NUMBER GUESSING GAME

A Project Report

Submitted By

Akshala.T (8115U23AM006)

in partial fulfillment of requirements for the award of the course

CGB1201 – JAVA PROGRAMMING

In

COMPUTER SCIENCE AND ENGINEERING

(ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING)

K. RAMAKRISHNAN COLLEGE OF ENGINEERING

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

SAMAYAPURAM - 621 112

DECEMBER - 2024

K. RAMAKRISHNAN COLLEGE OF ENGINEERING
(Autonomous)

SAMAYAPURAM-621 112

BONAFIDE CERTIFICATE

Certified that this project report on **" NUMBER GUESSING GAME"** is the bonafide work of **AKSHALA.T (8115U23AM006)** who carried out the project work during the academic year 2024 - 2025 under my supervision.

SIGNATURE	**SIGNATURE**
Dr.B.KIRAN BALA	Mr. P.GEETHA
HEAD OF THE DEPARTMENT,	**SUPERVISOR**
ASSOCIATED PROFESSOR	ASSISTANT PROFESSOR,
Department of Artificial Intelligence and Machine Learning,	Department of Artificial Intelligence and Data science,
K. Ramakrishnan College of Engineering,	K. Ramakrishnan College of Engineering,
Samayapuram, Trichy-621 112.	Samayapuram, Trichy-621 112.

.

Submitted for the End Semester Examination held on ……………

INTERNAL EXAMINER	**EXTERNAL EXAMINER**

DECLARATION

I jointly 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 fulfillment of the requirement of the award of the course **CGB1201- JAVA PROGRAMMING**

SIGNATURE

AKSHALA.T

Place : Samayapuram

Date :

ACKNOWLEDGEMENT

It is with great pride that I express our gratitude and in-debt to our institution **"K.Ramakrishnan College of Engineering (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. D. SRINIVASAN, B.E, M.E., Ph.D.,** Principal, who gave opportunity to frame the project the full satisfaction.

I whole heartily thanks to **Dr. B. KIRAN BALA, B.Tech., M.E., M.B.A., Ph.D., M.I.S.T.E., U.A.C.E.E., IAENG** ,Head of the department, **Artificial Intelligence and Machine Learning** for providing her encourage pursuing this project.

I express our deep expression and sincere gratitude to our project supervisor **Mr.P.GEETHA M.E,** Department of Artificial Intelligence and data Science, 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.

INSTITUTE VISION AND MISSION

VISION OF THE INSTITUTE:

To achieve a prominent position among the top technical institutions.

MISSION OF THE INSTIITUTE:

M1: To best own standard technical education parexcellence through state of the art infrastructure, competent faculty and high ethical standards.

M2: To nurture research and entrepreneurial skills among students in cutting edge technologies.

M3: To provide education for developing high-quality professionals to transform the society.

DEPARTMENT VISION AND MISSION

DEPARTMENT OF CSE(ARTIFICIAL INTELLIGENCE AND MACHINELEARNING)

Vision of the Department

To become a renowned hub for Artificial Intelligence and Machine Learning Technologies to produce highly talented globally recognizable technocrats to meet Industrial needs and societal expectations.

Mission of the Department

M1: To impart advanced education in Artificial Intelligence and Machine Learning, Built upon a foundation in Computer Science and Engineering.

M2: To foster Experiential learning equips students with engineering skills to Tackle real-world problems.

M3: To promote collaborative innovation in Artificial Intelligence, machine Learning, and related research and development with industries.

M4: To provide an enjoyable environment for pursuing excellence while upholding Strong personal and professional values and ethics.

Programme Educational Objectives (PEOs):

Graduates will be able to:

PEO1: Excel in technical abilities to build intelligent systems in the fields of Artificial Intelligence and Machine Learning in order to find new opportunities.

PEO2: Embrace new technology to solve real-world problems, whether alone or As a team, while prioritizing ethics and societal benefits.

PEO3: Accept lifelong learning to expand future opportunities in research and Product development.

Programme Specific Outcomes (PSOs):

PSO1: Ability to create and use Artificial Intelligence and Machine Learning Algorithms, including supervised and unsupervised learning, reinforcement Learning, and deep learning models.

PSO2: Ability to collect, pre-process, and analyze large datasets, including data Cleaning, feature engineering, and data visualization..

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 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 a simple interactive project designed to engage users in a fun and challenging activity while reinforcing basic programming concepts. The game revolves around the user attempting to guess a randomly generated number within a specified range. This project serves as an excellent example for demonstrating conditional logic, loops, user input handling, and random number generation in programming.

ABSTRACT WITH POs AND PSOs MAPPING

ABSTRACT	POs MAPPED	PSOs MAPPED
This project is a number guessing game where the user tries to guess the number generated by the computer. It helps learn basic programming concepts like loops and conditions.	**PO1,PO2,PO5 ,PO9**	**PSO1,PSO2, PSO3**

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

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Objective :

The objective of the Number Guessing Game is to create an interactive program that challenges users to guess a randomly generated number within a specified range. The program provides feedback on whether the guess is too high, too low, or correct, helping the player refine their attempts. The project serves as a learning tool for understanding fundamental programming concepts, problem-solving, and user interaction.

1.2 Overview :

The Number Guessing Game is a simple and interactive game where players attempt to guess a secret number within a specified range. It's an engaging way to test problem-solving and logical thinking skills. The game can be played solo or with others and is often used as an introductory project for learning programming concepts.

1. Setup:

- A secret number is randomly selected within a predefined range (e.g., 1 to 100).
- The range can vary depending on the difficulty level.

2. Gameplay:

- The player makes a guess.
- The game provides feedback:
- "Too high" if the guess is greater than the secret number.
- "Too low" if the guess is less than the secret number.
- The player adjusts their next guess based on the feedback.

3. Win Condition:

- The player wins by guessing the exact number.
- If the game has a limited number of attempts, the player loses if they fail to guess the number within the allowed tries.

4. Optional Features:

- Score Tracking: Record how many attempts were made.
- Difficulty Levels: Adjust the range or limit the number of attempts.
- Multiplayer Mode: Compete to see who guesses correctly first.

1.3 Java Programming Concepts:

1. Random Number Generation:
Use of the Random class or Math.random() to generate unpredictable numbers.
Example:

```
Random random = new Random();
int number = random.nextInt(100) + 1; // Generates a number between 1 and 100.
```

2. Conditional Statements:

Use of if-else or switch statements to compare user guesses with the target number and provide feedback.

3. Loops:
Implementation of while or do-while loops to allow repeated attempts until the correct number is guessed or the maximum attempts are exceeded.

4. Input Handling:
Using the Scanner class to capture user input from the console.
Example:

```
Scanner scanner = new Scanner(System.in);
System.out.print("Enter your guess: ");
int guess = scanner.nextInt();
```

5. Error Handling:
Validating user input to prevent invalid entries (e.g., non-numeric inputs or out-of-range guesses).

6. Functions/Methods:
Encapsulation of logic into reusable methods, such as one for generating numbers, another for validating guesses, and one for feedback.

7. Basic Class Structure:
Organizing code into a class with a main method and possibly additional methods for modularity and readability.

8. Optional Advanced Features:
Implementing features like score tracking (using variables), replay options, and difficulty levels to enhance functionality and user experience.

CHAPTER 2
PROJECT METHODOLOGY

2.1 Proposed Work :

Developing a Number Guessing Game is a straightforward project, but applying a structured methodology ensures clarity, efficiency, and quality in the development process. Below is a proposed methodology tailored for this project.

1. Requirement Analysis (Planning Phase)

Goal: Define the scope and objectives of the game.

Key Questions:
What features should the game have? (e.g., difficulty levels, score tracking, or a GUI version)
What programming language and tools will be used? (e.g., Java, Eclipse/IntelliJ)
Who are the target users?

Deliverables:
Project plan outlining the scope, timeline, and resources.
List of functional requirements (e.g., random number generation, user input handling, hint system).

2. Design Phase

System Design:
Define the architecture (e.g., console-based or GUI-based).

Break the program into components:
Random number generator
Input validation
Feedback system ("Too high/Too low")
Win/lose conditions
Create a flowchart or UML diagram for the game's logic.

UI Design:
For console-based: Plan the input/output format.
For GUI-based: Sketch a basic layout of buttons and displays (if applicable).

3. Development Phase

Iterative Development:
Follow Agile Methodology for small, incremental progress:

Sprint 1: Implement basic gameplay logic (generate a random number and take user input).
Sprint 2: Add feedback (e.g., "Too high" or "Too low") and win/lose conditions.
Sprint 3: Introduce advanced features like difficulty levels or score tracking.
Sprint 4: Finalize UI (if applicable).
Use version control (e.g., Git) to track progress.

Best Practices:

Write modular code for reusability.

Test small components before integrating.

4. Testing Phase

Types of Testing:

Unit Testing: Test individual functions (e.g., random number generation).

Integration Testing: Ensure all components work together.

User Acceptance Testing (UAT): Gather feedback from a small group of users.

Test Cases:

Check if the game correctly identifies "Too high" or "Too low."

Verify input validation (e.g., non-numeric or out-of-range inputs).

Ensure the game ends when the user guesses the correct number or exceeds attempts.

5. Deployment Phase

Deploy the game:

Console-based: Package it as a runnable .jar file.

GUI-based: Provide an installer or web link.

Include instructions for users on how to play.

6. Maintenance Phase

Gather feedback from users to fix bugs or add features.

Update the codebase with improvements (e.g., enhanced UI, multiplayer mode).

Ensure compatibility with future Java versions.

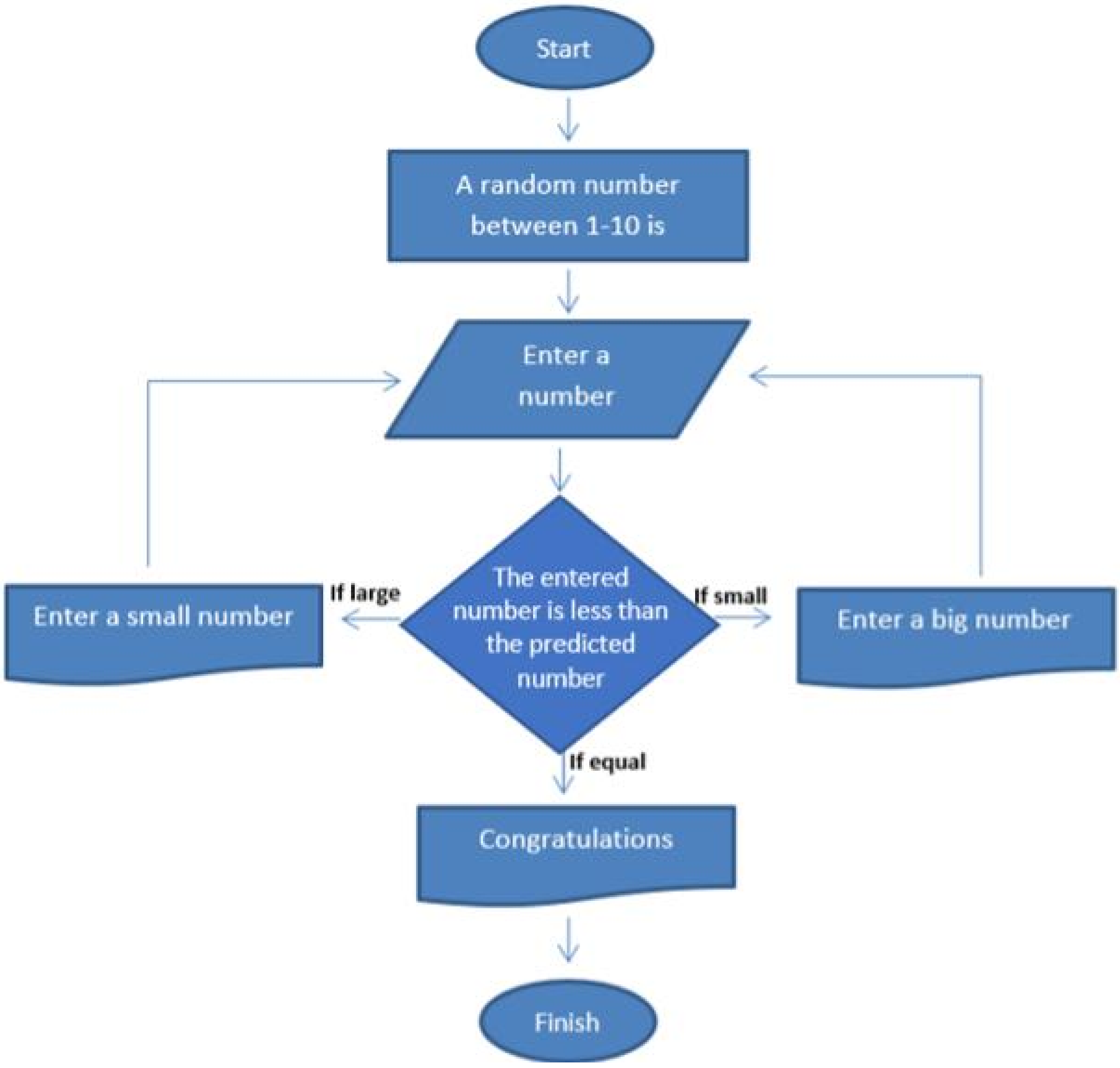
Why This Methodology Works

Iterative Approach: Breaks the project into manageable parts.

Testing-Driven: Ensures reliability at each stage.

Feedback-Driven: Allows for incremental improvements.

2.2 Block Diagram :



CHAPTER 3
MODULE DESCRIPTION

3.1. Random Number Generation

- Module Used: random
- Purpose: To generate a random number for the player to guess.
- Description: The random module provides functions to generate random numbers. In this game, you can use random.randint() to generate an integer within a specific range.

3.2. User Input Validation

- Module Used: None (uses built-in functions like input() and exception handling via try-except)
- Purpose: Ensures the player's input is valid (e.g., a number within the correct range).
- Description: Validate the user's input to prevent errors (e.g., entering non-numeric characters). You can handle exceptions using try-except or explicitly check the input type.

3.3. Feedback System

- Module Used: None (uses simple conditional statements)
- Purpose: Provide hints like "Too High" or "Too Low" to help the player.
- Description: Use if-else conditions to compare the player's guess to the secret number and give appropriate feedback.

3.4. Win/Loss Conditions

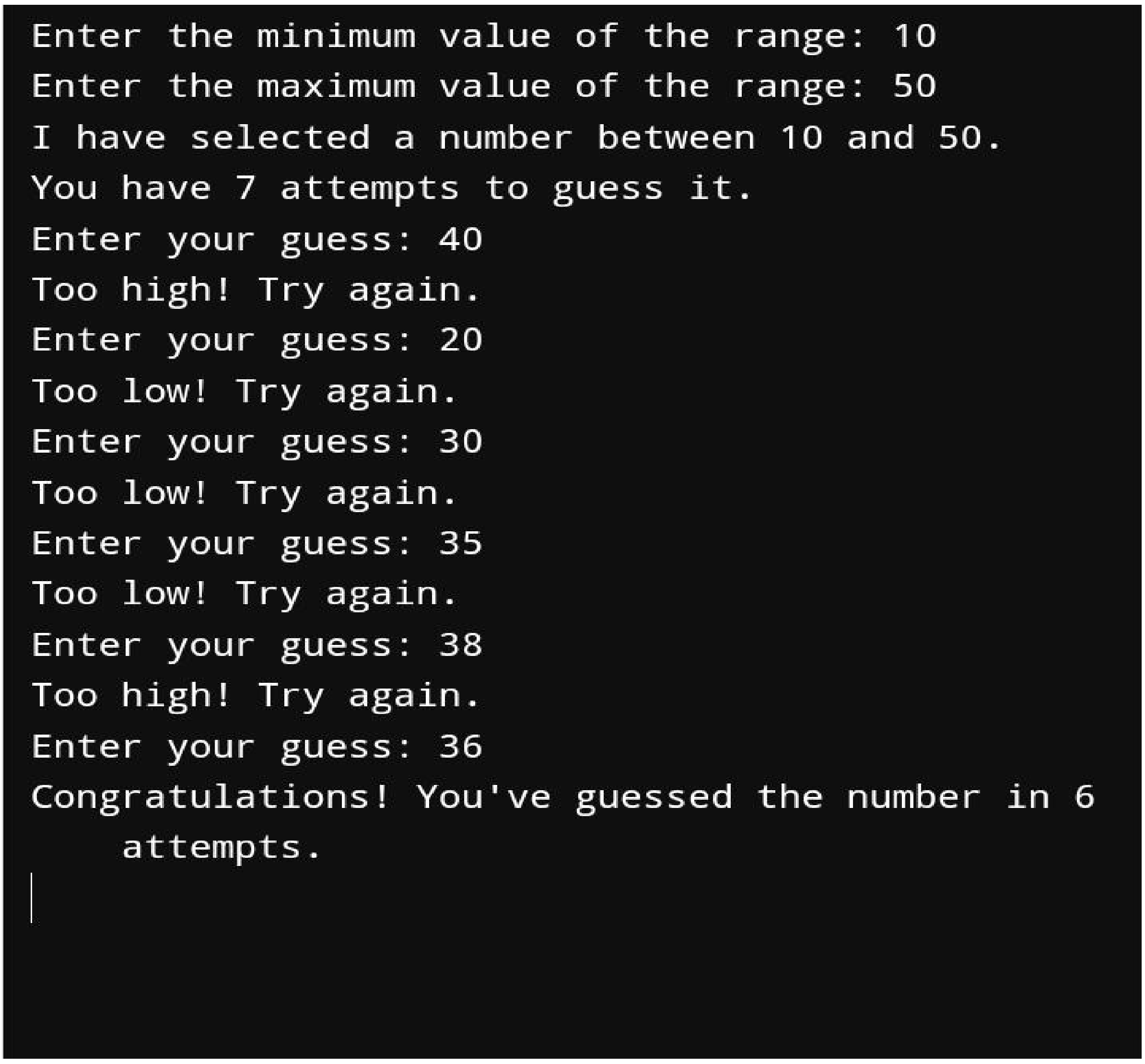
- Module Used: None (built-in logic)
- Purpose: Determine if the player wins by guessing correctly or loses by exceeding the allowed attempts.
- Description: Track the number of attempts and check whether the player guessed the number or ran out of guesses.

3.5. Replay Option

- Module Used: None (uses built-in input function for user interaction)
- Purpose: Allow the player to replay the game after a win or loss.
- Description: After the game ends, prompt the player to decide whether to replay or exit the game.

CHAPTER 4

RESULTS AND DISCUSSION



```
Enter the minimum value of the range: 10
Enter the maximum value of the range: 50
I have selected a number between 10 and 50.
You have 7 attempts to guess it.
Enter your guess: 40
Too high! Try again.
Enter your guess: 20
Too low! Try again.
Enter your guess: 30
Too low! Try again.
Enter your guess: 35
Too low! Try again.
Enter your guess: 38
Too high! Try again.
Enter your guess: 36
Congratulations! You've guessed the number in 6
    attempts.
```

DISCUSSION :

The first step in creating a number guessing game is to generate a random number for the player to guess. In Java, this can be done using the Random class. The nextInt() method of this class generates a random integer between 0 and the specified maximum value. For our game, we want the number to be between 1 and 100, so we will use the following code to generate our random number:

```
Random rand = new Random();
int numberToGuess = rand.nextInt(100) + 1;
```

Next, we need to prompt the player to enter their guess. We will use the Scanner class to read the player's input. The following code initializes the Scanner object and prompts the player to enter their guess:

```
Scanner scanner = new Scanner(System.in);
System.out.println("Guess a number between 1 and 100:");
int guess = scanner.nextInt();
```

Now that we have the player's guess, we can compare it to the randomly generated number. If the player's guess is too high, we will tell them to try again. If the player's guess is too low, we will tell them to try again. If the player's guess is correct, we will congratulate them and end the game.

Finally, we need to close the Scanner object to release any resources it is using. This can be done using the close() method.

DISCRIPTION ABOUT OUTPUT :

1. Setup of the Range: The user defines a minimum and maximum value for the range (e.g., 10 to 50 in this case).

2. Selection of a Random Number: The program selects a random number within the given range.

3. Number of Attempts: The user is allowed a certain number of attempts (7 attempts here) to guess the number.

4. Guess Feedback: After each guess, the program provides feedback:
"Too high!" if the guess is greater than the selected number.
"Too low!" if the guess is smaller than the selected number.

4. Win Condition: The game ends successfully when the user guesses the correct number.

6. Result Announcement: The program congratulates the user and displays the number of attempts taken.

In this particular session:
The user took 6 attempts to guess the correct number (36)

CHAPTER 5
CONCLUSION

The number guessing game project was a valuable exercise in programming and problem-solving. It provided an opportunity to apply fundamental concepts such as conditional statements, loops, random number generation, and user input handling. The game's simple design made it accessible, while its logic fostered an understanding of algorithm development and optimization. Through this project, we also enhanced our debugging and testing skills to ensure smooth gameplay. Overall, it serves as a great foundation for building more complex projects in the future.The Number Guessing Game not only provides a fun and interactive experience for users but also offers a solid foundation for learning and applying programming techniques. With further enhancements, such as adding difficulty levels, a graphical user interface (GUI), or multiplayer options, this project can be expanded into a more complex and engaging game. This project highlights the importance of planning, iterative development, and testing, which are essential skills in both beginner and professional software development.

APPENDIX

```
import java.util.Random;
import java.util.Scanner;

public class NumberGuessingGame { // Ensure class name matches file name

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        Random random = new Random();

        // Get the range from the user
        System.out.print("Enter the minimum value of the range: ");
        int minValue = scanner.nextInt();
        System.out.print("Enter the maximum value of the range: ");
        int maxValue = scanner.nextInt();

        // Validate the range
        if (minValue >= maxValue) {
            System.out.println("Invalid range. Minimum value must be less than
maximum value.");
            return;
        }

        // Generate a random number within the specified range
        int numberToGuess = random.nextInt(maxValue - minValue + 1) + minValue;
        int attempts = 0;
        final int MAX_ATTEMPTS = 7;
        boolean hasGuessedCorrectly = false;

        System.out.println("I have selected a number between " + minValue + " and " +
maxValue + ".");
        System.out.println("You have " + MAX_ATTEMPTS + " attempts to guess it.");

        while (attempts < MAX_ATTEMPTS) {
            System.out.print("Enter your guess: ");
            try {
                int userGuess = scanner.nextInt();
                attempts++;
if (userGuess < minValue || userGuess > maxValue) {
                    System.out.println("Your guess is out of the specified range. Please
guess a number between " + minValue + " and " + maxValue + ".");
                } else if (userGuess < numberToGuess) {
                    System.out.println("Too low! Try again.");
                } else if (userGuess > numberToGuess) {
                    System.out.println("Too high! Try again.");
                } else {
                    hasGuessedCorrectly = true;
                    System.out.println("Congratulations! You've guessed the number in " +
attempts + " attempts.");
                    break;
                }
            } catch (Exception e) {
                System.out.println("Invalid input. Please enter a valid integer.");
                scanner.next(); // Clear the invalid input
            }
        }

        if (!hasGuessedCorrectly) {
            System.out.println("Sorry, you've used all your attempts. The number was: "
+ numberToGuess);
        }

        // Close the scanner
        scanner.close();
    }
}
```

REFERENCE

1. R. Ahlswede, N. Cai, S.-Y. R. Li and R. W. Yeung, "Network information flow", IEEE Trans. Inf. Theory, vol. 46, no. 4, pp. 1204-1216, Apr. 2000.

2. R. Baber, D. Christofides, A. N. Dang, S. Riis and E. R. Vaughan, "Multiple unicasts graph guessing games and non-Shannon inequalities", Proc. Int. Symp. Netw. Coding (Netcod), pp. 1-6, Jun. 2013.

3.D. Christofides and K. Markström, "The guessing number of undirected graphs", Electron. J. Combinat., vol. 18, no. 1, pp. 192, 2011.

4.T. H. Chan, "Capacity regions for linear and abelian network codes", Proc. Inf. Theory Appl. Workshop, pp. 73-78, Jan./Feb. 2007.

5.T. Chan and A. Grant, "Network coding capacity regions via entropy functions", IEEE Trans. Inf. Theory, vol. 60, no. 9, pp. 5347-5374, Sep. 2014.

6.T. M. Cover and J. A. Thomas, Elements of Information Theory, Hoboken, NJ, USA:Wiley, 2006.

7.R. Dougherty, C. Freiling and K. Zeger, "Networks matroids and non-Shannon information inequalities", IEEE Trans. Inf. Theory, vol. 53, no. 6, pp. 1949-1969, Jun. 2007.

8.R. Dougherty, C. Freiling and K. Zeger, Non-Shannon information inequalities in four random variables, Apr. 2011, [online] Available: https://arxiv.org/abs/1104.3602.

9.C. Fragouli and E. Soljanin, "Network coding fundamentals", Found. Trends Netw., vol. 2, no. 1, pp. 1-133, 2007.

10.L. Guillé, T. Chan and A. Grant, "The minimal set of Ingleton inequalities", IEEE Trans. Inf. Theory, vol. 57, no. 4, pp. 1849-1864, Apr. 2011.