GUESS THE NUMBER GAME

A PROJECT REPORT

Submitted by

TEJES SHREE J (2303811710422167)

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 **“GUESS THE NUMBER GAME”** is the bonafide work of **TEJES SHREE J (2303811710422167)** 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

CGB1201-JAVA PROGRAMMING
Mr.MALARMANNAN A, M.E.,
SUPERVISOR
ASSISTANTP ROFESSOR

SIGNATURE	**SIGNATURE**
Mrs.A.Delphin Carolina Rani, M.E.,Ph.D.,	Mr. A. Malarmannan, 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
ASSISTANTPROFESSOR

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

INTERNAL EXAMINER

EXTERNAL EXAMINER

DECLARATION

I declare that the project report on **“GUESS THE NUMBER 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

TEJES SHREE J

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 "Guess the Number" game is a classic number-based guessing game that combines elements of probability, logic, and user interaction to provide an engaging and educational experience. In this game, the computer or a program generates a random number within a specified range, typically between 1 and 100, and the player is tasked with guessing the number within a limited number of attempts or until the correct number is found. Each guess triggers feedback from the system, indicating whether the player's input is too high, too low, or correct. This feedback loop enables the player to refine their guesses and apply logical reasoning to converge on the correct answer.The game fosters essential cognitive skills, such as problem-solving, pattern recognition, and strategic thinking, as players must adjust their approach based on the information provided after each attempt. Additionally, variations of the game can introduce different levels of difficulty by altering the range of numbers, limiting the number of guesses, or incorporating a scoring system based on the number of attempts taken to reach the solution. The "Guess the Number" game serves as a fundamental programming exercise for beginners, teaching concepts such as loops, conditional statements, functions, and random number generation. It can be implemented in various programming languages, including Python, Java, C++, and JavaScript, and can be enhanced with graphical user interfaces (GUIs) for a more interactive user experience.

ABSTRACT WITH POs AND PSOs MAPPING

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

ABSTRACT	POs MAPPED	PSOs MAPPED
The "Guess the Number" game is a classic number-based guessing game that combines elements of probability, logic, and user interaction to provide an engaging and educational experience. In this game, the computer or a program generates a random number within a specified range, typically between 1 and 100, and the player is tasked with guessing the number within a limited number of attempts or until the correct number is found.	**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

1.1 Objective

The objective of the "Guess the Number" game is to engage players in a fun and interactive challenge where they must correctly guess a randomly generated number within a specified range. The game provides immediate feedback on each guess, indicating whether the guessed number is "Too High" or "Too Low," guiding players toward the correct answer. It also incorporates different difficulty levels—Easy, Medium, and Hard—to cater to players of varying skill levels, with each level adjusting the range of numbers and the number of allowed attempts. The goal is to guess the correct number using the fewest attempts possible, adding an element of strategy and critical thinking. Additionally, the game fosters replayability by offering players the option to play again or exit, making it an enjoyable and repeatable experience that tests their logical deduction and patience.

1.2 Overview

The **"Guess the Number"** game is a simple yet engaging Java-based application designed to test a player's ability to logically deduce a randomly generated number within a specified range. The game offers three difficulty levels—Easy, Medium, and Hard—each varying the range of possible numbers and the number of allowed attempts, making it suitable for both casual players and those seeking a greater challenge. Players receive real-time feedback on each guess, helping them adjust their subsequent guesses until they find the correct number. The game tracks the number of attempts and displays the result once the correct number is guessed or the attempts are exhausted. With a user-friendly interface, customizable gameplay, and an option to replay or exit after each round, the game provides an entertaining and interactive experience that enhances logical thinking and decision-making skills.

1.3 Java Programming Concepts

1. Object oriented programming:

- **Encapsulation:**

The game’s logic can be encapsulated within classes, such as a Game class that manages the game's state, a Player class for user data, and a Difficulty class to define ranges and limits. This ensures that data (e.g., random number, attempts) and methods (e.g., guess validation, feedback generation) are bundled together, promoting modularity and data hiding.

- **Abstraction:**

Abstraction simplifies complex operations like random number generation, user input handling, and feedback processing. For instance, a method such as checkGuess() might abstract the comparison logic, while a startGame() method can manage the overall game flow without exposing implementation details to the user.

- **Inheritance:**

Difficulty levels (Easy, Medium, Hard) can be implemented as subclasses inheriting from a base Difficulty class. The base class might define common properties (e.g., number range, attempt count), while subclasses override or extend these properties for specific difficulty levels.

- **Polymorphism:**

Polymorphism allows methods like getRange() or getAttempts() to behave differently based on the difficulty level. For instance, calling these methods on an Easy, Medium, or Hard object could return different values based on the specific implementation in each subclass.

- **Class and Object Design:**

The game could use objects to represent the player, the game session, or difficulty configurations, making the code modular and reusable. Each object would interact with others to create the full game experience.

2. Random Number Generation:

The use of Java's Random class (or similar) to generate a random number within a defined range demonstrates efficient use of built-in utilities.

3. Control structure:

Conditional statements (if-else) for feedback generation and loops for multiple attempts are fundamental concepts integrated into the game.

4. Input/Output Handling:

The game incorporates user interaction through inputs (guesses) and outputs (feedback and results).

5. Game Flow Management:

Methods to track attempts, determine win/loss, and manage replay options demonstrate the structured organization of the game's flow.

6. Exception Handling

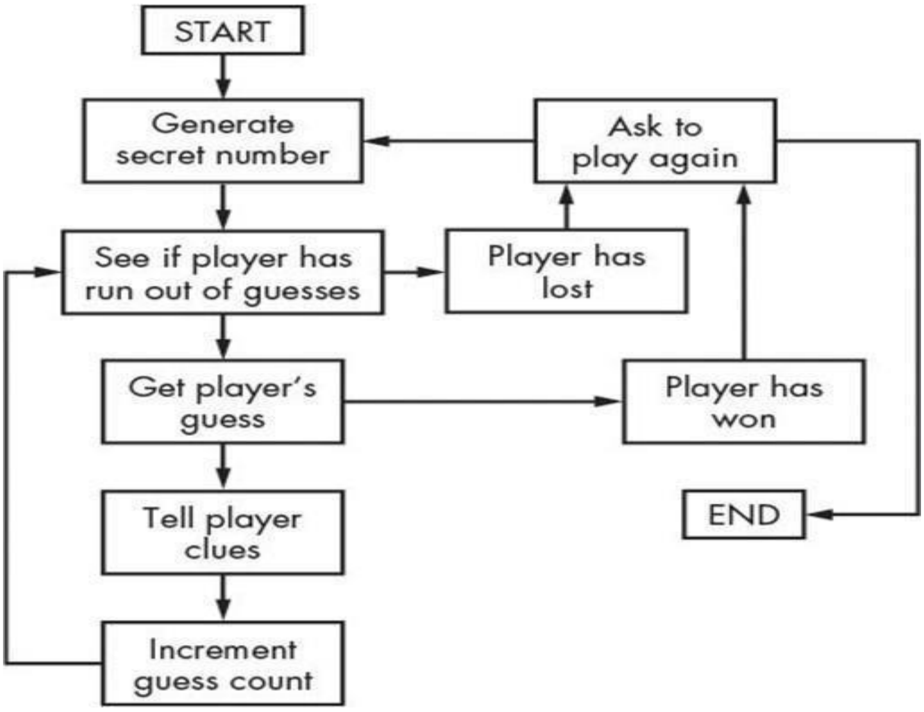
To improve the user experience, exception handling is implemented to manage invalid inputs, such as non-integer values. A try-catch block can be used to catch InputMismatchException when the player enters invalid data, prompting the user to enter a valid number. This prevents the game from crashing due to incorrect input and ensures a smoother gameplay experience.

CHAPTER 2
PROJECT METHODOLOGY

2.1 Proposed Work

The proposed work involves the development of a Java-based interactive game, *"Guess the Number,"* with an emphasis on functionality, user engagement, and scalability. The project will begin with designing the core game mechanics, including random number generation within a specific range and providing real-time feedback ("Too High," "Too Low," or "Correct!") for each guess. The system will include a mechanism to track the number of attempts and determine win/loss conditions. To cater to diverse users, three difficulty levels—Easy, Medium, and Hard—will be implemented, each varying the number range and allowable attempts. A modular approach will be adopted, leveraging Object-Oriented Programming (OOP) principles to ensure maintainability and future extensibility.

2.2 Block Diagram



CHAPTER 3
MODULE DESCRIPTION

3.1 Game Initialization Module

Description: Handles the setup of the game, including generating the random number, initializing variables like the number of attempts, and setting the difficulty level based on user selection.

3.2 Input Handling Module

Description: Manages player interactions by accepting guesses and ensuring input is valid (e.g., within the specified range and numeric).

3.3 Feedback and Evaluation Module

Description: Provides feedback for each guess and evaluates game progress.

3.4 Difficulty Management Module

Description: Manages the game settings for different difficulty levels

3.5 Replay and Exit Module

Description: Provides options for players to replay the game or exit after a round.

CHAPTER 4
CONCLUSION & FUTURE SCOPE

4.1 CONCLUSION

The "Guess the Number" game is a fun and interactive activity that tests a player's ability to think critically and strategically. The game begins with one player, or a computer program, selecting a random number within a predetermined range, such as 1 to 100. The player's goal is to guess the hidden number by making educated guesses. After each attempt, feedback is provided, indicating whether the guess is too high, too low, or correct. This feedback helps players refine their guesses and narrow down the possibilities with each turn. The game not only offers an exciting challenge but also encourages players to use logical reasoning, pattern recognition, and even probability to make better guesses.

4.2 FUTURE SCOPE

The "Guess the Number" game is a great interactive experience, and there are several ways to enhance and extend its functionality in the future. One possible improvement is to **introduce more advanced difficulty settings**, such as adding **variable ranges for each difficulty level** (e.g., Easy: 1-50, Medium: 1-100, Hard: 1-1000) and **adjusting the number of attempts** based on the difficulty. Additionally, **saving the player's best scores** (least number of attempts to guess correctly) can add a competitive element. You could also implement **multiplayer modes**, allowing players to take turns guessing numbers against each other or a system that compares their scores. Another interesting feature could be **leaderboards** to track the best scores across different difficulty levels.

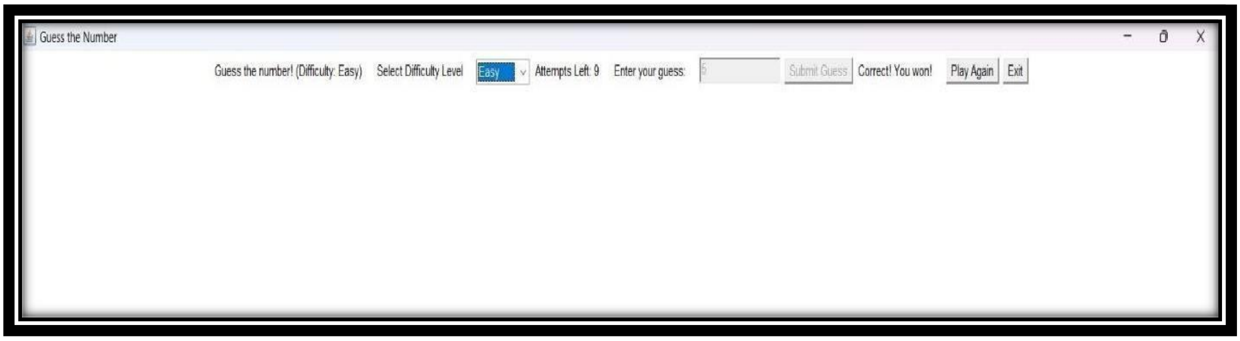
APPENDIX A

(SOURCE CODE)

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

public class GuessTheNumberGame extends Frame {
    // Declare components
    private Label instructionLabel, feedbackLabel, attemptsLabel,
difficultyLabel;
    private TextField guessField;
    private Button submitButton, playAgainButton, exitButton;
    private Choice difficultyChoice;

    private int targetNumber, attempts, maxAttempts, rangeMin, rangeMax;
    private Random rand;

    public GuessTheNumberGame() {
        // Set the window properties
        setTitle("Guess the Number");
        setSize(400, 350);
        setLayout(new FlowLayout());

        // Initialize components
        instructionLabel = new Label("Welcome to Guess the Number Game!");
        feedbackLabel = new Label("");
        attemptsLabel = new Label("Attempts Left: ");
        difficultyLabel = new Label("Select Difficulty Level");
        guessField = new TextField(10);
        submitButton = new Button("Submit Guess");
        playAgainButton = new Button("Play Again");
        exitButton = new Button("Exit");

        difficultyChoice = new Choice();
        difficultyChoice.add("Easy");
        difficultyChoice.add("Medium");
        difficultyChoice.add("Hard");

        // Add components to the frame
        add(instructionLabel);
        add(difficultyLabel);
        add(difficultyChoice);
        add(attemptsLabel);
        add(new Label("Enter your guess:"));
        add(guessField);
        add(submitButton);
        add(feedbackLabel);
        add(playAgainButton);
        add(exitButton);

        // Set default visibility of buttons
        playAgainButton.setVisible(false);
        exitButton.setVisible(false);

        // Set up event listeners
        submitButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                handleGuess();
            }
        });

        playAgainButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                resetGame();
            }
        });

        exitButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                System.exit(0);
            }
        });

        // Start a new game
        resetGame();

        // Make the window visible
        setVisible(true);
    }

    private void resetGame() {
        // Initialize game variables based on difficulty
        String difficulty = difficultyChoice.getSelectedItem();
        if (difficulty.equals("Easy")) {
            rangeMin = 1;
            rangeMax = 50;
            maxAttempts = 10;
        } else if (difficulty.equals("Medium")) {
            rangeMin = 1;
            rangeMax = 100;
            maxAttempts = 7;
        } else if (difficulty.equals("Hard")) {
            rangeMin = 1;
            rangeMax = 200;
            maxAttempts = 5;
        }

        // Create a new random number to guess
        rand = new Random();
        targetNumber = rand.nextInt(rangeMax - rangeMin + 1) + rangeMin;

        // Reset attempts and update UI
        attempts = 0;
        feedbackLabel.setText("");
        attemptsLabel.setText("Attempts Left: " + maxAttempts);
        guessField.setText("");

        // Show number of attempts and difficulty level
        instructionLabel.setText("Guess the number! (Difficulty: " + difficulty +
")");


        // Reset buttons
        submitButton.setEnabled(true);
        guessField.setEnabled(true);
        playAgainButton.setVisible(false);
        exitButton.setVisible(false);
    }

    private void handleGuess() {
        try {
            // Get the player's guess
            int playerGuess = Integer.parseInt(guessField.getText());

            // Increase attempts count
            attempts++;
            int remainingAttempts = maxAttempts - attempts;
            attemptsLabel.setText("Attempts Left: " + remainingAttempts);

            // Provide feedback on the guess
            if (playerGuess < targetNumber) {
                feedbackLabel.setText("Too Low! Try again.");
            } else if (playerGuess > targetNumber) {
                feedbackLabel.setText("Too High! Try again.");
            } else {
                feedbackLabel.setText("Correct! You won!");
                endGame();
                return;
            }

            // If the player has exceeded the maximum number of attempts
            if (remainingAttempts == 0) {
                feedbackLabel.setText("Game Over! The correct number was " +
targetNumber);
                endGame();
             }



        } catch (NumberFormatException e) {
            feedbackLabel.setText("Please enter a valid number!");
        }
    }

    private void endGame() {
        // Disable input and show the Play Again and Exit buttons
        submitButton.setEnabled(false);
        guessField.setEnabled(false);
        playAgainButton.setVisible(true);
        exitButton.setVisible(true);
    }

    public static void main(String[] args) {
        new GuessTheNumberGame();
    }
}
```

APPENDIX B
(SCREENSHOTS)

DIFFICULTY LEVEL: EASY



Guess the Number

Guess the number! (Difficulty: Easy)

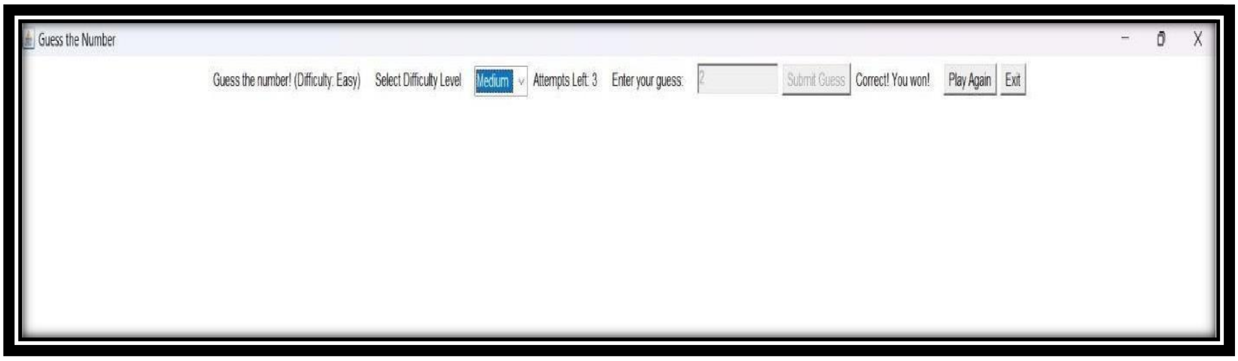
Select Difficulty Level: Easy

Attempts Left: 9

Enter your guess: 5

Correct! You won!

DIFFICULTY LEVEL: MEDIUM



Guess the Number

Guess the number! (Difficulty: Easy)

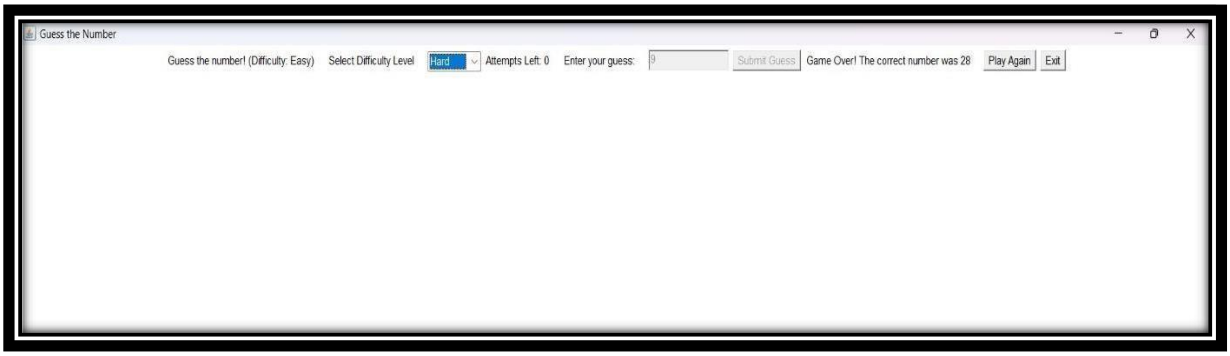
Select Difficulty Level: Medium

Attempts Left: 3

Enter your guess: 2

Correct! You won!

DIFFICULTY LEVEL: HARD



Guess the Number

Guess the number! (Difficulty: Easy)

Select Difficulty Level: Hard

Attempts Left: 0

Enter your guess: 9

Game Over! The correct number was 28

REFERENCES

1."Brown, J. (2020) 'Building a Number Guessing Game in Java', Journal of Software Development, Vol.15, No.3, pp.123-127."

2."Clark, A. and Smith, D. (2019) 'Game Development with Java: Guess the Number Example', Proceedings of the International Java Programming Conference, London, UK, pp.89-95."

3.Java SE 17 Documentation: Java.util.Random Class

4.GeekesforGeeks: https://www.geeksforgeeks.org/number-guessing-game-in-java/

5. Java2Blog: https://java2blog.com/number-guessing-game-java/