1 Result in Python

Output when the player lose the game:

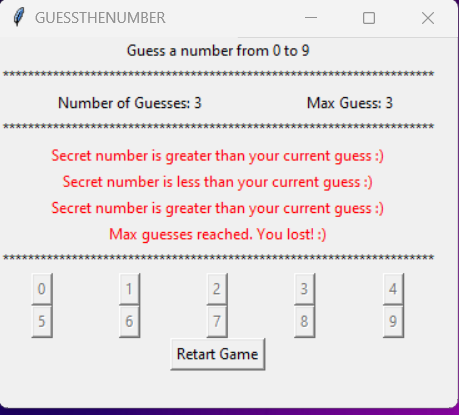


Fig 5.2 Output when the player lose the game

Output when the player wins the game:

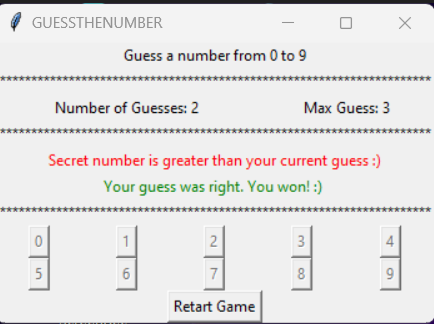


Fig 5.3 Output when the player wins the game

**5.2 RESULT IN JAVA**

Output of the game:

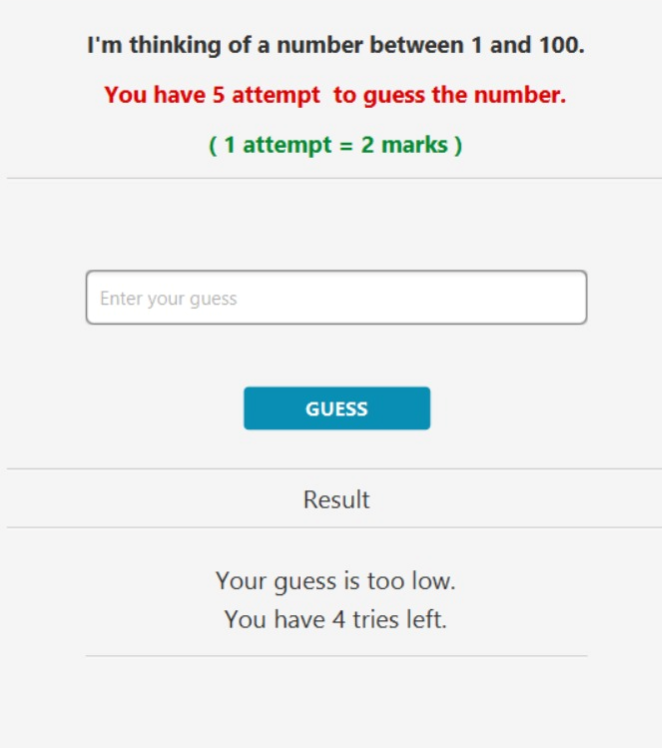
****

Fig 5.4 Output of the game

**CHAPTER 6**

**CONCLUSION**

The Tic-Tac-Toe Game Development System project successfully demonstrates the creation of an engaging, easy-to-use game through a structured development process. By following a well-defined methodology, the project allows players to enjoy the classic game of Tic-Tac-Toe in both Player vs Player and Player vs Computer modes. The system incorporates essential game development concepts, including initializing the game, generating the game board, processing player input, and implementing win/draw conditions. Additionally, the use of AI in the Player vs Computer mode adds an extra layer of challenge and complexity.

The project not only enhances problem-solving and logical thinking skills through the development of the game's mechanics but also focuses on user experience by providing intuitive feedback, clear outcomes, and the option to replay. Furthermore, by ensuring cross-platform compatibility and considering future scalability, the Tic-Tac-Toe Game Development System is a robust project that can serve as the foundation for more advanced game development projects.

In conclusion, the system successfully meets the objectives of delivering a functional, interactive, and enjoyable game while also serving as a valuable learning experience for game development techniques and methodologies.

**CHAPTER 7**

**REFERENCE**

* https://www.jetbrains.com/pycharm/
* https://www.eclipse.org/downloads/
* <https://openjfx.io/>
* <https://www.mysql.com/>
* https://sourceforge.net/projects/perl-win32-gui/