**Pong Game**

Afrin IT-23039

September 2024

Department Of Information & Communication Techology

Mawlana Bhasani Science & Technology University

Santosh Tangail

**1. Project Title:**

**Pong Game: A Classic Arcade Recreation**

**2. Overview:**

This project aims to develop a digital version of the classic arcade game, Pong, using C++ and console-based graphics. Pong is a two-player game where each player controls a paddle, attempting to hit the ball back and forth without missing. The objective is to score points by making the opponent miss the ball. This proposal outlines the game mechanics, key features, and the technical stack for the project.

**3. Objective:**

The primary goal of the project is to:

* Develop a console-based Pong game that simulates the original arcade game.
* Provide an engaging user experience with responsive controls and smooth gameplay.
* Implement two-player functionality, scoring, and game-over conditions.

**4. Game Features:**

**4.1 Core Mechanics:**

* **Paddle Control**:  
  Each player controls a paddle:
  + Player 1 uses the W (up) and S (down) keys to move their paddle.
  + Player 2 uses the Up Arrow (up) and Down Arrow (down) keys for movement.
* **Ball Movement**:  
  The ball moves diagonally and bounces off the walls. If it hits a paddle, it bounces back. If a player misses, the opponent scores a point.
* **Scoring System**:
  + The game keeps track of each player's score.
  + The first player to reach a pre-set score limit (e.g., 10) wins the game, or the game can be played indefinitely with a manual quit option.

**4.2 Game Loop:**

* Continuous game loop that updates the ball’s position, checks for collisions with paddles or walls, updates the score, and refreshes the game screen.

**4.3 Game UI:**

* Console-based interface with basic graphics:
  + Paddles: Represented by vertical bars (||).
  + Ball: Represented by the character O.
  + Walls: Boundaries on the top and bottom represented by #.
  + Score: Displayed at the top of the screen for both players.

**4.4 Keyboard Input:**

* Real-time, non-blocking keyboard input handling to control paddle movements using \_kbhit() and \_getch() functions.

**4.5 Game Modes:**

* **Two-Player Mode**: Two human players compete against each other on the same machine.
* **Single-Player Mode** (optional enhancement): One player can compete against a computer-controlled paddle using basic AI logic.

**4.6 Game Over and Restart:**

* The game ends when a player reaches a winning score or presses the Esc key to quit.
* Display the final score and an option to restart or exit.

**5. Technical Requirements:**

**5.1 Development Platform:**

* **Programming Language**: C++
* **Development Environment**: Windows console application
* **Libraries**:
  + conio.h: For real-time keyboard input.
  + windows.h: For console screen manipulation (cursor movement, screen clearing).

**5.2 Target Platform:**

* **Operating System**: Windows (due to dependencies on windows.h and conio.h libraries).
* **Console-Based UI**: The game will run in a command prompt or terminal window.

**6. Project Timeline:**

| **Task** | **Duration** |
| --- | --- |
| Game Design & Planning | 1 week |
| Setup Development Environment | 1 day |
| Basic Paddle and Ball Logic | 1 week |
| Implement Scoring System | 2 days |
| Add Wall Collision & Ball Bouncing | 3 days |
| Refine Graphics & UI | 3 days |
| Add Player Controls | 2 days |
| Final Testing & Debugging | 1 week |
| Optional Features (AI, Settings) | 1 week |

**7. Potential Extensions (Optional):**

* **AI Opponent**: Implement basic AI for single-player mode.
* **Difficulty Levels**: Introduce varying ball speeds or paddle sizes for different difficulty settings.
* **Power-Ups**: Add special elements like speed boosts or paddle size changes for a more dynamic gameplay.
* **Sound Effects**: Add sound effects for paddle hits and scoring.

**8. Project Deliverables:**

* **Executable Pong Game**: A fully functional Pong game playable in the console.
* **Game Documentation**: Instructions for installation, usage, and controls.
* **Source Code**: Well-documented and modular source code.

**9. Risk Assessment:**

* **Console Limitations**: The game will rely on console-based rendering, which may limit advanced graphics and sound, but ensures compatibility with lower-end systems.
* **Cross-Platform Compatibility**: The current implementation targets Windows. A separate version using platform-independent libraries may be needed for other operating systems.

**10. Conclusion:**

This project will create a simple, interactive Pong game that recreates the retro arcade experience in a modern console environment. It’s a fun, easy-to-understand project that demonstrates real-time input handling, collision detection, and basic game loop design, which can be extended for additional features later.