# BRICK BREAKER GAME IN C++

**SRS**

HATEEM TAHIR (032)
HARIS AWAN (023)
ABDULAH KHALID (035)
HAMZA RIZWAN (005)

# TABLE OF CONTENTS

8. EXTERNAL INTERFACE REQUIREMENTS
    8.1. Input
    8.2. Output

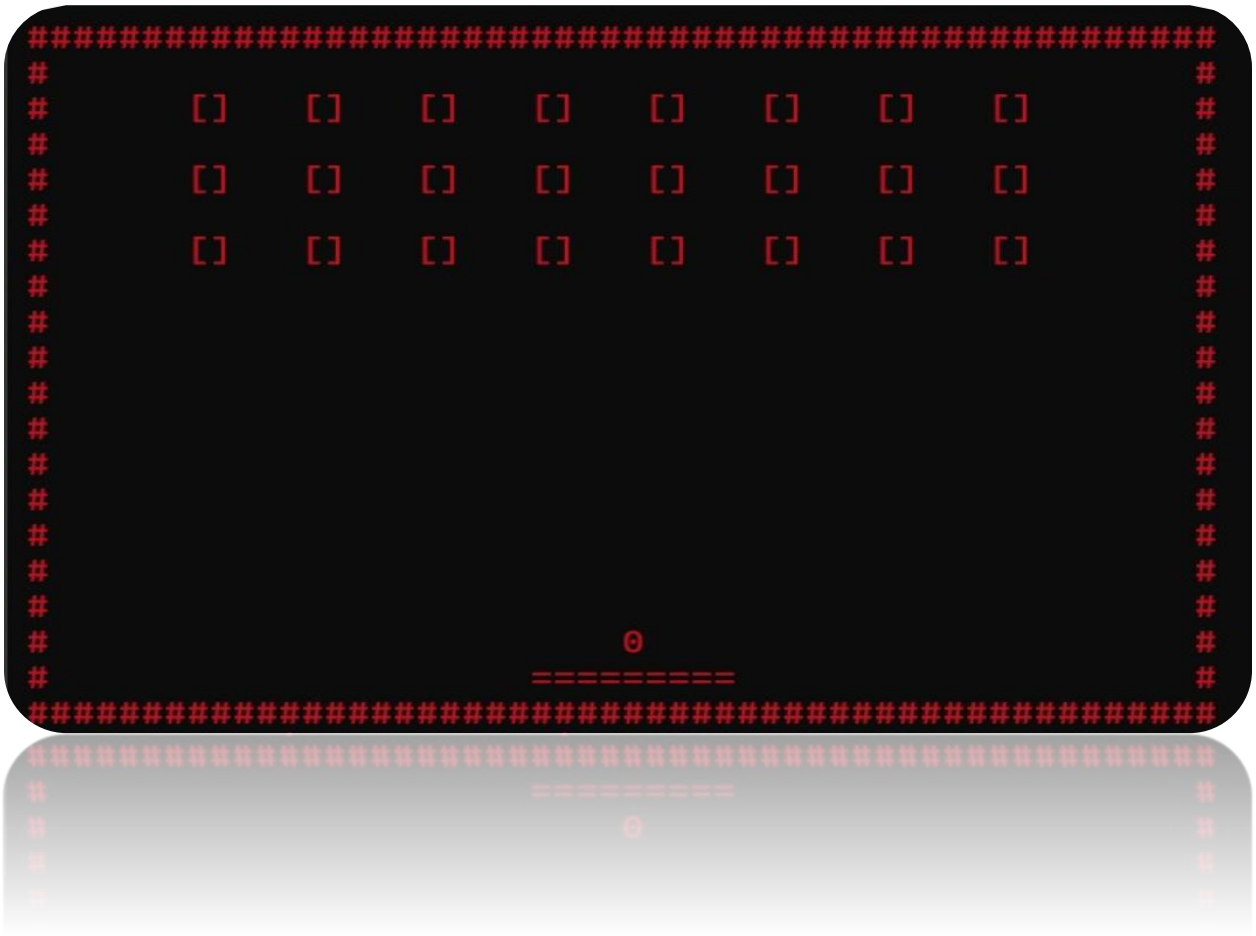
# SOFTWARE REQUIREMENTS SPECIFICATION BRICK BREAKER GAME

## 1. INTRODUCTION

The Brick Breaker game is a console-based game where the player controls a slider to bounce a ball to break bricks. The game ends when the player loses all lives or breaks all the bricks. The game includes a power-up feature, a scoring system, and multiple levels.

## 2. GAME FEATURES

- **Bricks**: The game has a set of bricks arranged on the screen. These bricks break when the ball hits them.



- **Ball**: The ball bounces off walls, bricks, and the slider.
- **Slider**: The player can move a horizontal slider left or right to bounce the ball.
- **Power-up**: A power-up is available when a certain number of bricks are broken, making it easier for the player to win.
- **Lives**: The player starts with a number of lives. The player loses a life if the ball falls below the screen.


- **Levels**: The game progresses through levels. After clearing all bricks, the player advances to the next level with increased difficulty (faster ball speed).

## 3. SYSTEM REQUIREMENTS

- **Platform**: Windows (console-based)
- **Programming Language**: C++
- **Libraries**:
    - windows.h for console management.
    - conio.h for keyboard input.

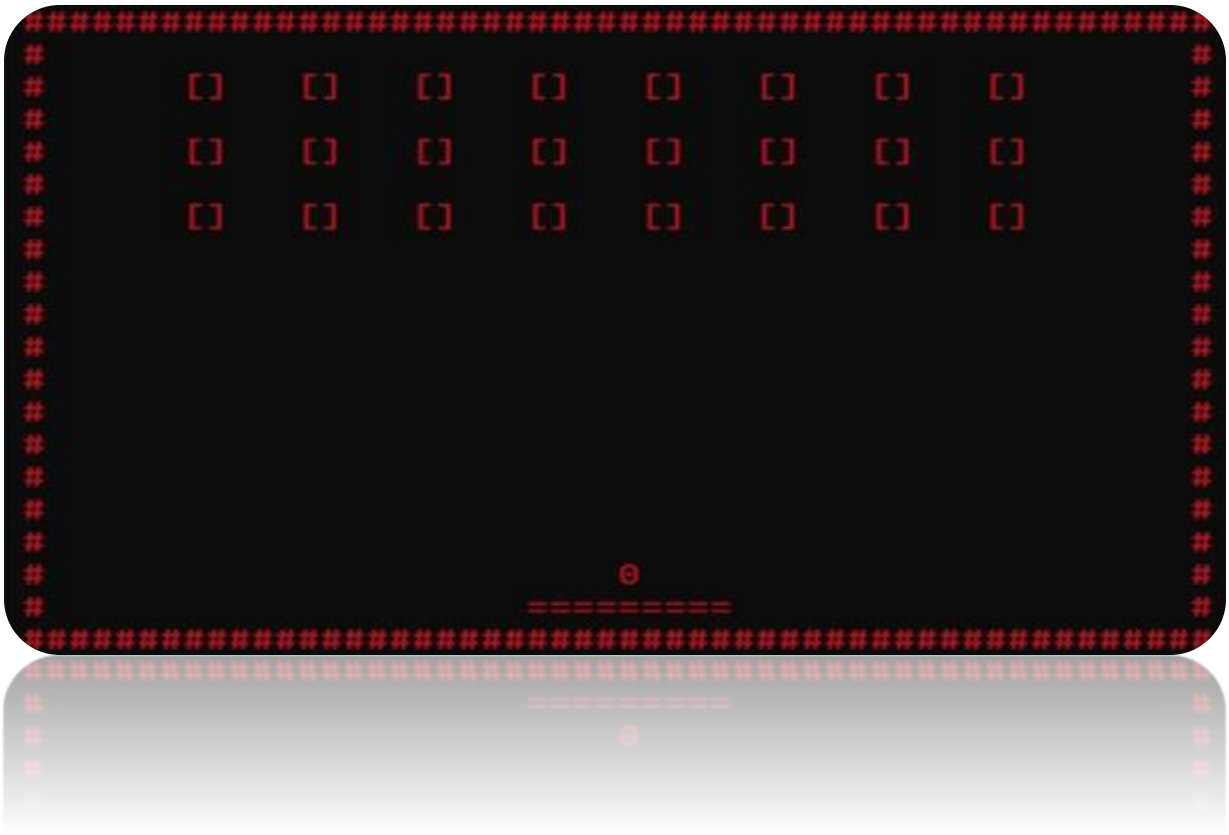
## 4. FUNCTIONAL REQUIREMENTS

### 4.1 MAIN MENU

- The user is shown a main menu with the following options:
    - **Start Game**: Begins the game.
    - **Instructions**: Displays the game instructions.
    - **Quit**: Exits the game.

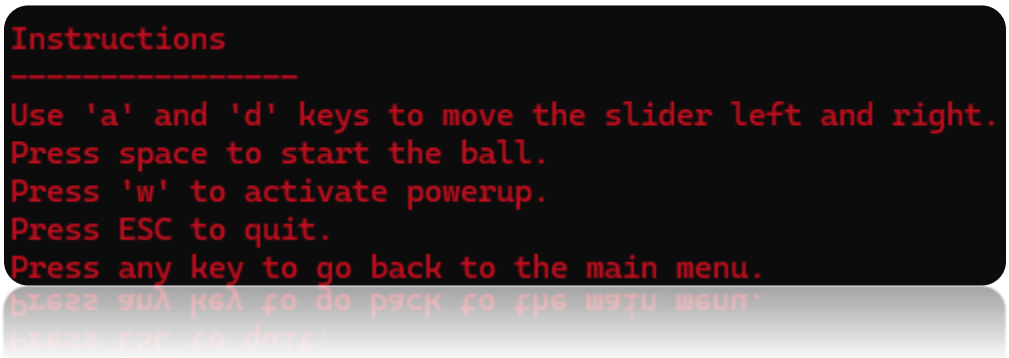

### 4.2 GAME START

- When the game starts, the screen is cleared, and the player sees the following:
    - A border around the screen.
    - Bricks arranged in rows.
    - A slider at the bottom of the screen.
    - A ball ready to start.



### 4.3 PLAYER CONTROLS

- The player can control the slider using:
    - **'a'** to move the slider left.
    - **'d'** to move the slider right.
    - **Spacebar (32)** to start the ball.
    - **'w'** to activate the power-up (if available).
    - **ESC** to exit the game.



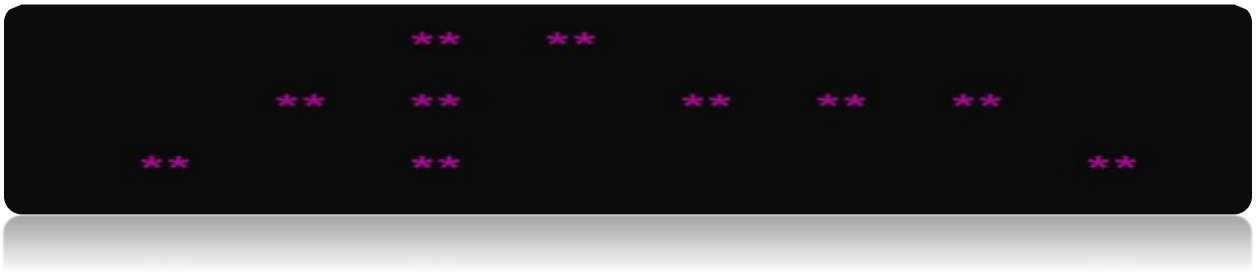
### 4.4 GAME MECHANICS

- The ball bounces off:
    - **Walls**: Reverses horizontal or vertical direction.
    - **Slider**: Bounces back vertically.
    - **Bricks**: Breaks the brick and bounces back.


### 4.5 POWER-UP

- A power-up appears when only 10 or fewer bricks remain.


- Activating the power-up removes all remaining bricks and gives extra points.



### 4.6 GAME OVER

- The game ends when:
    - The player runs out of lives.
    - All bricks are broken.
- The player can choose to restart the game after a game over.

## 5. NON-FUNCTIONAL REQUIREMENTS

- **Performance**: The game should run smoothly on a typical Windows machine.
- **User Interface**: The interface should be clear, with the ball, slider, and bricks easy to see and control.
- **Usability**: The game should be easy to start, play, and exit using simple keyboard controls.

## 6. SYSTEM DESIGN

### 6.1 CONSOLE MANAGEMENT

- Use SetConsoleCursorPosition to move the cursor and draw objects at specific locations.
- SetConsoleTextAttribute is used to change text color (e.g., for the power-up animation).

### 6.2 GAME LOGIC

- **Bricks**: The game uses a list of bricks and a set to track visible bricks. When a brick is hit by the ball, it is removed from the screen and added to the score.
- **Ball Movement**: The ball moves in a specific direction, bouncing off walls, bricks, and the slider. When the ball collides with a brick or the slider, its direction changes.
- **Lives**: Each time the ball falls below the screen, the player loses a life. The game continues until the player runs out of lives or breaks all the bricks.
- **Level Progression**: After completing a level by breaking all the bricks, the game progresses to the next level, increasing the difficulty by speeding up the ball.

## 7. LIMITATIONS AND ASSUMPTIONS

- **Platform Limitation**: The game is designed to run on Windows OS with console support.
- **Game Speed**: The speed of the ball is adjusted with each level to increase difficulty.
- **Controls**: The game is played using basic keyboard keys.

## 8. EXTERNAL INTERFACE REQUIREMENTS

- **Input**: Keyboard inputs (a, d, spacebar, w, ESC).
- **Output**: Console output for game status, including score, lives, and level.