NUMBER GUESSING GAME

A PROJECT REPORT

Submitted by

P.SURYA (2303811710421163)

in partial fulfillment of requirements for the award of the course

CGB1201 - JAVA PROGRAMMING

In

COMPUTER SCIENCE AND ENGINEERING

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(An Autonomous Institution, affiliated to Anna University Chennai and Approved by AICTE, New Delhi)

SAMAYAPURAM – 621 112

NOVEMBER- 2024

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY (AUTONOMOUS)

SAMAYAPURAM – 621 112

BONAFIDE CERTIFICATE

Certified that this project report on **“ NUMBER GUESSING GAME”** is the bonafide work of **SURYA P (2303811710421163)** who carried out the project work during the academic year 2024 - 2025 under my supervision.

CGB1201-JAVA PROGRAMMING Dr.A.DELPHIN CAROLINA RANI, M.E.,Ph.D., HEAD OF THE DEPARTMENT PROFESSOR [illegible]	CGB1201-JAVA PROGRAMMING Mr.MALARMANNAN A, M.E., [illegible] ASSISTANTP ROFESSOR [illegible]
SIGNATURE	**SIGNATURE**
Dr.A.Delphin Carolina Rani, M.E.,Ph.D.,	Mr. A. Malarmanna M.E.,
HEAD OF THE DEPARTMENT	**SUPERVISOR**
PROFESSOR	ASSISTANT PROFESSOR
Department of CSE	Department of CSE
K.Ramakrishnan College of Technology (Autonomous)	K.Ramakrishnan College of Technology (Autonomous)
Samayapuram–621112.	Samayapuram–621112.

Submitted for the viva-voce examination held on 06.12.2024

CGB1201-JAVA PROGRAMMING Mr. M. SARAVANAN, M.E., INTERNAL EXAMINER ASSISTANT PROFESSOR [illegible]	CGB1201-JAVA PROGRAMMING Mr.R. KARTHIK, M.E., EXTERNAL EXAMINER ASSISTANT PROFESSOR 8138-SCE, TRICHY. [illegible] 06/12/2024
INTERNAL EXAMINER	EXTERNAL EXAMINER

DECLARATION

I declare that the project report on **“NUMBER GUESSING GAME”** is the result of original work done by us and best of our knowledge, similar work has not been submitted to **“ANNA UNIVERSITY CHENNAI”** for the requirement of Degree of **BACHELOR OF ENGINEERING**. This project report is submitted on the partial fulfilment of the requirement of the completion of the course **CGB1201 - JAVA PROGRAMMING.**

.

Signature

[illegible]

SURYA P

Place: Samayapuram

Date: 06.12.2024

ACKNOWLEDGEMENT

It is with great pride that I express our gratitude and in-debt to our institution "**K.Ramakrishnan College of Technology (Autonomous)**", for providing us with the opportunity to do this project.

I glad to credit honourable chairman **Dr. K. RAMAKRISHNAN**, **B.E.,** for having provided for the facilities during the course of our study in college.

I would like to express our sincere thanks to our beloved Executive Director**Dr. S. KUPPUSAMY, MBA, Ph.D.,** for forwarding to our project and offering adequate duration in completing our project.

I would like to thank **Dr. N. VASUDEVAN, M.Tech., Ph.D.,** Principal, who gave opportunity to frame the project the full satisfaction.

I whole heartily thanks to **Dr. A. DELPHIN CAROLINA RANI, M.E.,Ph.D.,** Head of the department, **COMPUTER SCIENCE AND ENGINEERING** for providing her encourage pursuing this project.

I express our deep expression and sincere gratitude to our project supervisor **Mr. A. MALARMANNAN, M.E.,** Department of **COMPUTER SCIENCE AND ENGINEERING,** for his incalculable suggestions, creativity, assistance and patience which motivated us to carry out this project.

I render our sincere thanks to Course Coordinator and other staff members for providing valuable information during the course.

I wish to express our special thanks to the officials and Lab Technicians of our departments who rendered their help during the period of the work progress.

VISION OF THE INSTITUTION

To serve the society by offering top-notch technical education on par with global standards

MISSION OF THE INSTITUTION

- Be a center of excellence for technical education in emerging technologies by exceeding the needs of the industry and society.
- Be an institute with world class research facilities
- Be an institute nurturing talent and enhancing the competency of students to transform them as all-round personality respecting moral and ethical values

VISION OF DEPARTMENT

To be a center of eminence in creating competent software professionals with research and innovative skills.

MISSION OF DEPARTMENT

M1: Industry Specific: To nurture students in working with various hardware and software platforms inclined with the best practices of industry.

M2: Research: To prepare students for research-oriented activities.

M3: Society: To empower students with the required skills to solve complex technological problems of society.

PROGRAM EDUCATIONAL OBJECTIVES

1. PEO1: Domain Knowledge

To produce graduates who have strong foundation of knowledge and skills in the field of Computer Science and Engineering.

2. PEO2: Employability Skills and Research

To produce graduates who are employable in industries/public sector/research organizations or work as an entrepreneur.

3. PEO3: Ethics and Values

To develop leadership skills and ethically collaborate with society to tackle real-world challenges.

PROGRAM SPECIFIC OUTCOMES (PSOs)

PSO 1: Domain Knowledge

To analyze, design and develop computing solutions by applying foundational concepts of Computer Science and Engineering.

PSO 2: Quality Software

To apply software engineering principles and practices for developing quality software for scientific and business applications.

PSO 3: Innovation Ideas

To adapt to emerging Information and Communication Technologies (ICT) to innovate ideas and solutions to existing/novel problems

PROGRAM OUTCOMES (POs)

Engineering students will be able to:

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences
3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions

5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

ABSTRACT

The Number Guessing Game is an interactive and entertaining application designed to test users' logical reasoning and predictive skills. The game challenges players to guess a randomly generated number within a specified range, with feedback provided to guide their guesses. Core functionalities include dynamic difficulty adjustment, intuitive user input handling, and real-time feedback. The application is structured with modular components, ensuring flexibility and maintainability, with primary modules including Game Logic, User Interface, Input Validation, and Scoring Mechanism. The application employs a simple yet engaging graphical or text-based user interface, enabling seamless interaction. The game logic incorporates random number generation and range-based hint systems, fostering an engaging and challenging user experience. It features robust input validation and error handling to ensure reliability and smooth gameplay. The scoring mechanism tracks user performance, providing a measure of efficiency and accuracy in guessing. This Number Guessing Game serves as a fun and educational tool for users of all ages, promoting cognitive skills and problem-solving abilities while offering an engaging and enjoyable experience.

ABSTRACT WITH POs AND PSOs MAPPING

CO 5 : BUILD JAVA APPLICATIONS FOR SOLVING REAL-TIME PROBLEMS.

ABSTRACT	POs MAPPED	PSOs MAPPED
The Number Guessing Game is an interactive and entertaining application designed to test users' logical reasoning and predictive skills. The game challenges players to guess a randomly generated number within a specified range, with feedback provided to guide their guesses. Core functionalities include dynamic difficulty adjustment, intuitive user input handling, and real-time feedback. The application is structured with modular components, ensuring flexibility and maintainability, with primary modules including Game Logic, User Interface, Input Validation, and Scoring Mechanism.	**PO1 -3** **PO2 -3** **PO3 -3** **PO4 -3** **PO5 -3** **PO6 -3** **PO7 -3** **PO8 -3** **PO9 -3** **PO10 -3** **PO11-3** **PO12 -3**	**PSO1 -3** **PSO2 -3** **PSO3 -3**

Note: 1- Low, 2-Medium, 3- High

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

The Number Guessing Game is a simple yet engaging interactive program designed to develop and enhance foundational programming skills in an enjoyable way. As technology becomes increasingly central to learning and problem-solving, this game provides an effective platform for practicing key concepts in a hands-on manner. It challenges users to guess a randomly selected number, offering feedback after each attempt to guide them closer to success.

Developed using Java, the project employs a modular architecture to ensure clarity, maintainability, and scalability. It utilizes essential programming constructs such as random number generation, conditionals, loops, and user input handling. Each module, including game logic, feedback handling, and flow control, plays a critical role in creating a seamless and interactive gaming experience.

With its focus on active learning, the Number Guessing Game serves as an excellent educational tool for beginners exploring programming. It fosters problem-solving skills, logical thinking, and a deeper understanding of coding fundamentals, making it a valuable addition to any developer’s learning journey.

1.1 Objective:

The primary objective of the Number Guessing Game is to provide an engaging platform for learning and practicing core programming concepts through interactive gameplay. The project is designed to:

1. Introduce Basic Programming Skills: Enable learners to apply fundamental concepts such as random number generation, loops, conditionals, and user input handling in a practical scenario.
2. Encourage Logical Thinking: Promote analytical skills by challenging players to narrow down possibilities and strategize their guesses based on feedback.
3. Offer a User-Friendly Experience: Provide a seamless and intuitive interface for players to interact with the game, ensuring a smooth and enjoyable experience.

4. Foster Code Modularity: Demonstrate the importance of modular programming by breaking the game into manageable components like input handling, feedback generation, and game flow control.
5. Enhance Problem-Solving Abilities: Create an engaging environment where learners can test and improve their problem-solving skills while having fun.

Through these objectives, the Number Guessing Game serves as an effective tool for beginners to develop and reinforce programming knowledge in an enjoyable and rewarding way. Its scalable and flexible design ensures adaptability for various educational and learning contexts.

1.2 Overview

The **Number Guessing Game** is a simple yet engaging Java-based application designed to introduce and reinforce core programming concepts through an interactive gaming experience. Its primary goal is to provide a fun and practical way for users to apply foundational coding skills while promoting logical thinking and problem-solving.

At its core, the game challenges players to guess a randomly generated number within a predefined range. Each guess prompts immediate feedback—indicating whether the number is too high, too low, or correct—guiding players toward the solution. The game tracks attempts and continues until the correct number is guessed, making it both a test of logic and persistence. The application is built with a modular architecture, comprising distinct components such as Random Number Generation, Input Handling, Feedback Mechanism, Game Logic, and Flow Control. Each module plays a specific role, ensuring a seamless and maintainable design. The game loop and user interaction are handled efficiently using Java constructs such as loops, conditionals, and the Scanner class for input.

By offering a user-friendly and interactive platform, the Number Guessing Game serves as an excellent learning tool for programming beginners. Its simplicity and scalability make it a versatile resource for educational purposes, fostering confidence and skills in coding through a practical and enjoyable approach.

1.3 Java Programming Concepts

The development of the **Number Guessing Game** incorporates several fundamental Java programming concepts, along with GUI elements using **Java Swing,** to create a functional, maintainable, and interactive application. These concepts include:

1. **Random Number Generation:**
   - The Random class or Math.random() is used to generate a random number within a predefined range, forming the foundation of the game.
2. **Control Structures:**
   - **Loops:** A while loop enables the game to continue until the correct number is guessed.
   - **Conditionals: if-**else statements evaluate the player's guess, providing feedback on whether it is too high, too low, or correct.
3. **User Input Handling:**
   - The Scanner class is used for console-based input.
   - For GUI-based input, Java Swing components like JTextField and JButton handle user interactions.
4. **Java Swing for GUI Development:**
   - **Components:** The graphical interface is built using Swing components such as JFrame, JPanel, JLabel, JButton, and JTextField.
   - **Event Handling:** Player interactions, like clicking a "Submit" button or entering a guess, are managed using event listeners such as ActionListener to provide dynamic feedback.
   - **Feedback Display:** Swing elements like JLabel or JOptionPane display messages to guide the user based on their input.
5. **Variables and Data Types:**
   - Variables store essential game elements, such as the random number, player guesses, and the number of attempts, ensuring proper data management throughout the game.
6. **Modularity:**
   - The program is divided into logical modules, such as number generation, input handling, feedback generation, and GUI rendering, making it easier to maintain and extend.
7. **Exception Handling:**
   - try-catch blocks manage potential errors, such as invalid user inputs or program

interruptions, ensuring the game runs smoothly.

8. **Object-Oriented Programming (OOP):**
   - Classes and objects are utilized to encapsulate data and behavior, such as game logic and GUI functionality, promoting reusability and scalability.
9. **Code Reusability:**
   - Methods are implemented to encapsulate repetitive logic, such as validating inputand updating feedback, ensuring cleaner and more modular code.
10. **Integration of Console and GUI:**
    - While the game may initially use console input for simplicity, the addition of Java Swing components enhances interactivity and provides a polished, user-friendly experience.

By combining fundamental programming concepts with GUI development in Java Swing, the Number Guessing Game becomes an engaging project that demonstrates the versatility of Java. It provides an excellent opportunity to practice both functional programming and user interface design.

CHAPTER 2

PROJECT METHODOLOGY

2.1 Proposed Work

The **proposed work** for the **Number Guessing Game** project involves the development of an interactive and engaging application that allows users to guess a randomly generated number, with real-time feedback guiding them toward the correct answer. The main goal is to design a user-friendly game that not only entertains but also serves as a platform for learning fundamental **Java programming** concepts. Below is a detailed breakdown of the proposed work:

1. **Game Setup:**
   - A predefined range (e.g., 1 to 100) will be established for the random number generation.
   - A method will be implemented to generate the target number using the Random class or Math.random().
2. **User Interaction:**
   - Players will interact with the game through a console interface initially, with a later transition to a graphical user interface (GUI) using Java Swing.
   - The game will prompt users to input their guesses and validate the input to ensure it is numeric and within the specified range.

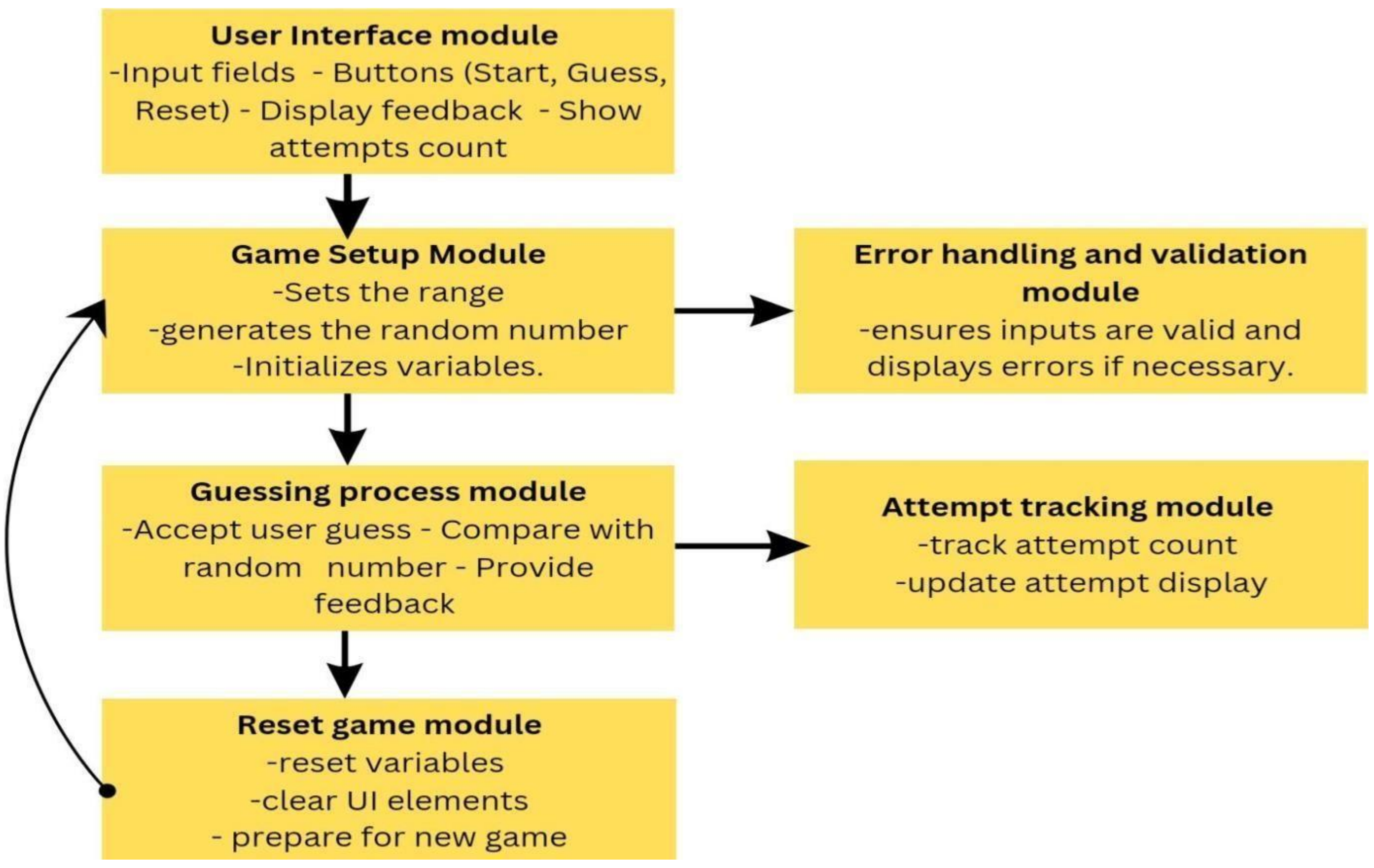
3. **Feedback Mechanism:**
   - After each guess, the game will provide feedback, such as "too high," "too low," or "correct," to guide the user.
   - Feedback will be displayed via console output or GUI elements like JLabel or JOptionPane.
4. **Game Loop:**
   - A loop will be implemented to allow continuous gameplay until the player guesses the correct number.
   - The game will track the number of attempts and display this information at the end.
5. **Graphical User Interface (GUI):**
   - The GUI will be developed using **Java Swing**, incorporating components like JFrame, JLabel, JTextField, and JButton to enhance interactivity.
   - Users will enter their guesses through a JTextField, and feedback will be displayed dynamically on the interface.
6. **Error Handling**:
   - Robust error handling will be implemented to manage invalid inputs, ensuring the

game remains stable and user-friendly.
- For example, non-numeric inputs will trigger error messages prompting the user to try again.

7. **Code Modularity**:
   - The project will be structured into distinct modules, including random number generation, input handling, feedback generation, and game logic.
   - Modular design will ensure maintainability and ease of debugging.
8. **Testing and Debugging**:
   - Thorough testing will be conducted to ensure the correctness of random number generation, input validation, feedback mechanism, and GUI components.
   - Edge cases, such as repeated guesses or out-of-range inputs, will also be tested.
9. **Deployment:**
   - The final product will be compiled into an executable JAR file for easy use.
   - Documentation will be provided, including a user guide detailing how to play the game and interact with its features.

This structured approach ensures the development of a robust, interactive, and educational Number Guessing Game that caters to both entertainment and learning needs.

2.2 Block Diagram


User Interface module
-Input fields - Buttons (Start, Guess, Reset) - Display feedback - Show attempts count
Game Setup Module
-Sets the range
-generates the random number
-Initializes variables.
Error handling and validation module
-ensures inputs are valid and displays errors if necessary.
Guessing process module
-Accept user guess - Compare with random number - Provide feedback
Attempt tracking module
-track attempt count
-update attempt display
Reset game module
-reset variables
-clear UI elements
- prepare for new game

CHAPTER 3

MODULE DESCRIPTION

3.1Game Setup Module

Objective: To set up the game environment by initializing the game range and preparing foruser interaction.

Description:

The user provides the range (start and end) in the start RangeField and end Range from to Field The module validates the range to ensure that the start is less than the end of the It gen erate a random target number within the user-defined ranget using Irrandom nextInt().Sets to To the fun initial number of attempts to 0 and updates the attemptsLabel.oEnables or disables components (guess Field and guess Button) based on the from to game state. Provides feedbackthrough the messageLabel such as “Game started! Guess the number”.

Objective:

To handle the player’s to the function guesses and provide real-time feedback based on the comparison with the target number.

Description:

Input: The user enters their guess in the guessField and clicks the guessButton to the form I submit the guess.FunctionalityAfter the guess is entered, it is compared with the target number .If the guess is too low, too high, or correct, feedback is was provided via the messageLabel.The number of guesses is incremented after the ro eeach t attempt and displayed in attemptsLabel.If correct guess is made,the game ends by the disabling the guess input and button.Feedback from feedback (“Too high”, “Too low” “Correct”) is was the best displayed dynamically based on the guess comparison.

3.1 Attempt Tracking Module

Input The number of attempts increases with every new guess.FunctionalityTracks them to ber of guesses made by the player and updates the attempts counter (attemptsLabel) after the for each guess.the guess is entered, it is compared with the target number.If the guess is too low, too high,in the to it or correct, feedback is provided via the messageLabel.The number of guesses isincremented after each attempt and todisplayed in attempts Label.If the correct guess is made

The game ends by disabling the guess input and button.Feedback:The feedback ("Too high","Too low to", "Correct") is displayed dynamically based on the guess comparison.Reset Game Modul**e**

Objective: To reset the game to the funct its initial state, allowing the player to start a new round.

Error Handling and Validation Module

Objective: To handle invalid user inputs and ensure to the form game runs without interruptions.

Description:

Validates that the input range consists of integers and that the start range is less fu than the end range.If a user inputs Invalid data in (non--numeric values or out-of-range guesses), an error message is displayed in messageLabel.

3.2 User Interface (UI) Module

Objective: To the function of provide a graphical user interface for the player to interact with the game.

Description:

Input: The user interacts with the interface to set the range, make guesses, and view feedback.FunctionalityThe main window consists of input fields (startRangeField, endRangeField, guessField), buttons (startButton, guessButton, resetButton), and labels (messageLabel, attemptsLabel).Users can enter the range, start the game, submit guesses, and reset the game.Feedback is displayed dynamically based on the player's actions, and components are enabled or disabled based on the game state.

CHAPTER 4

CONCLUSION & FUTURE SCOPE

4.1 CONCLUSION

The Number Guessing Game successfully serves as an interactive platform for learning fundamental programming concepts in a fun and engaging way. By incorporating core Java principles such as random number generation, event handling, and GUI development using Java Swing, the project provides a seamless user experience while reinforcing essential coding skills. The modular architecture of the game ensures maintainability and scalability, with key features such as range validation, real-time feedback, attempt tracking, and game resetting. This project not only demonstrates practical programming techniques but also offers opportunities for future improvements, such as adding difficulty levels, high-score tracking, or multiplayer functionality. Overall, the Number Guessing Game meets its educational objectives and showcases the potential of interactive software to enhance learning and development skills.

4.2 FUTURE SCOPE

1. Accessibility: Include voice feedback or keyboard-only controls for better
2. Better Graphics: Improve the look with animations, themes, and custom designs.
3. Data Saving: Save player progress, scores, and settings for future sessions.
4. Educational Use: Create a tutorial or learning mode to teach programming concepts

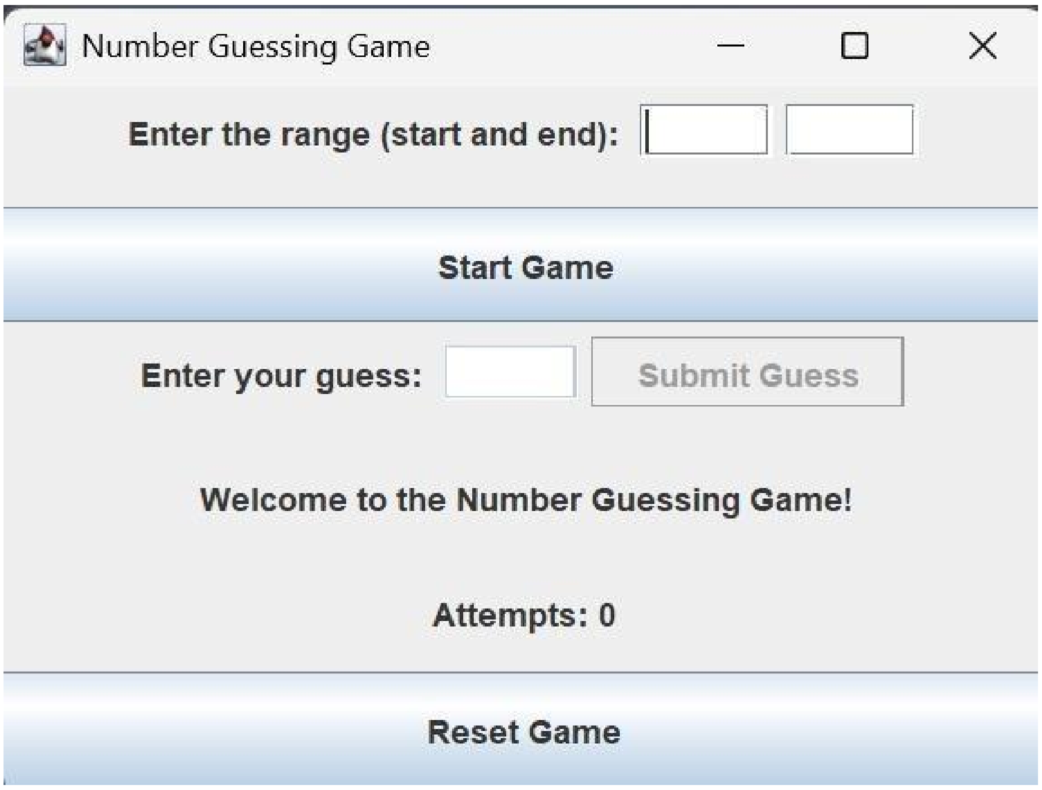
APPENDIX -A (Source Code)

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.Random;

public class Main extends JFrame {
    private JTextField startRangeField, endRangeField, guessField;
    private JLabel messageLabel, rangeLabel, guessPromptLabel, attemptsLabel;
    private JButton startButton, guessButton, resetButton;
    private int targetNumber, guesses;
    private Random random;

    public Main() {
        setTitle("Number Guessing Game");
        setSize(400, 300);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new BorderLayout());
        random = new Random();

        JPanel mainPanel = new JPanel();
        mainPanel.setLayout(new GridLayout(6, 1));

        JPanel rangePanel = new JPanel();
        rangeLabel = new JLabel("Enter the range (start and end): ");
        startRangeField = new JTextField(5);
        endRangeField = new JTextField(5);
        rangePanel.add(rangeLabel);
        rangePanel.add(startRangeField);
        rangePanel.add(endRangeField);

        startButton = new JButton("Start Game");
        startButton.addActionListener(new StartGameListener());

        JPanel guessPanel = new JPanel();
        guessPromptLabel = new JLabel("Enter your guess: ");
        guessField = new JTextField(5);
        guessButton = new JButton("Submit Guess");
        guessButton.addActionListener(new GuessListener());
        guessPanel.add(guessPromptLabel);
```

```
   guessPanel.add(guessField);
   guessPanel.add(guessButton);

      messageLabel = new JLabel("Welcome to the Number Guessing
Game!",JLabel.CENTER);

   attemptsLabel = new JLabel("Attempts: 0", JLabel.CENTER);

   resetButton = new JButton("Reset Game");
   resetButton.addActionListener(new ResetGameListener());

   mainPanel.add(rangePanel);
   mainPanel.add(startButton);
   mainPanel.add(guessPanel);
   mainPanel.add(messageLabel);
   mainPanel.add(attemptsLabel);
   mainPanel.add(resetButton);

   add(mainPanel, BorderLayout.CENTER);

   resetGame();
}

private void resetGame()
   { startRangeField.setText("");
   endRangeField.setText("");
   guessField.setText("");
   messageLabel.setText("Welcome to the Number Guessing Game!");
   attemptsLabel.setText("Attempts: 0");
   targetNumber = 0;
   guesses = 0;
   guessField.setEnabled(false);
   guessButton.setEnabled(false);
}

private class StartGameListener implements ActionListener
   {@Override
   public void actionPerformed(ActionEvent e)
      {try {
         int startRange = Integer.parseInt(startRangeField.getText().trim());
         int endRange = Integer.parseInt(endRangeField.getText().trim());

         if (startRange >= endRange) {
            messageLabel.setText("Invalid range! Start should be less than end.");
            } else {
                targetNumber = random.nextInt(endRange - startRange + 1) + startRange;
                guesses = 0;
                messageLabel.setText("Game started! Guess the number.");
                attemptsLabel.setText("Attempts: 0");
                guessField.setEnabled(true);
                guessButton.setEnabled(true);
            }
        } catch (NumberFormatException ex)
            { messageLabel.setText("Invalid input! Please enter valid
            integers.");
        }
    }
  }
  private class GuessListener implements ActionListener
    {@Override
    public void actionPerformed(ActionEvent e)
        {try {
            int guess = Integer.parseInt(guessField.getText().trim());
            guesses++;
            if (guess < targetNumber)
                { messageLabel.setText("Too low! Try
                again.");
            } else if (guess > targetNumber)
                { messageLabel.setText("Too high! Try
                again.");
            } else {
                messageLabel.setText("Congratulations! You guessed it in " + guesses + "
attempts.");
                guessField.setEnabled(false);
                guessButton.setEnabled(false);
            }
            attemptsLabel.setText("Attempts: " + guesses);
        } catch (NumberFormatException ex) {
            messageLabel.setText("Invalid input! Please enter a valid number.");
        }
    }
  }
  private class ResetGameListener implements ActionListener
    {@Override
    public void actionPerformed(ActionEvent e)
        {resetGame();
    }
  }
```

```
    public static void main(String[] args)
        {SwingUtilities.invokeLater(() -> {
            Main game = new Main();
            game.setVisible(true);
        });
    }
}
```

APPENDIX -B (Screenshots)


Number Guessing Game

Enter the range (start and end):

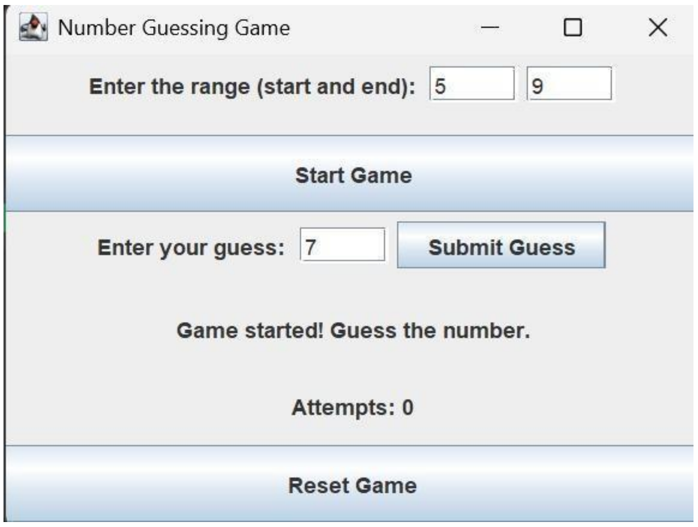
Start Game

Enter your guess: Submit Guess

Welcome to the Number Guessing Game!

Attempts: 0

Reset Game


Number Guessing Game

Enter the range (start and end): 5 9

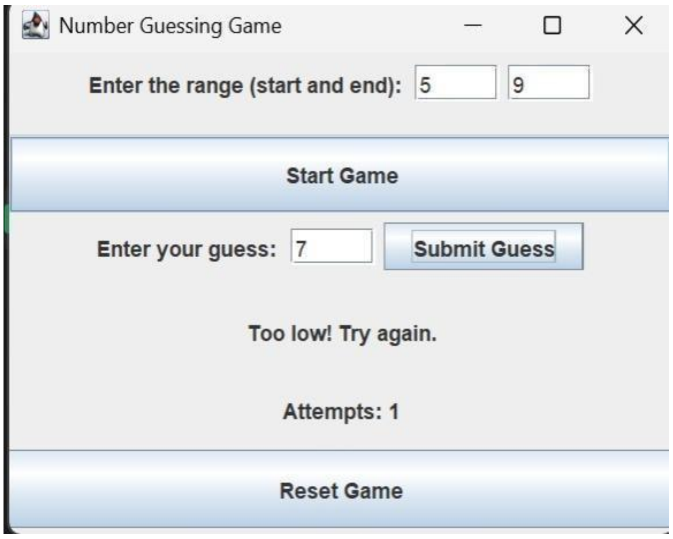
Start Game

Enter your guess: 7 Submit Guess

Game started! Guess the number.

Attempts: 0

Reset Game


Number Guessing Game

Enter the range (start and end): 5 9

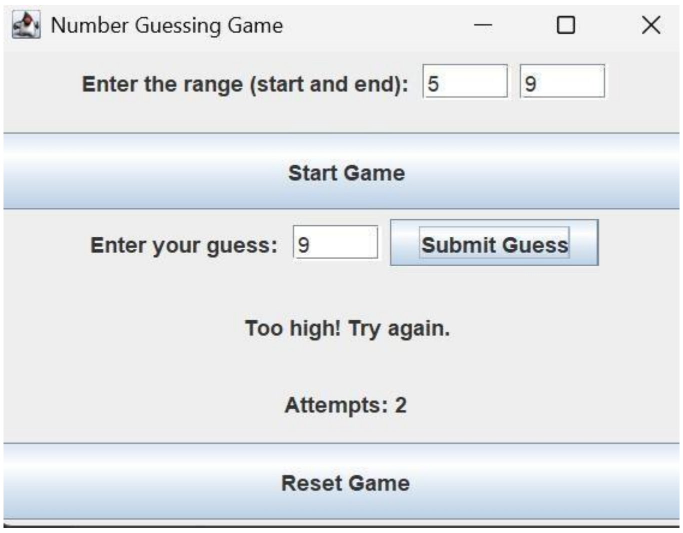
Start Game

Enter your guess: 7 Submit Guess

Too low! Try again.

Attempts: 1

Reset Game


Number Guessing Game

Enter the range (start and end): 5 9

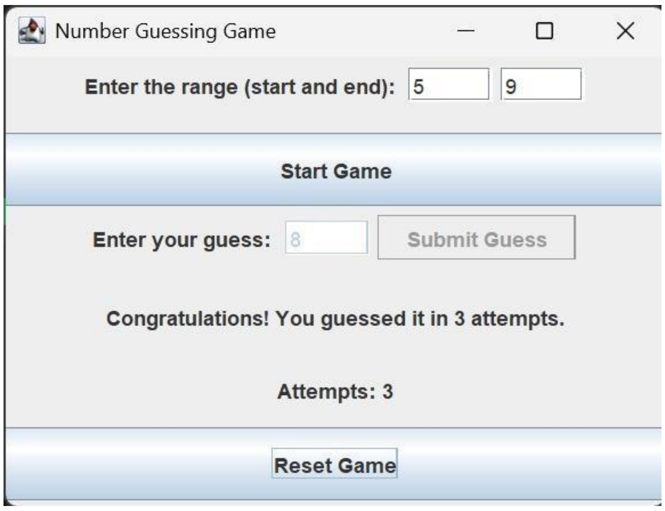
Start Game

Enter your guess: 9 Submit Guess

Too high! Try again.

Attempts: 2

Reset Game


Number Guessing Game

Enter the range (start and end): 5 9

Start Game

Enter your guess: 8 Submit Guess

Congratulations! You guessed it in 3 attempts.

Attempts: 3

Reset Game

REFERENCES

1. Deitel, H. M., & Deitel, P. J. (2019). Java: How to Program (11th Edition). Pearson Education.
2. Geeks forGeeks. (2024). Java Swing Tutorial. Retrieved fromhttps://www.geeksforgeeks.org/java-swing/
3. Oracle. (2024). Java Random Class. Oracle Documentation. Retrieved from https://docs.oracle.com/javase/8/docs/api/java/util/Random.html
4. Stack Overflow. (2024). Java Swing Example: Simple Number Guessing Game. Retrieved from https://stackoverflow.com/questions/