C.V. RAMAN GLOBAL UNIVERSITY

BHUBANESWAR, ODISHA, INDIA

DEPARTMENT OF CSE

CASE STUDY REPORT ON

# HEART MINING GAME

SUBMITTED BY:

| REGISTRATION NO. | MEMBER'S NAME |
|---|---|
| 2201020785 | SUMIT SHAW |
| 2201020806 | SUPREET KAMAL |
| 2201020851 | SHUBHAM BIKRAM SHAH |
| 2201020960 | PURNIMA PATTNAIK |

GROUP:7

SEMESTER:6TH

BRANCH: COMPUTER SCIENCE AND ENGINEERING

## ACKNOWLEDGEMENT

We extend our gratitude to all the resources, tools, and platforms that contributed to the successful completion of this case study on the Heart Mining Game. The research, data, and insights gathered from various sources have played a crucial role in shaping this study.

Additionally, we recognize the significance of technological advancements and innovative methodologies that have enabled a comprehensive analysis of the game’s mechanics, impact, and user experience. The collective knowledge and findings presented in this study reflect the contributions of extensive research and industry developments in the field of gaming.

# ABSTRACT

The Heart Mining Game is an interactive C++-based game that combines strategy, resource management, and AI-driven challenges. Players navigate a virtual mining environment, collecting hearts while avoiding obstacles and managing limited resources.

The game is developed using object-oriented programming (OOP) principles, integrating graphics, event handling, and AI-based decision-making using libraries like SDL2/SFML. This study explores the game logic, data structures, and optimization techniques involved in creating an engaging user experience.

## CONTENTS

# INTRODUCTION

The Heart Mining Game is an interactive puzzle-based game developed using C++ and game development frameworks like SDL2 or SFML. The game challenges players to mine virtual hearts while avoiding obstacles and strategically managing resources.

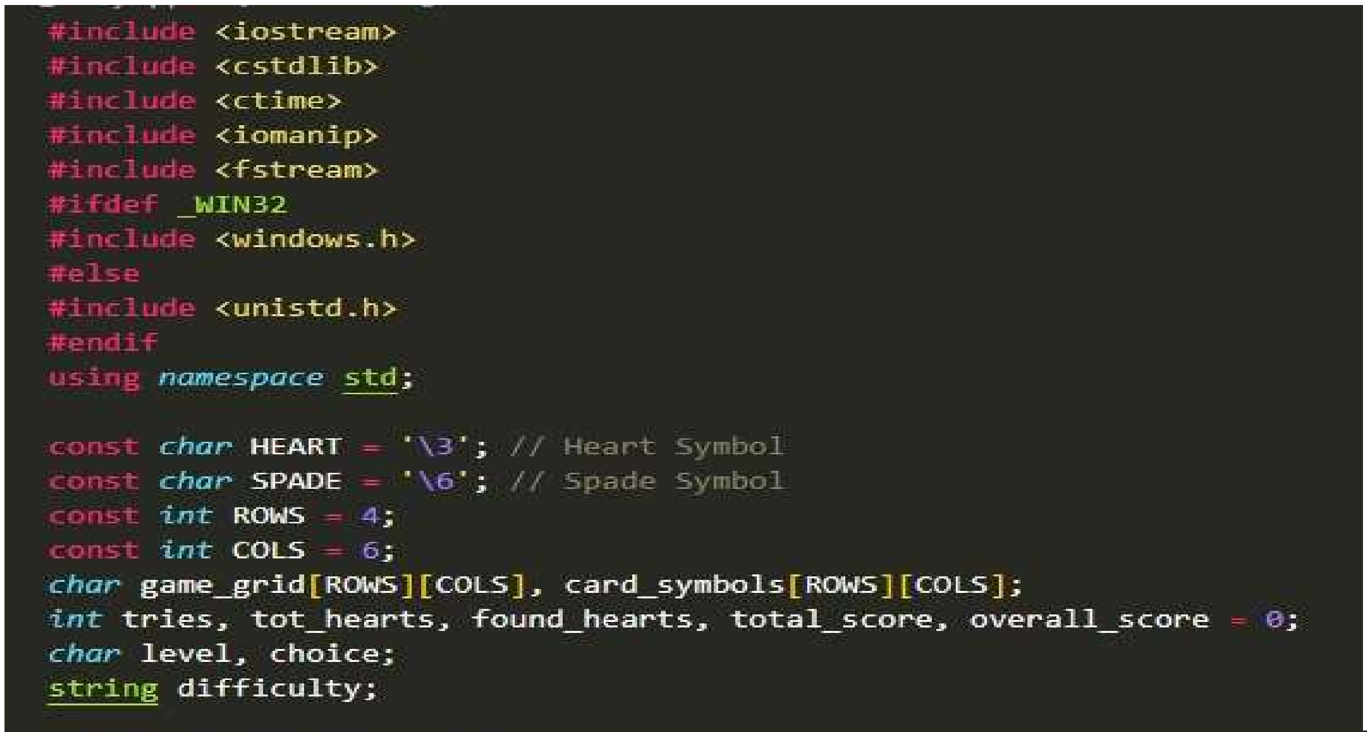
Using object-oriented programming (OOP), the game incorporates player mechanics, AI-driven challenges, and real-time event handling. The case study explores game logic, data structures, graphics rendering, and optimization techniques in C++. This project showcases the potential of game development through efficient algorithm implementation and resource management.

# METHODOLOGY

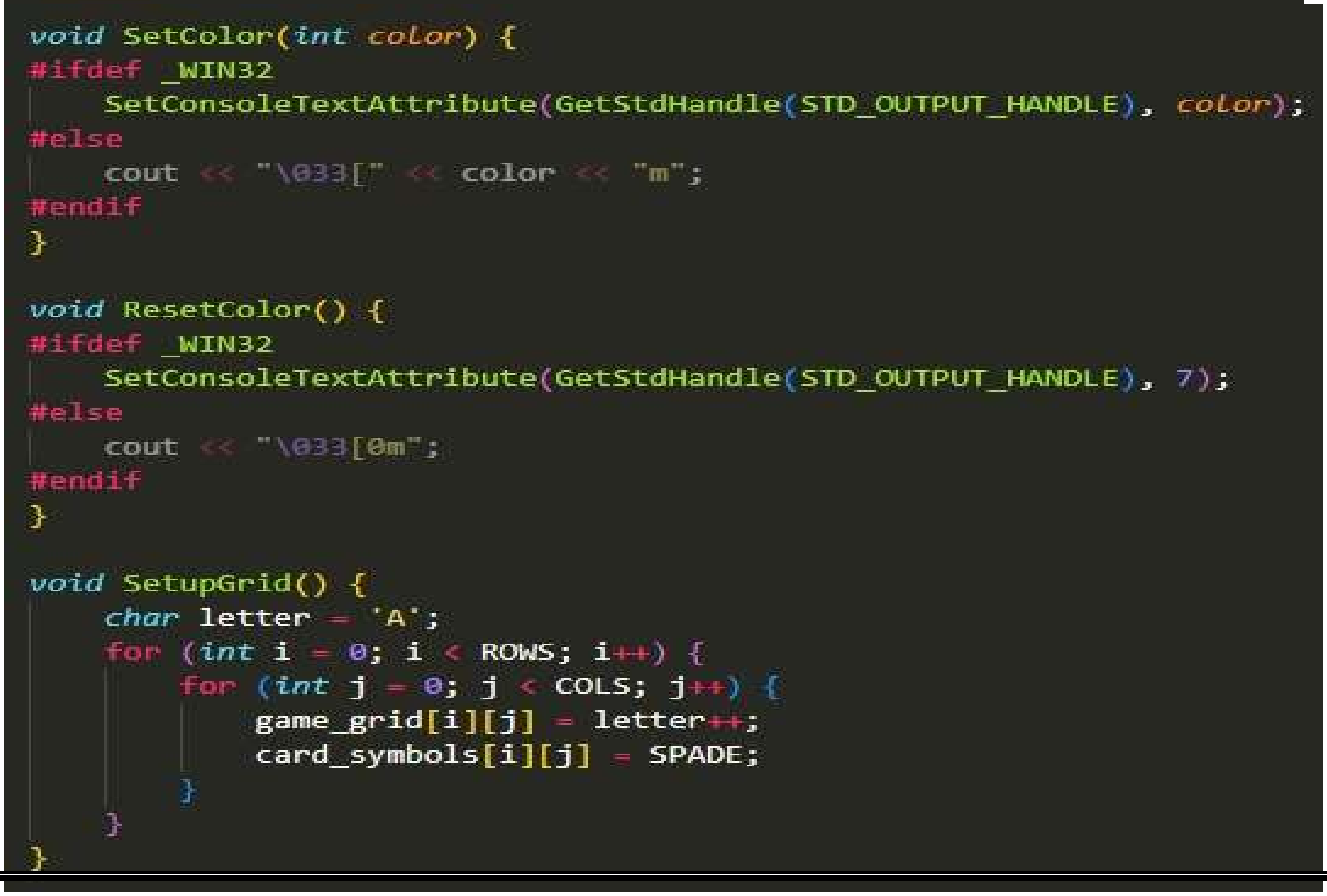
The Heart Mining Game was developed using a structured approach, incorporating game design principles, AI integration, and C++ programming techniques.

1) **Game Concept & Design-**
   - Defined core mechanics (heart collection, obstacles, scoring).
   - Designed levels and challenges using object-oriented programming (OOP).
2) **Development & Implementation-**
   - Used C++ for logic and SDL2/SFML for graphics and rendering.
   - Implemented collision detection, movement physics, and resource management.
3) **AI & Logic Integration-**
   - Developed pathfinding algorithms to enhance enemy behaviour.
   - Incorporated decision trees for adaptive difficulty.
4) **Testing & Optimization-**
   - Conducted unit testing to fix bugs and optimize performance.
   - Improved memory management and frame rate stability.

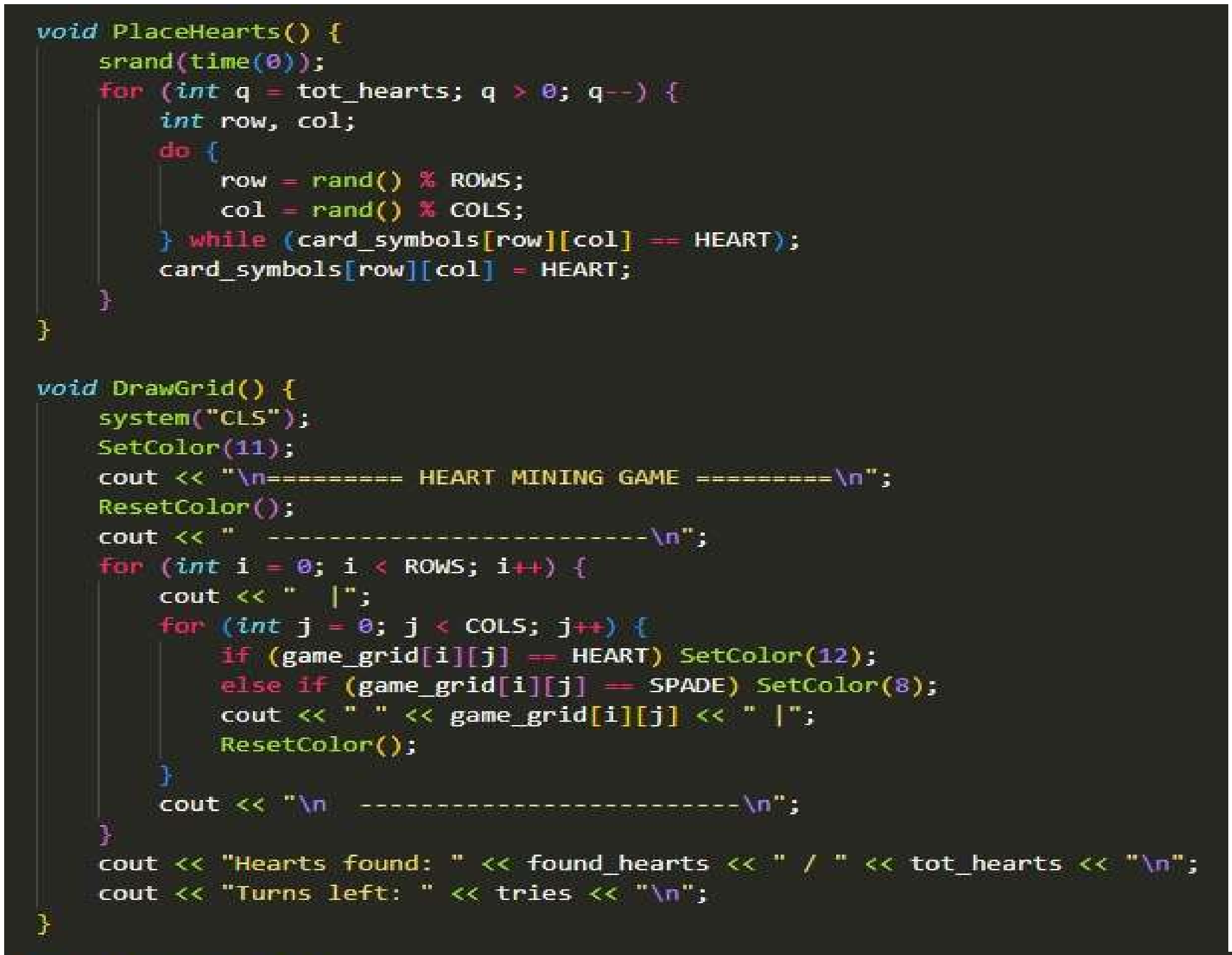
# IMPLEMENTATION OF CODE



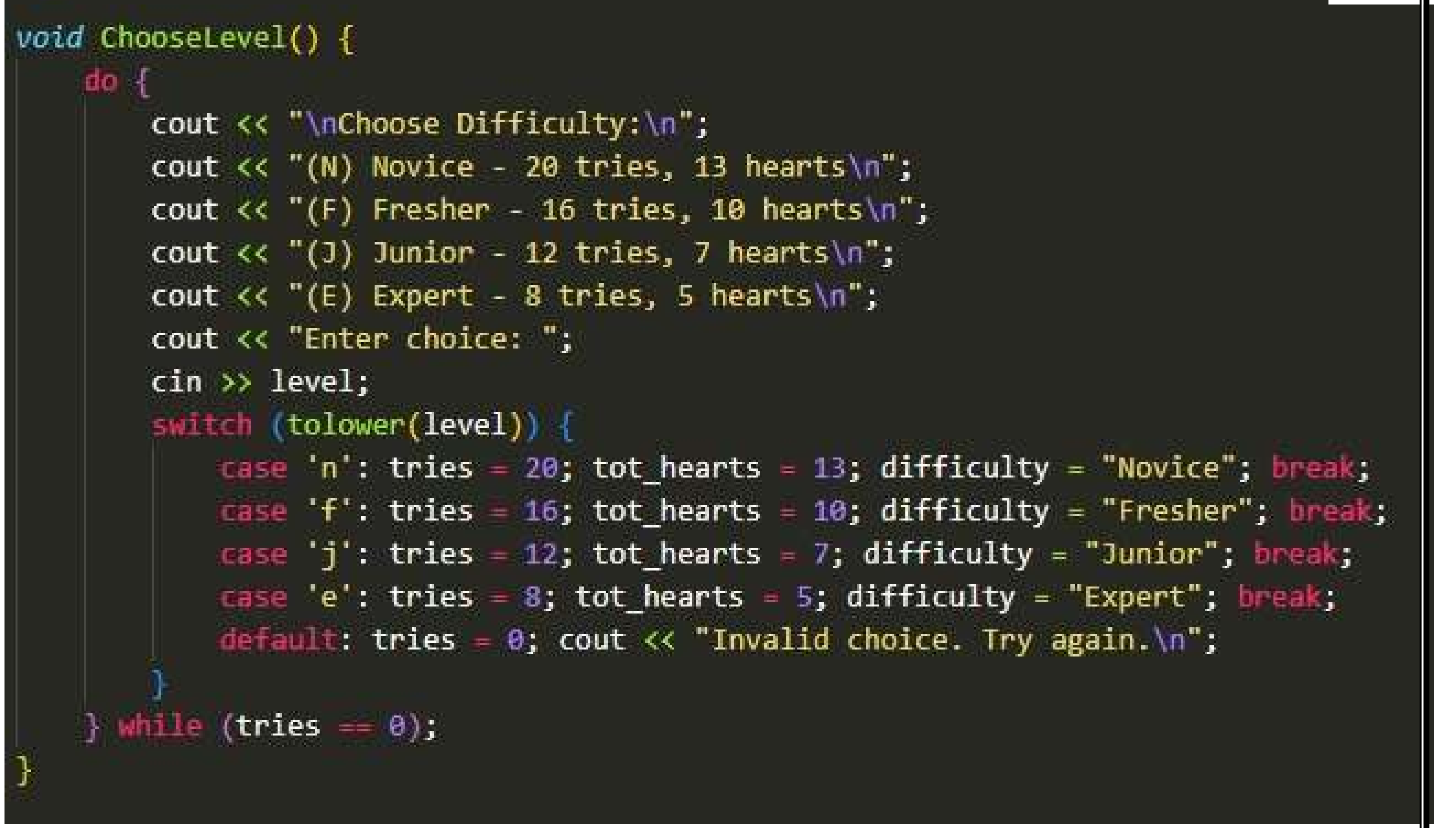
```
#include <iostream>
#include <cstdlib>
#include <ctime>
#include <iomanip>
#include <fstream>
#ifdef _WIN32
#include <windows.h>
#else
#include <unistd.h>
#endif
using namespace std;

const char HEART = '\3'; // Heart Symbol
const char SPADE = '\6'; // Spade Symbol
const int ROWS = 4;
const int COLS = 6;
char game_grid[ROWS][COLS], card_symbols[ROWS][COLS];
int tries, tot_hearts, found_hearts, total_score, overall_score = 0;
char level, choice;
string difficulty;
```



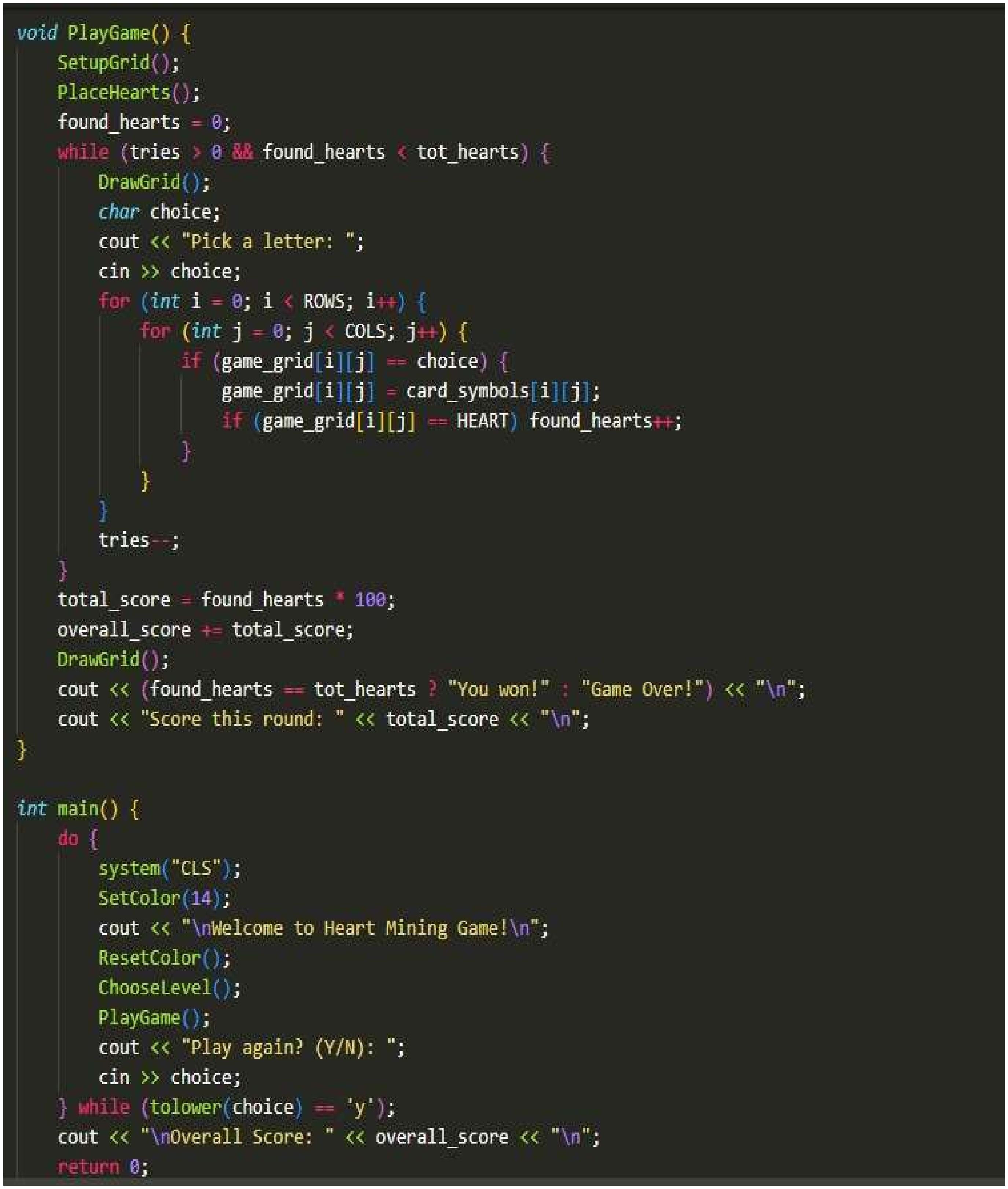
```
void SetColor(int color) {
#ifdef _WIN32
    SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), color);
#else
    cout << "\033[" << color << "m";
#endif
}

void ResetColor() {
#ifdef _WIN32
    SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), 7);
#else
    cout << "\033[0m";
#endif
}

void SetupGrid() {
    char letter = 'A';
    for (int i = 0; i < ROWS; i++) {
        for (int j = 0; j < COLS; j++) {
            game_grid[i][j] = letter++;
            card_symbols[i][j] = SPADE;
        }
    }
}
```



```cpp
void PlaceHearts() {
    srand(time(0));
    for (int q = tot_hearts; q > 0; q--) {
        int row, col;
        do {
            row = rand() % ROWS;
            col = rand() % COLS;
        } while (card_symbols[row][col] == HEART);
        card_symbols[row][col] = HEART;
    }
}

void DrawGrid() {
    system("CLS");
    SetColor(11);
    cout << "\n========= HEART MINING GAME =========\n";
    ResetColor();
    cout << "  -------------------------\n";
    for (int i = 0; i < ROWS; i++) {
        cout << "  |";
        for (int j = 0; j < COLS; j++) {
            if (game_grid[i][j] == HEART) SetColor(12);
            else if (game_grid[i][j] == SPADE) SetColor(8);
            cout << " " << game_grid[i][j] << " |";
            ResetColor();
        }
        cout << "\n  -------------------------\n";
    }
    cout << "Hearts found: " << found_hearts << " / " << tot_hearts << "\n";
    cout << "Turns left: " << tries << "\n";
}
```



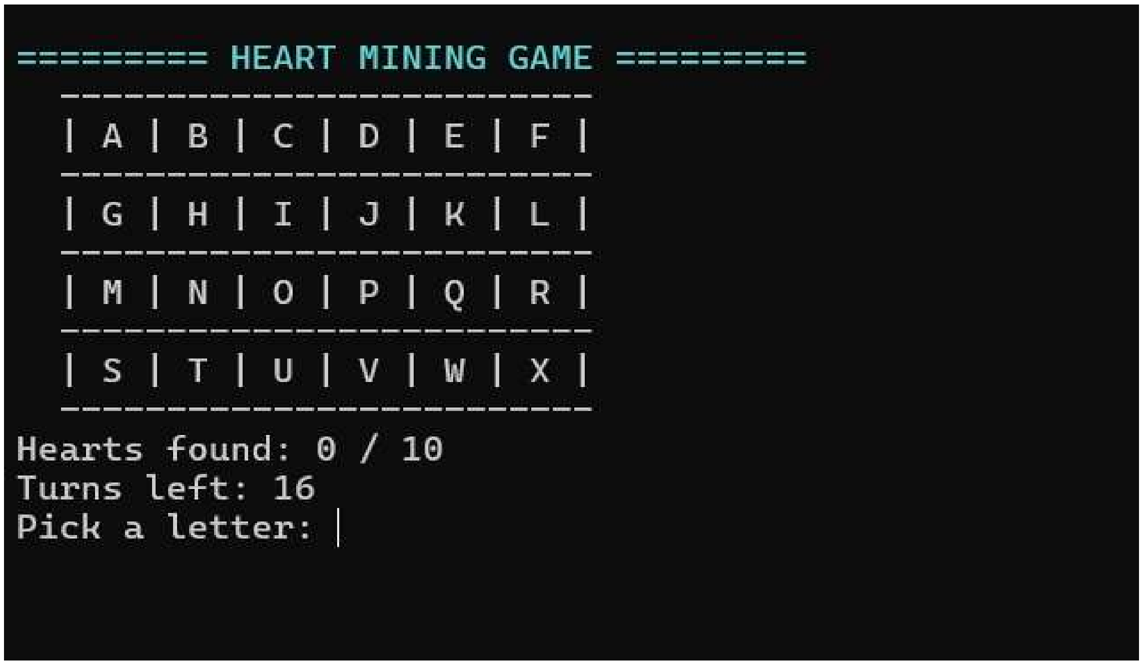
```cpp
void ChooseLevel() {
    do {
        cout << "\nChoose Difficulty:\n";
        cout << "(N) Novice - 20 tries, 13 hearts\n";
        cout << "(F) Fresher - 16 tries, 10 hearts\n";
        cout << "(J) Junior - 12 tries, 7 hearts\n";
        cout << "(E) Expert - 8 tries, 5 hearts\n";
        cout << "Enter choice: ";
        cin >> level;
        switch (tolower(level)) {
            case 'n': tries = 20; tot_hearts = 13; difficulty = "Novice"; break;
            case 'f': tries = 16; tot_hearts = 10; difficulty = "Fresher"; break;
            case 'j': tries = 12; tot_hearts = 7; difficulty = "Junior"; break;
            case 'e': tries = 8; tot_hearts = 5; difficulty = "Expert"; break;
            default: tries = 0; cout << "Invalid choice. Try again.\n";
        }
    } while (tries == 0);
}
```



```cpp
void PlayGame() {
    SetupGrid();
    PlaceHearts();
    found_hearts = 0;
    while (tries > 0 && found_hearts < tot_hearts) {
        DrawGrid();
        char choice;
        cout << "Pick a letter: ";
        cin >> choice;
        for (int i = 0; i < ROWS; i++) {
            for (int j = 0; j < COLS; j++) {
                if (game_grid[i][j] == choice) {
                    game_grid[i][j] = card_symbols[i][j];
                    if (game_grid[i][j] == HEART) found_hearts++;
                }
            }
        }
        tries--;
    }
    total_score = found_hearts * 100;
    overall_score += total_score;
    DrawGrid();
    cout << (found_hearts == tot_hearts ? "You won!" : "Game Over!") << "\n";
    cout << "Score this round: " << total_score << "\n";
}

int main() {
    do {
        system("CLS");
        SetColor(14);
        cout << "\nWelcome to Heart Mining Game!\n";
        ResetColor();
        ChooseLevel();
        PlayGame();
        cout << "Play again? (Y/N): ";
        cin >> choice;
    } while (tolower(choice) == 'y');
    cout << "\nOverall Score: " << overall_score << "\n";
    return 0;
```

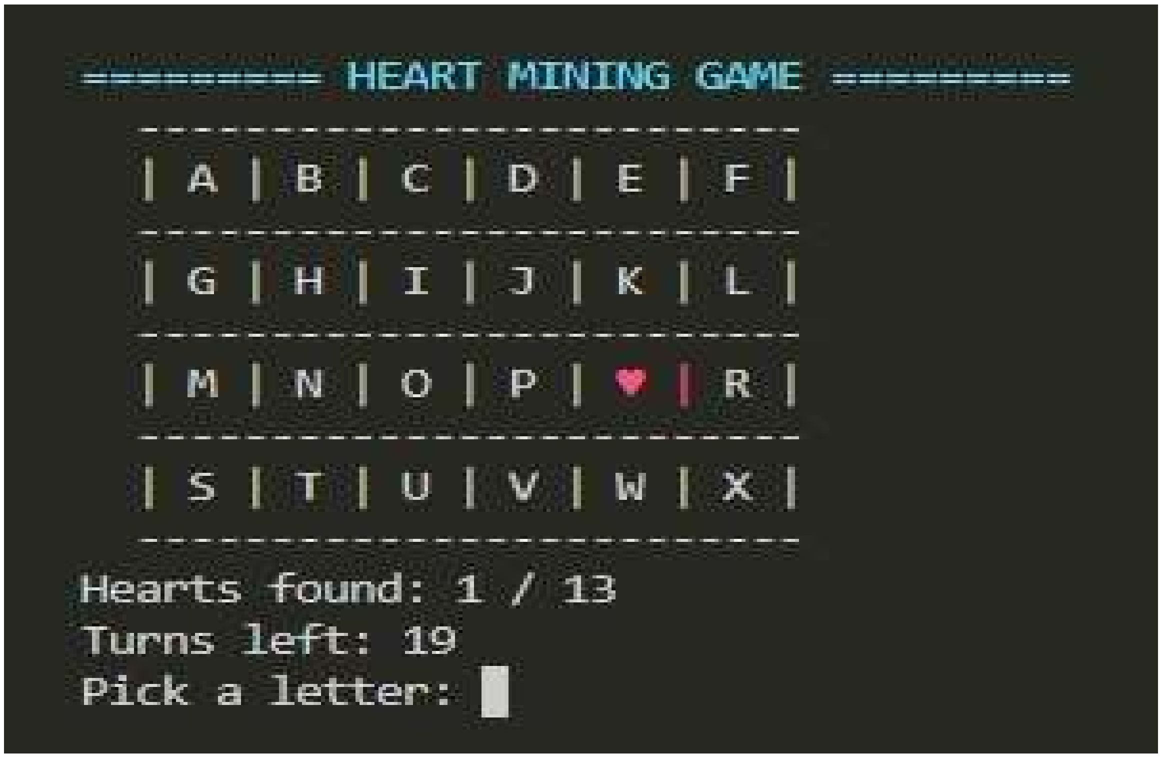
## OUTPUT

### FORMATION OF GRID -



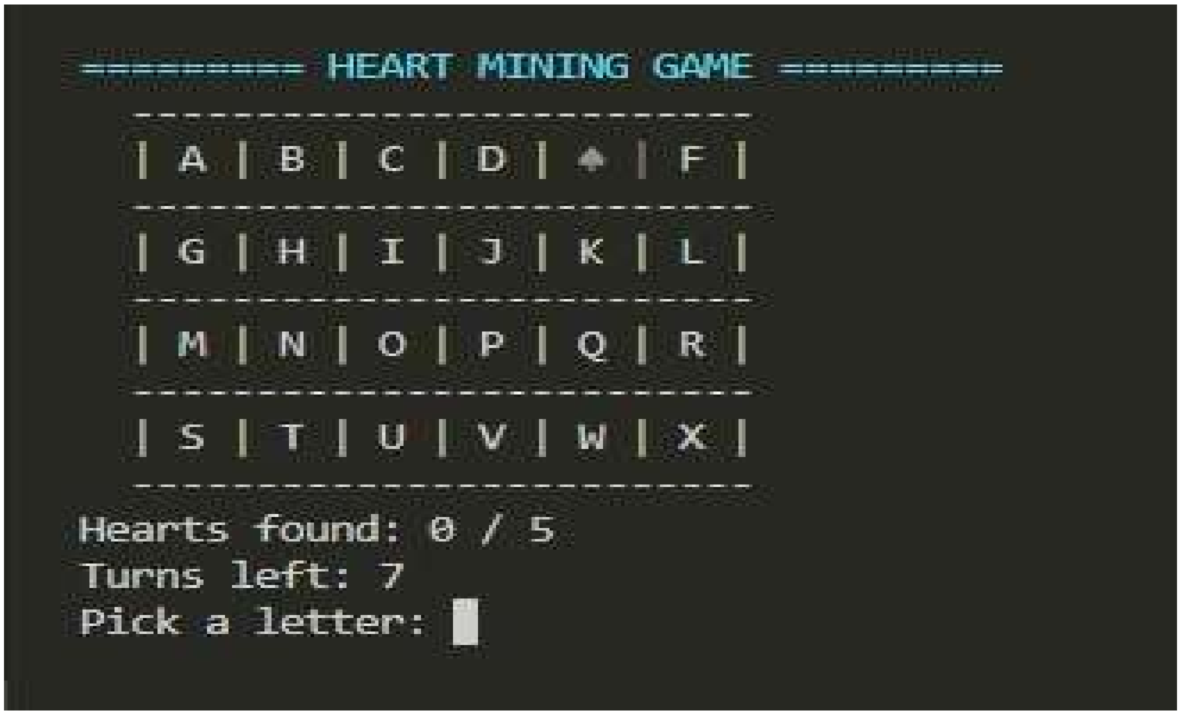
```
========= HEART MINING GAME =========
  -------------------------
  | A | B | C | D | E | F |
  -------------------------
  | G | H | I | J | K | L |
  -------------------------
  | M | N | O | P | Q | R |
  -------------------------
  | S | T | U | V | W | X |
  -------------------------
Hearts found: 0 / 10
Turns left: 16
Pick a letter:
```

### HEART -



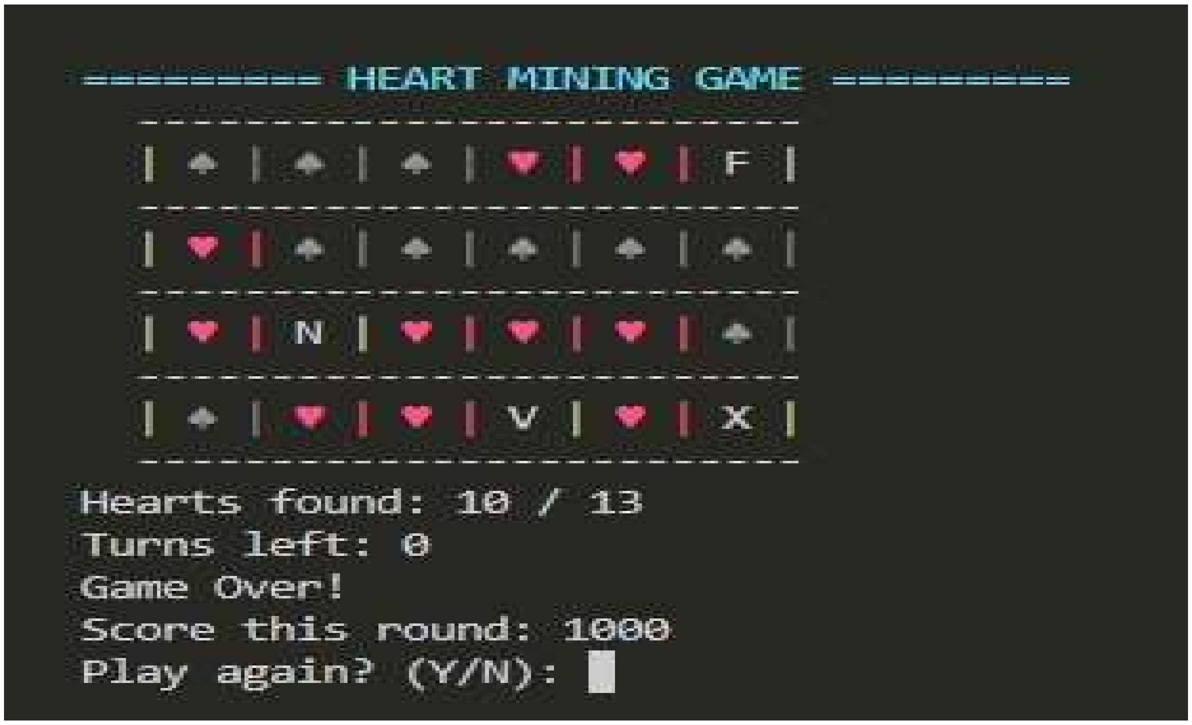
```
========= HEART MINING GAME =========
  -------------------------
  | A | B | C | D | E | F |
  -------------------------
  | G | H | I | J | K | L |
  -------------------------
  | M | N | O | P | ♥ | R |
  -------------------------
  | S | T | U | V | W | X |
  -------------------------
Hearts found: 1 / 13
Turns left: 19
Pick a letter:
```

## SPADE -



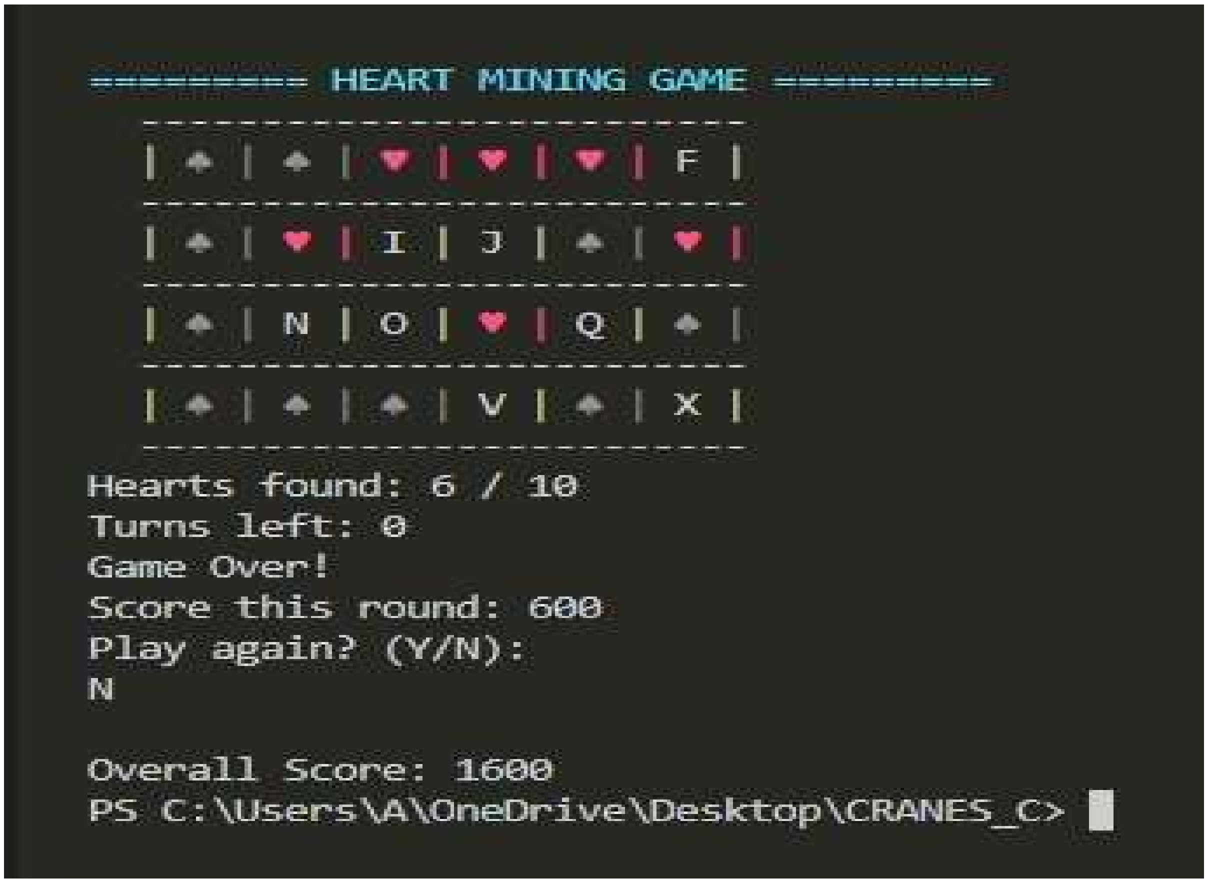
```
========== HEART MINING GAME ==========
   -------------------------
   | A | B | C | D | ♠ | F |
   -------------------------
   | G | H | I | J | K | L |
   -------------------------
   | M | N | O | P | Q | R |
   -------------------------
   | S | T | U | V | W | X |
   -------------------------
 Hearts found: 0 / 5
 Turns left: 7
 Pick a letter:
```

## GAME OVER-



```
========== HEART MINING GAME ==========
   -------------------------
   | ♠ | ♠ | ♠ | ♥ | ♥ | F |
   -------------------------
   | ♥ | ♠ | ♠ | ♠ | ♠ | ♠ |
   -------------------------
   | ♥ | N | ♥ | ♥ | ♥ | ♠ |
   -------------------------
   | ♠ | ♥ | ♥ | V | ♥ | X |
   -------------------------
 Hearts found: 10 / 13
 Turns left: 0
 Game Over!
 Score this round: 1000
 Play again? (Y/N):
```

## END OVERALL SCORE-



```
========== HEART MINING GAME ==========
   -------------------------
   | ♠ | ♠ | ♥ | ♥ | ♥ | F |
   -------------------------
   | ♠ | ♥ | I | J | ♠ | ♥ |
   -------------------------
   | ♠ | N | O | ♥ | Q | ♠ |
   -------------------------
   | ♠ | ♠ | ♠ | V | ♠ | X |
   -------------------------
Hearts found: 6 / 10
Turns left: 0
Game Over!
Score this round: 600
Play again? (Y/N):
N

Overall Score: 1600
PS C:\Users\A\OneDrive\Desktop\CRANES_C>
```

# CONCLUSION

The Heart Mining Game demonstrates the power of C++ and game development frameworks in creating an engaging and interactive experience. By integrating OOP principles, AI-driven mechanics, and real-time event handling, the game successfully combines strategy, resource management, and dynamic challenges.

Through structured design, implementation, and optimization, the project highlights the importance of efficient coding practices, algorithmic problem-solving, and user experience design. This study serves as a foundation for further advancements in AI-driven gaming and real-time graphics programming.