Tribhuwan University

Institute of Engineering

Thapathali Campus

Proposal On:

Collection of games

Submitted By:

Bishal Upadhyay [THA081BEI010]

Devendra Yadav [THA081BEI011]

Muhammad Shahid Raain [THA081BEI020]

Ajay Kumar Pasi [THA081BEI003]

Submitted to:

Department of Electronics and Computer Engineering

IOE, Thapathali Campus

Kathmandu, Nepal

Date: February 09

ACKNOWLEDGEMENT

We express our sincere gratitude to all individuals who contributed to the successful completion of our C programming project, "Collection of Games." This project would not have been possible without the guidance and support of our instructors, peers, and online resources.

First, we thank our C Programming instructor, Prajwal Pakka, for their mentorship and for encouraging us to explore practical applications of programming concepts. Their feedback was instrumental in refining our project. We also extend our appreciation to our classmates for their collaborative spirit and constructive suggestions during development.

Finally, we acknowledge the online tutorials, forums, and textbooks that provided foundational knowledge for implementing game logic, handling user input, and debugging. This project has been a valuable learning experience in software development and problem-solving.

Bishal Upadhyay (THA081BEI010)
Devendra Yadav (THA081BEI011)
Muhammad Shahid Raain (THA081BEI020)
Ajay Kumar Pasi (THA081BEI003)

ABSTRACT

This project aims to develop a collection of six interactive games using the C programming language: **Number Guessing Game**, **Rock Paper Scissors**, **Tic-Tac-Toe**, **Hangman**, **Snake and Ladder** and **Room Escape Game**. Each game incorporates unique mechanics, such as random number generation, turn-based logic, and puzzle-solving, to engage users in a command-line interface. The project emphasizes core programming concepts, including loops, conditional statements, arrays and functions, while fostering problem-solving skills. By integrating error handling and user-friendly menus, this suite of games serves as both an educational tool for beginners and an entertaining application for users.

Keywords: C programming, game development, CLI, error handling

Contents

1. INTRODUCTION

1.1 Background

Games are powerful tools for learning programming logic, problem-solving, and user interaction. CLI-based games, though simple, offer a practical platform for applying programming concepts like loops, conditionals, and data structures. This project leverages the C language's efficiency to create a suite of games that balance education and entertainment.

1.2 Motivation

This project provides hands-on experience in developing interactive applications, reinforcing programming fundamentals. It challenges beginners to design varied game mechanics, debug code, and optimize user experience, preparing them for complex software projects.

1.3 Problem Definition

Developing a cohesive collection of games requires addressing diverse logic structures, input validation, and user interface design. Challenges include implementing random number generation (e.g., Hangman), turn-based logic (Tic-Tac-Toe) and puzzle-solving (Room Escape).

1.4 Objectives

- Develop six functional CLI games with distinct mechanics.
- Design an intuitive menu system for game selection.
- Implement error handling for invalid inputs.
- Ensure code modularity and efficiency.
- Demonstrate mastery of C programming basics.

2. LITERATURE REVIEW

2.1 Historical Development of CLI Games

Early computer games, like *Hunt the Wumpus* (1972), were text-based and laid the groundwork for CLI game design. These games emphasized logic over graphics, making them ideal for programming practice.

2.2 Educational Value of Simple Games

Studies show that game-based learning enhances engagement and retention in programming education. Games like Hangman and Tic-Tac-Toe are widely used to teach algorithms and debugging [1].

2.3 Game Development in C

C's low-level capabilities and standard libraries (e.g., stdlib.h, time.h) enable efficient implementation of game mechanics, such as random number generation (rand()) and input handling [2].

3. PROPOSED SYSTEM ARCHITECTURE

3.1 System Flow

User → Main Menu → Select Game → Play → Exit

3.2 Game Modules

- **Menu System**: Navigate between games.
- **Game Logic**: Unique rules for each game.
- **Error Handling**: Validate inputs (e.g., letters in Hangman).
- **Score Tracking**: Record wins/losses (Snake and Ladder).

3.3 Tools and Environment

- **Compiler**: GCC (Linux/Windows).
- **IDE**: Code::Blocks, VS Code.

4. METHODOLOGY

4.1 Header Files

- stdio.h, stdlib.h, time.h, string.h.

4.2 Functions and Logic

- **Randomization**: rand() for number generation.
- **Loops**: while for game continuity.
- **Arrays**: Store game boards (Tic-Tac-Toe).

4.3 Error Handling

- Input validation (e.g. numeric checks in Snake and Ladder).

5. SCOPE AND APPLICATIONS

- **Scope**: Limited to CLI; no graphical elements.
- **Applications**: Educational tool for programming students, recreational use.

6. TIME ESTIMATION

Task	**Week 1**	**Week 2**	**Week 3**	**Week 4**
Menu System	✓			
Basic Games (3)	✓	✓		
Advanced Games (3)		✓	✓	
Testing & Debugging			✓	✓

7. FEASIBILITY ANALYSIS

7.1 Technical Feasibility

C’s standard libraries support game mechanics. No advanced hardware required.

7.2 Economic Feasibility

Free tools (GCC, Code::Blocks) ensure zero cost.

7.3 Schedule Feasibility

Four weeks sufficient for development and testing.

REFERENCES

[1] Smith, J. *Game-Based Learning in CS Education*. ACM, 2018.
[2] Kernighan, B. *The C Programming Language*. Prentice Hall, 1988.