GAMING
GAMING NEXUS
COVER PAGE

SYNOPSIS

Table of Contents

Team

Introduction

Objectives

About The Project

GAMING NEXUS

Team

BATCH : F14

1. Aryan Varshney (992401030154)
2. Aneri Gupta (992401030164)
3. Tarushi Goel (992401030173)
4. Kartik Kalp Pandey (992401030175)

Introduction

In today's digital age, computer games serve as an effective platform for learning and applying core programming concepts. Our project, "Gaming Nexus", is a multi-game collection designed to demonstrate key aspects of programming, logic-building, and user interaction. The project includes seven different games, each showcasing various techniques such as recursion, random number generation, string manipulation, and user feedback.

The games in Game Zone range from word-based puzzles to classic strategy and memory games, providing a diverse and engaging experience for players :

Lucky 7 : A luck-based game where players aim to hit a range of above seven, below seven or seven through a random dice roll.

Scramble Word : In this word puzzle, players are presented with jumbled letters and must guess the original word.

Maze Game : A maze is generated using recursion, and players must navigate from start to finish, testing problem-solving skills.

Cards Matching : A memory-based game where players flip cards to find and match identical pairs.

Word-Guessing Game : Players guess a hidden word by suggesting letters, with limited incorrect guesses allowed.

Rock, Paper, Scissors : The traditional game where players choose between rock, paper, or scissors to outsmart their opponent.

Number Guessing Game : The player attempts to guess a randomly generated number, receiving hints after each guess.

Game Zone integrates fun and learning by utilizing essential programming principles. Through this project, we aim to demonstrate our proficiency in coding, problem-solving, and interactive design while creating an enjoyable user experience.

Objectives

1 Design and Develop Engaging Games : Create a collection of interactive games, including Lucky 7, Scramble Word, Maze Game, Cards Matching, Word Guessing, Rock-Paper-Scissors, and Number Guessing.

GAMING NEXUS

2

Implement Game Mechanics : Utilize appropriate programming techniques and algorithms to establish core functionalities and rules for each game.

3

Apply appropriate algorithms for game functionalities :
Lucky 7: Implement a random number generator for luck-based outcomes.
Maze Game: Use recursion to generate and solve mazes effectively.
Word Guessing Game: Develop algorithms for tracking guesses and remaining attempts.

4 User Interaction and Experience :
Enhance user experience that promote ease of use and engagement, ensuring clear instructions and feedback.

5 Conduct Thorough Testing : Test each game rigorously to identify and fix bugs, ensuring smooth gameplay and adherence to game rules.

6 Optimize Performance : Optimize code for efficient execution, especially in games involving recursion and real-time interactions.

About the Project

Player Registration : The project begins with a player registration system. Each new player will be required to register, providing essential information to create a unique account. Upon successful registration, the player will be given access to the gaming platform.

Assigned Balance : After registration, each player will be credited with an initial balance of Rs. 500. This balance serves as the starting point for all in-game activities and will be used to place bets or participate in games.

Player Login : Registered players will log in using their unique credentials (username and password). This system ensures that only the registered player can access their account, maintaining security and preventing unauthorized access.

Game Selection : Upon login, the player will be presented with a menu of games to choose from. They can participate in any game as long as they have a positive balance in their account. The player can continue to play until their balance reaches zero or until they decide to exit.

RULES :

The player places a bet on the game using a portion of their balance.

- If the player wins, the bet amount is doubled and added to their balance.
- If the player loses, the bet amount is deducted from their balance.

MySQL Database Integration : MySQL will be used to store all relevant data about the players, including registration details, balance, and game history. Each player's account is password-protected to ensure security.

If a player's account balance becomes zero, the system will automatically assign Rs. 250 when the player login again, to allow the player to continue enjoying the games.

The "GAMING NEXUS" project aims to provide an engaging and secure gaming platform for players. The integration of MySQL ensures that player data is securely stored and accessible, while the gameplay rules offer fair and exciting opportunities for the players' gaming experiences.

Topics of SDF 1 Used :

1. Arrays
2. Pointers
3. Functions
4. Flow of Control
5. Recursion
6. Structures

TABLES

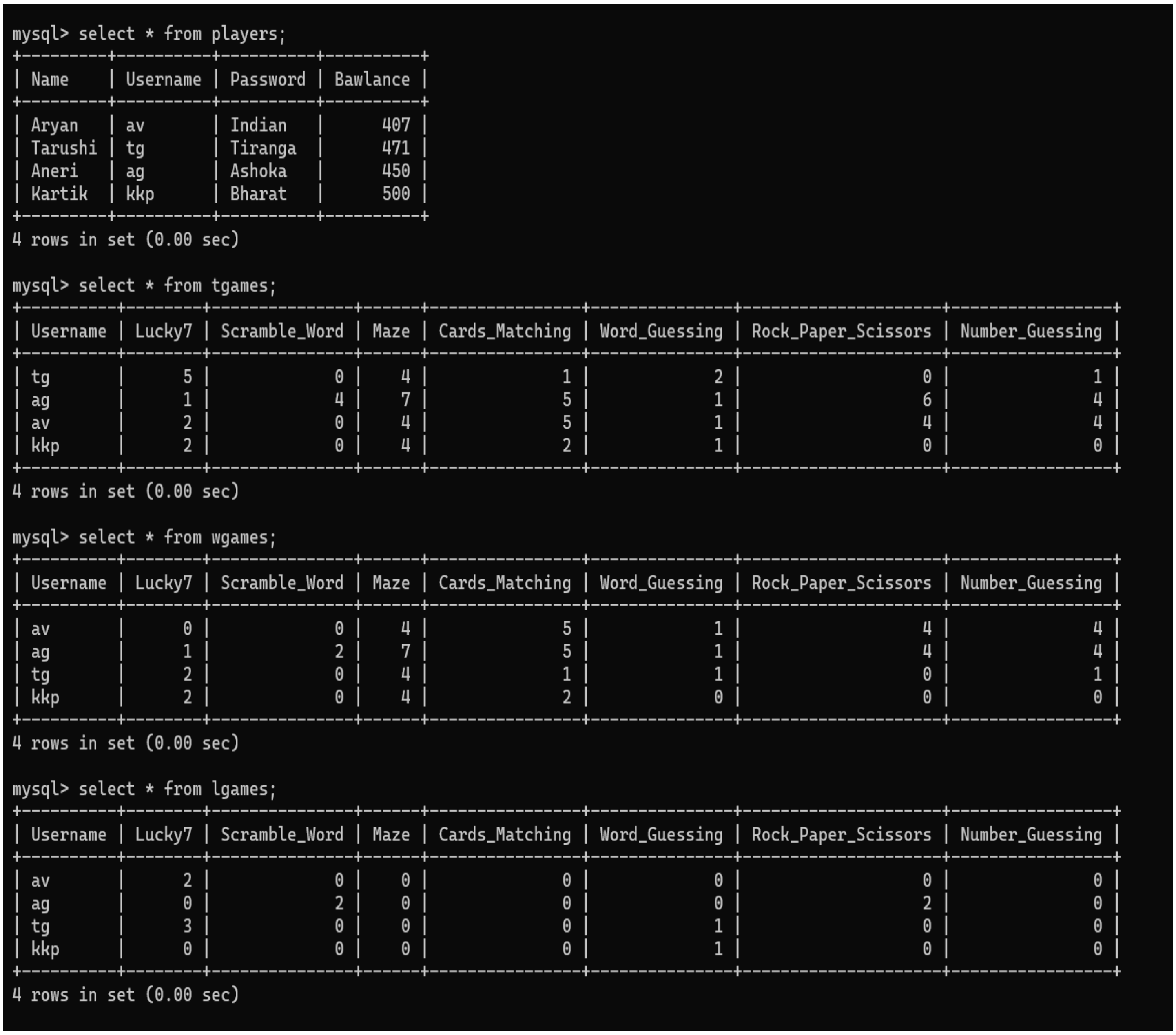
1. **Players** - for storing player details
2. **TGames** - for storing the count of total games played
3. **WGames** - for storing the count of games won
4. **LGames** - for storing the count of games lost

C CODE

```
#include <mysql.h>
void main ()
{
MYSQL *conn;
    char *server = "localhost";
    char *user= "root";
    char *password = "Taru@23sep";
    char *database = NULL;

    conn = mysql_init(NULL);
    conn = 
mysql_real_connect(conn,server,user,password,database,0,NULL,0);

    mysql_query(conn,"create database if not exists Gaming_Nexus");
    mysql_query(conn,"use Gaming_Nexus");
    mysql_query(conn,"create table if not exists Players(Name 
varchar(40), Username varchar(40), Password varchar(40), Balance 
int(5))");
    mysql_query(conn,"create table if not exists TGames(Username 
varchar(40), Lucky7 int(5), Scramble_Word int(5), Maze int(5), 
Cards_Matching int(5), Word_Guessing int(5), Rock_Paper_Scissors 
int(5), Number_Guessing int(5))");
    mysql_query(conn,"create table if not exists WGames(Username 
varchar(40), Lucky7 int(5), Scramble_Word int(5), Maze int(5), 
Cards_Matching int(5), Word_Guessing int(5), Rock_Paper_Scissors 
int(5), Number_Guessing int(5))");
```

```
    mysql_query(conn,"create table if not exists LGames(Username
varchar(40), Lucky7 int(5), Scramble_Word int(5), Maze int(5),
Cards_Matching int(5), Word_Guessing int(5), Rock_Paper_Scissors
int(5), Number_Guessing int(5))");

    mysql_commit(conn);
    mysql_close(conn);
}
```

MYSQL



```
mysql> select * from players;
+---------+----------+----------+----------+
| Name    | Username | Password | Bawlance |
+---------+----------+----------+----------+
| Aryan   | av       | Indian   |      407 |
| Tarushi | tg       | Tiranga  |      471 |
| Aneri   | ag       | Ashoka   |      450 |
| Kartik  | kkp      | Bharat   |      500 |
+---------+----------+----------+----------+
4 rows in set (0.00 sec)

mysql> select * from tgames;
+----------+--------+---------------+------+----------------+---------------+---------------------+-----------------+
| Username | Lucky7 | Scramble_Word | Maze | Cards_Matching | Word_Guessing | Rock_Paper_Scissors | Number_Guessing |
+----------+--------+---------------+------+----------------+---------------+---------------------+-----------------+
| tg       |      5 |             0 |    4 |              1 |             2 |                   0 |               1 |
| ag       |      1 |             4 |    7 |              5 |             1 |                   6 |               4 |
| av       |      2 |             0 |    4 |              5 |             1 |                   4 |               4 |
| kkp      |      2 |             0 |    4 |              2 |             1 |                   0 |               0 |
+----------+--------+---------------+------+----------------+---------------+---------------------+-----------------+
4 rows in set (0.00 sec)

mysql> select * from wgames;
+----------+--------+---------------+------+----------------+---------------+---------------------+-----------------+
| Username | Lucky7 | Scramble_Word | Maze | Cards_Matching | Word_Guessing | Rock_Paper_Scissors | Number_Guessing |
+----------+--------+---------------+------+----------------+---------------+---------------------+-----------------+
| av       |      0 |             0 |    4 |              5 |             1 |                   4 |               4 |
| ag       |      1 |             2 |    7 |              5 |             1 |                   4 |               4 |
| tg       |      2 |             0 |    4 |              1 |             1 |                   0 |               1 |
| kkp      |      2 |             0 |    4 |              2 |             0 |                   0 |               0 |
+----------+--------+---------------+------+----------------+---------------+---------------------+-----------------+
4 rows in set (0.00 sec)

mysql> select * from lgames;
+----------+--------+---------------+------+----------------+---------------+---------------------+-----------------+
| Username | Lucky7 | Scramble_Word | Maze | Cards_Matching | Word_Guessing | Rock_Paper_Scissors | Number_Guessing |
+----------+--------+---------------+------+----------------+---------------+---------------------+-----------------+
| av       |      2 |             0 |    0 |              0 |             0 |                   0 |               0 |
| ag       |      0 |             2 |    0 |              0 |             0 |                   2 |               0 |
| tg       |      3 |             0 |    0 |              0 |             1 |                   0 |               0 |
| kkp      |      0 |             0 |    0 |              0 |             1 |                   0 |               0 |
+----------+--------+---------------+------+----------------+---------------+---------------------+-----------------+
4 rows in set (0.00 sec)
```

MAIN PROGRAM

C CODE

```c
//HEADER FILES
#include <mysql.h>
#include <stdio.h>
#include <time.h>
#include <string.h>
#include <stdlib.h>

// EXTRA DEFINITIONS
#define WIDTH 10
#define HEIGHT 10

// GAME FUNCTIONS
void Lucky7();
void Scramble_Word(char username[], MYSQL *conn);
void Maze(char username[], MYSQL *conn);
void Cards_Matching(char username[], MYSQL *conn);
int Word_Guessing(char username[], MYSQL *conn);
void Rock_Paper_Scissors(char username[], MYSQL *conn);
void Number_Guessing(char username[], MYSQL *conn);


// EXTRA FUNCTIONS which are used in the above GAMING FUNCTIONS
int moneyChecker(int bet, char username[],MYSQL *conn);
void total(char game[], char username[],MYSQL *conn);
void win(char game[], char username[],MYSQL *conn);
void lose(char game[], char username[],MYSQL *conn);
void shuffle_word(char *word);
void rule_card_matching();
int calculate_score(int chances);
const char* get_card_type(int value);
void display_word(const char *word, const int *revealed);
void generate_maze(char maze[HEIGHT][WIDTH], int x, int y);
void display_maze(char maze[HEIGHT][WIDTH]);

// DEFINING STRUCTURES FOR THE GAME
typedef struct {
    int value;
    int is_matched;
} Card;

void initialize_grid(Card *cards, int size);
void shuffle_cards(Card *cards, int size);
void display_grid(Card *cards, int size);
void display_remaining_choices(Card *cards, int size);
int check_match(Card *cards, int index1, int index2);
void remove_matched_cards(Card *cards, int index1, int index2);
int calculate_correct_choices(Card *cards, int size);


typedef struct {
    int x, y;
```

```
} Position;

int navigate_maze(char maze[HEIGHT][WIDTH], Position *current, char
direction, int steps, Position end);

// FUNTIONS for EXTRA FUNCTIONALITY FOR USER
void T_W_L_Details(char username[],MYSQL *conn);
void Account_Details(char username[],MYSQL *conn);

// MAIN FUNCTION
void main() {
    MYSQL *conn;
    MYSQL_RES *result;
    MYSQL_ROW record;
    char Query[256];
    char *server = "localhost";
    char *user  = "root";
    char *password1 = "Taru@23sep";
    char *database = "Gaming_Nexus";

    conn = mysql_init(NULL);
    if (!mysql_real_connect(conn, server, user, password1, database, 0,
NULL, 0)) {
        fprintf(stderr, "Failed to connect to database: %s\n",
mysql_error(conn));
        return;
    }

    int i;
    for (i=0;i<=117;i++){
        printf("*");
    }
    printf("\n");
    for (i=0;i<=50;i++){
        printf(" ");
    }
    printf("GAMING NEXUS\n");
    for (i=0;i<=117;i++){
        printf("*");
    }
    printf("\n");

    int starting_window;
    char permit = 'N';
    char name[20], username[20], password[20];

    while (1) {
            printf("PRESS 1 FOR CREATING ACCOUNT\n");
            printf("PRESS 2 FOR LOG IN\n");
            printf("Do you want to CREATE ACCOUNT/LOG IN : ");
            scanf("%d", &starting_window);
            getchar(); // Clear newline from input buffer after scanf

            if (starting_window == 1) {
                printf("\nEnter Name : ");
                fgets(name, sizeof(name), stdin);
                name[strcspn(name, "\n")] = '\0';
                int username_exists;
                do {
                    printf("\nEnter Username : ");
                    fgets(username, sizeof(username), stdin);
                    username[strcspn(username, "\n")] = '\0';

                    sprintf(Query, "SELECT username FROM players WHERE
username = '%s'", username);
                    mysql_query(conn, Query);
                    result = mysql_store_result(conn);

                    username_exists = (mysql_num_rows(result) > 0);
                    if (username_exists) {
                        printf("\nTHIS USERNAME ALREADY EXISTS\n");
                    }
                    mysql_free_result(result);
                } while (username_exists);

                printf("\nEnter Password : ");
                fgets(password, sizeof(password), stdin);
                password[strcspn(password, "\n")] = '\0';

                sprintf(Query, "INSERT INTO players VALUES('%s', '%s',
'%s', 500)", name, username, password);
                mysql_query(conn, Query);
                sprintf(Query, "Insert into TGames
values('%s',0,0,0,0,0,0,0)",username);
                mysql_query(conn, Query);
                sprintf(Query, "Insert into WGames
values('%s',0,0,0,0,0,0,0)",username);
                mysql_query(conn, Query);
                sprintf(Query, "Insert into LGames
values('%s',0,0,0,0,0,0,0)",username);
                mysql_query(conn, Query);

                mysql_commit(conn);
                printf("\nACCOUNT CREATED & LOGGED IN\n");
                permit = 'Y';
                break;
            }

            else if (starting_window == 2) {
                int user_exists = 0;

                // Input Username
                while (1) {
                    printf("\nEnter Username : ");
                    fgets(username, sizeof(username), stdin);
                    username[strcspn(username, "\n")] = '\0';

                    // Query to check if the username exists
                    sprintf(Query, "SELECT username FROM players WHERE
username = '%s'", username);
                    if (mysql_query(conn, Query)) {
                        fprintf(stderr, "MySQL Query Error: %s\n",
mysql_error(conn));
                        continue;
                    }
                    result = mysql_store_result(conn);
                    if (result == NULL) {
                        fprintf(stderr, "MySQL Store Result Error: 
%s\n", mysql_error(conn));
                        continue;
                    }

                    // Check if the username is found
                    if (mysql_num_rows(result) == 0) {
                        printf("\nTHIS USERNAME IS NOT REGISTERED\n");
                        mysql_free_result(result);
                        continue;
                    } else {
                        user_exists = 1;
                        mysql_free_result(result);
                        break;
                    }
                }

                // Input Password
                if (user_exists) {
                    int valid_login = 0;

                    while (1) {
                        printf("\nEnter Password : ");
                        fgets(password, sizeof(password), stdin);
                        password[strcspn(password, "\n")] = '\0';

                        // Query to validate password
                        sprintf(Query, "SELECT password FROM players 
WHERE username = '%s'", username);
                        if (mysql_query(conn, Query)) {
                            fprintf(stderr, "MySQL Query Error: %s\n", 
mysql_error(conn));
                            continue;
                        }

                        result = mysql_store_result(conn);
                        if (result == NULL) {
                            fprintf(stderr, "MySQL Store Result Error: 
%s\n", mysql_error(conn));
                            continue;
                        }

                        // Fetch the record and validate the password
                        record = mysql_fetch_row(result);
                        if (record != NULL && strcmp(record[0], 
password) == 0) {
                            printf("\nLOGGED IN SUCCESSFULLY\n");
                            permit = 'Y';
                            valid_login = 1;
                            mysql_free_result(result);
                            break;
                        } else {
                            printf("\nINCORRECT PASSWORD\n");
                            mysql_free_result(result);
                        }
                    }

                    if (!valid_login) {
                        permit = 'N';
                    }
                }

                // Adding Rs. 250 to the user's account whose balance
is 0.
                sprintf(Query,"select balance from players where
username = '%s'",username);
                mysql_query(conn,Query);
                result = mysql_store_result(conn);
                record = mysql_fetch_row(result); //used to be rs
                if (atoi(record[0]) == 0){
                    sprintf(Query,"update players set balance = 250
where username = '%s'",username);
                    mysql_query(conn,Query);
                    printf("\nYOUR ACCOUNT BALANCE is 0.\nRs. 250 ADDED
TO YOUR ACCOUNT\n");
                }

                break;
            }

        else{
                printf("\nKINDLY ENTER A VALID NUMBER\n");
        }
    }

    // PROGRAM AFTER LOGGING IN
    char program_choice='Y';
    char gamezone_choice='Y';
    int program,gamezone;
    while(program_choice=='Y' || program_choice=='y'){
        printf("\n\nPRESS 1 to enter in GAME ZONE\n");
        printf("PRESS 2 to see GAMEZONE PERFORMANCE\n");
        printf("PRESS 3 to see ACCOUNT DETAILS\n");
        printf("ENTER CHOICE : ");
        scanf("%d", &program);
        if (program == 1){
            printf("\n\n******* GAME ZONE *******\n\n");
            while(gamezone_choice=='Y' || gamezone_choice=='y') {
                printf("\n\nPRESS 1 to play LUCKY 7\n");
                printf("PRESS 2 to play SCRAMBLE WORDS\n");
                printf("PRESS 3 to play MAZE GAME\n");
                printf("PRESS 4 to play CARDS MATCHING GAME\n");
                printf("PRESS 5 to play WORD GUESSING GAME\n");
                printf("PRESS 6 to play ROCK, PAPER & SCISSORS\n");
                printf("PRESS 7 to play NUMBER GUESSING GAME\n");
                printf("CHOOSE GAME TO PLAY : ");
                scanf("%d",&gamezone);
                if(gamezone==1)
                {
                    Lucky7(username, conn);
                }
                else if(gamezone==2)
                {
                    Scramble_Word(username, conn);
                }
                else if(gamezone==3)
                {
                    Maze(username, conn);
                }
                else if(gamezone==4)
                {
                    Cards_Matching(username, conn);
                }
                else if(gamezone==5)
                {
                    Word_Guessing(username, conn);
                }
                else if(gamezone==6)
                {
                    Rock_Paper_Scissors(username, conn);
                }
                else if(gamezone==7)
                {
                    Number_Guessing(username, conn);
                }
                else{
                    printf("INVALID CHOICE !!");
                }

                sprintf(Query,"select balance from players where
username = '%s'",username);
                mysql_query(conn,Query);
                result = mysql_store_result(conn);
                record = mysql_fetch_row(result);//used to be rs
                if (atoi(record[0]) == 0){
                    printf("\nYOUR ACCOUNT BALANCE is 0.\n");
                    printf("YOU ARE OUT OF GAME ZONE.\n");
                    printf("To enter game zone again. Log in again. It
will automatically add Rs. 250 to your account.\n");
                    break;
                }
                printf("\nDo you want to continue in the GAME ZONE
(Y/N): ");
                fflush(stdin);
                scanf("%c",&gamezone_choice);
            }
        }
        else if(program == 2){
            T_W_L_Details(username,conn);
        }
        else if(program == 3){
            Account_Details(username,conn);
        }
        else{
            printf("KINDLY ENTER A VALID NUMBER !!\n\n");
        }
        printf("\n\nDO YOU WANT TO CONTINUE THE PROGRAM (Y/N): ");
        fflush(stdin);
        scanf("%c",&program_choice);
    }
    mysql_commit(conn);
    mysql_close(conn);


    printf("\n");
    for (i=0;i<=117;i++){
        printf("*");
    }
    printf("\n");
    for (i=0;i<=44;i++){
        printf(" ");
    }
    printf("YOU HAVE BEEN LOGGED OUT\n");
    for (i=0;i<=44;i++){
        printf(" ");
    }
    printf("THANK YOU FOR CHECKING IN\n");
    for (i=0;i<=117;i++){
        printf("*");
    }
}

// FUNCTIONS are created below

void Lucky7(char username[], MYSQL *conn)
{
    int bet;
    int dice;
    int choice;
    char Query[100];

    srand(time(0));
    int lower = 2, upper = 12;
    printf("WELCOME TO LUCKY 7\n\n");

    int betPermit = 1;
    while (betPermit == 1){
        printf("RULES :\n");
        printf("You have to choose in between LESS THAN 7 , 7 & GREATER
THAN 7.\n");
        printf("Two dices will be rolled.\n");
        printf("If you win then your bet money will be doubled.\n");
        printf("If you lose then you will lose your bet money.\n\n");

        printf("PLACE YOUR BET : ");
        scanf("%d",&bet);
        betPermit = moneyChecker(bet,username,conn);
    }
    printf("ENTER ANY NUMBER ACCORDING TO THE RANGE YOU WANT TO CHOOSE
:");
    scanf("%d",&choice);
    printf("\nDICE ROLLED !!\n");
    dice = (rand()%(upper - lower + 1)) + lower;
    printf("THE DICE SHOWED %d\n",dice);
    if (dice > 7 && choice > 7 || dice == 7 && choice == 7 || dice < 7
&& choice < 7){
        printf("YOU WON !!\n");
        sprintf(Query,"update players set balance = balance + %d where
username = '%s'",bet,username);
```

```c
        mysql_query(conn,Query);
        win("Lucky7",username,conn);
    }
    else {
        printf("YOU LOST !!\n");
        sprintf(Query,"update players set balance = balance - %d where
username = '%s'",bet,username);
        mysql_query(conn,Query);
        lose("Lucky7",username,conn);
    }
    total("Lucky7",username,conn);
    mysql_commit(conn);
}
void Scramble_Word(char username[], MYSQL *conn){
    printf("\n\n        SCRAMBLE WORDS GAME\n\n");
    char Query[100];
    //Taking bet from user
    int bet;
    int betPermit = 1;
    while (betPermit == 1){
        printf("RULES :\n");
        printf("You have to unscramble the given word in 10
attempts.\n");
        printf("If you win then your bet money will be doubled.\n");
        printf("If you lose then you will lose your bet money.\n");
        printf("HINT : The words are related to computer science
only.\n\n");

        printf("PLACE YOUR BET : ");
        scanf("%d",&bet);
        betPermit = moneyChecker(bet,username,conn);
    }
    // List of words for the game
    char *words[] = {"programming", "computer", "scramble", "game",
"developer","software","project","algorithm","compilation","iteration"}
;
    int num_words = sizeof(words) / sizeof(words[0]);
    // Random number generator
    srand(time(0));

    // Selecting a random word from the list
    char original_word[50];
    strcpy(original_word, words[rand() % num_words]);

    // Creating a scrambled version of the word
    char scrambled_word[50];
    strcpy(scrambled_word, original_word);
    shuffle_word(scrambled_word);

    // displaying random word
    char guess[50];
    printf("UNSCRAMBLE THE WORD : %s\n", scrambled_word);
    int turns=0;
    //maximum number of turns is 10
    while(turns<10)
    {
        //prompting the user to guess the word
        printf("YOUR GUESS (%d): ",turns+1);
        scanf("%s", guess);

        //checking if the guessed word is correct
        if (strcmp(guess, original_word) == 0)
        {
            printf("CONGRATULATIONS !! YOU GUESSED IT RIGHT !!\n");
            sprintf(Query,"update players set balance = balance + %d
where username = '%s'",bet,username);
            mysql_query(conn,Query);
            win("Scramble_Word",username,conn);
            break;
        }
        else
        {   if (turns==9)
            {
                printf("INCORRECT! \n");
                turns++;
            }
            else
            {
                printf("Incorrect! Try again !\n");
                turns++;
            }
        }
    }
    //when maximum number of turns to guess the word has exceeded
printing the original word
    if(turns==10)
    {
        printf("YOU LOST !!\n");
        printf("Your number of chances to guess have finsihed.\n ");
        printf("The unscrambled word is %s.", original_word);
        sprintf(Query,"update players set balance = balance - %d where
username = '%s'",bet,username);
        mysql_query(conn,Query);
        lose("Scramble_Word",username,conn);
    }
    total("Scramble_Word",username,conn);
    mysql_commit(conn);
}

void Maze(char username[], MYSQL *conn) {
    printf("\n\n        MAZE GAME\n\n");
    char Query[100];
    //Taking bet from user
    int bet;
    int betPermit = 1;
    while (betPermit == 1){
        printf("RULES :\n");
        printf("S means Starting Point.\n");
        printf("P means Present Position.\n");
        printf("E means Exit Point.\n");
        printf("You have to enter the direction and the number of
steps.\n");
        printf("You don't have any chance to lose this game\n.");
        printf("On winning, your bet money will be doubled.\n\n");

        printf("PLACE YOUR BET : ");
        scanf("%d",&bet);
        betPermit = moneyChecker(bet,username,conn);
    }

    char maze[HEIGHT][WIDTH];
    Position start = {1, 1};
    Position end;
    Position current = start;
    char direction;
    int steps;

    // Initialize the maze with walls
    for (int i = 0; i < HEIGHT; i++) {
        for (int j = 0; j < WIDTH; j++) {
            maze[i][j] = '#';
        }
    }

    // Generate the maze
    srand(time(NULL));
    generate_maze(maze, start.x, start.y);

    // Find a valid end position (an open space)
    for (int i = HEIGHT - 2; i > 0; i--) {
        for (int j = WIDTH - 2; j > 0; j--) {
            if (maze[i][j] == ' ') {
                end.x = i;
                end.y = j;
                break;
            }
        }
        if (end.x != 0 && end.y != 0) break; // Break the outer loop if
a position is found
    }

    // Set start and end positions
    maze[start.x][start.y] = 'S'; // Start
    maze[end.x][end.y] = 'E';     // End

    // Display the generated maze
    display_maze(maze);

    while (1) {
        printf("Enter direction (r - right, l - left, f - front, b -
back) and steps: ");
        scanf(" %c %d", &direction, &steps);

        if (navigate_maze(maze, &current, direction, steps, end)) {
            printf("\nYou have reached the exit!\n");
            sprintf(Query,"update players set balance = balance + %d
where username = '%s'",bet,username);
            mysql_query(conn,Query);
            win("Maze",username,conn);
            total("Maze",username,conn);
            mysql_commit(conn);
            break;
        } else {
            printf("Current position: (%d, %d)\n", current.x,
current.y);
            display_maze(maze);
        }
    }
}

void Cards_Matching(char username[], MYSQL *conn) {
    int a, b, chances = 0, size = 16;
    char choice;
    Card cards[16];
    rule_card_matching();
    initialize_grid(cards, size);
    char Query[100];
    //Taking bet from user
    int bet;
    int betPermit = 1;
    while (betPermit == 1){
        printf("PLACE YOUR BET : ");
        scanf("%d",&bet);
        betPermit = moneyChecker(bet,username,conn);
    }
    int correct_choices;
    shuffle_cards(cards, size);
    while (1)
        {
        display_grid(cards, size);
        printf("\nEnter the card numbers: ");
        scanf("%d%d", &a, &b);
        // Validate user input
        if (a < 1 || a > size || b < 1 || b > size || a == b || cards[a
- 1].is_matched || cards[b - 1].is_matched) {
            printf("\nInvalid input. Please try again.\n");
            continue;
        }
        printf("Card 1: %s\n", get_card_type(cards[a - 1].value));
        printf("Card 2: %s\n", get_card_type(cards[b - 1].value));
        chances++;
        if (check_match(cards, a - 1, b - 1)) {
            printf("\nIt's a match!\n");
            remove_matched_cards(cards, a - 1, b - 1);
        } else {
            printf("\nNo match. Try again.\n");
        }
        // Check if the game is over
        int matches = 0;
        for (int i = 0; i < size; i++) {
            if (cards[i].is_matched) {
                matches++;
            }
        }
        if (matches == size) {
            int score = calculate_score(chances);
            int correct_choices = calculate_correct_choices(cards,
size);
            printf("\nCongratulations! You've matched all the cards in
%d chances.\n", chances);
            printf("Your score: %d\n", score);
            printf("Total correct choices: %d\n", correct_choices);
            break;
        }

        // Ask user if they want to exit the game
        printf("Do you want to exit the game? (y/n): ");
        scanf(" %c", &choice);
        if (choice == 'y' || choice == 'Y') {
            int score = calculate_score(chances);
            correct_choices = calculate_correct_choices(cards, size);
            printf("You have exited the game.\n");
            printf("Total correct choices: %d\n", correct_choices);
            break;
        }
        // Display remaining choices
        display_remaining_choices(cards, size);
    }
    sprintf(Query,"update players set balance = balance + %d where
username = '%s'",(correct_choices/8)*bet,username);
    mysql_query(conn,Query);
    win("Cards_Matching",username,conn);
    total("Cards_Matching",username,conn);
    mysql_commit(conn);
}

int Word_Guessing(char username[], MYSQL *conn) {
    printf("\n\n        WORD GUESSING GAME\n\n");
    char Query[100];
    //Taking bet from user
    int bet;
    int betPermit = 1;
    while (betPermit == 1){
        printf("RULES :\n");
        printf("You have to input a letter everytime to guess a
word.\n");
        printf("You get 10 guesses.\n");
        printf("If you win then your bet money will be doubled.\n");
        printf("If you lose then you will lose your bet money.\n\n");

        printf("PLACE YOUR BET : ");
        scanf("%d",&bet);
        betPermit = moneyChecker(bet,username,conn);
    }
    // List of words for the game
    char *words[] = {"programming", "computer", "game", "developer",
"software", "project", "algorithm", "compilation", "iteration"};
    int num_words = sizeof(words) / sizeof(words[0]);

    // Random number generator
    srand(time(0));

    // Selecting a random word from the list
    char original_word[50];
    strcpy(original_word, words[rand() % num_words]);

    // Create a mask for revealed letters
    int word_length = strlen(original_word);
    int revealed[word_length];
    for (int i = 0; i < word_length; i++) {
        revealed[i] = 0; // Initially, no letters are revealed
    }

    printf("Guess the word by guessing one letter at a time!\n");

    int turns = 0;
    int max_turns = 10;
    char guess;

    while (turns < max_turns) {
        printf("\nYour current word: ");
        display_word(original_word, revealed);

        // Prompt the player for a guess
        printf("Enter your guess (%d): ",turns+1);
        scanf(" %c", &guess);

        // Check if the guessed letter is in the word
        int correct = 0;
        for (int i = 0; i < word_length; i++) {
            if (original_word[i] == guess && !revealed[i]) {
                revealed[i] = 1;
                correct = 1;
            }
        }

        if (correct) {
            printf("Good guess!\n");
        } else {
            printf("Incorrect! Try again. Remaining attempts: %d\n",
max_turns - turns - 1);
            turns++;
        }

        // Check if the entire word has been revealed
        int all_revealed = 1;
        for (int i = 0; i < word_length; i++) {
            if (!revealed[i]) {
                all_revealed = 0;
                break;
            }
        }

        if (all_revealed) {
            printf("\nCongratulations! You guessed the word: %s\n",
original_word);
            sprintf(Query,"update players set balance = balance + %d
where username = '%s'",bet,username);
            mysql_query(conn,Query);
            win("Word_Guessing",username,conn);
            total("Word_Guessing",username,conn);
            mysql_commit(conn);
            return 0;
        }
    }

    // Game over, reveal the word
    printf("\nGame over! You've used all your attempts.\n");
    printf("The word was: %s\n", original_word);
    sprintf(Query,"update players set balance = balance - %d where
username = '%s'",bet,username);
    mysql_query(conn,Query);
    lose("Word_Guessing",username,conn);
    total("Word_Guessing",username,conn);
    mysql_commit(conn);
    return 0;
}

void Rock_Paper_Scissors(char username[], MYSQL *conn) {
    printf("\n\n        ROCK, PAPER, SCISSORS GAME\n\n");
    char Query[100];
    //Taking bet from user
    int bet;
    int betPermit = 1;
    while (betPermit == 1){
        printf("RULES :\n");
        printf("You Have To Choose from Rock, Paper & Scissors.\n");
        printf("The Computer will also choose from Rock, Paper &
Scissors.\n");
        printf("If you win then your bet money will be doubled.\n");
        printf("If you lose then you will lose your bet money.\n\n");

        printf("PLACE YOUR BET : ");
        scanf("%d",&bet);
        betPermit = moneyChecker(bet,username,conn);
    }
    char user_choice;
    int computer_choice;

    // random number generator
    srand(time(0));

    // prompt the user for their choice
    printf("Enter your choice (R for Rock, P for Paper, S for
Scissors): ");
    scanf(" %c", &user_choice);
    if (user_choice == 'R'||user_choice == 'P'||user_choice == 'S')
    {
         // To generate a random choice for the computer (0 for Rock, 1
for Paper, 2 for Scissors)
        computer_choice = rand() % 3;

        // Displaying the computer's choice

        if (computer_choice == 0)
        {
            printf("Computer chose Rock\n");
        }
        else if (computer_choice == 1)
        {
            printf("Computer chose Paper\n");
        }
        else
        {
            printf("Computer chose Scissors\n");
        }

        // Determining the winner

        // when user chooses Rock

        if (user_choice == 'R')
        {
            if (computer_choice == 0)
            {
                printf("It's a tie!\n");
            }
            else if (computer_choice == 1)
            {
                printf("You lose! Paper beats Rock\n");
                sprintf(Query,"update players set balance = balance -
%d where username = '%s'",bet,username);
                mysql_query(conn,Query);
                lose("Rock_Paper_Scissors",username,conn);
            }
            else
            {
                printf("You win! Rock beats Scissors\n");
                sprintf(Query,"update players set balance = balance +
%d where username = '%s'",bet,username);
                mysql_query(conn,Query);
                win("Rock_Paper_Scissors",username,conn);
            }
        }

        // when user chooses Paper

        else if (user_choice == 'P')
        {
            if (computer_choice == 0)
            {
                printf("You win! Paper beats Rock\n");
                sprintf(Query,"update players set balance = balance +
%d where username = '%s'",bet,username);
                mysql_query(conn,Query);
                win("Rock_Paper_Scissors",username,conn);
            }
            else if (computer_choice == 1)
            {
                printf("It's a tie!\n");
            }
            else
            {
                printf("You lose! Scissors beat Paper\n");
                sprintf(Query,"update players set balance = balance -
%d where username = '%s'",bet,username);
                mysql_query(conn,Query);
                lose("Rock_Paper_Scissors",username,conn);
            }
        }
        // when user chooses Scissors

        else if (user_choice == 'S')
        {
            if (computer_choice == 0)
            {
                printf("You lose! Rock beats Scissors\n");
                sprintf(Query,"update players set balance = balance -
%d where username = '%s'",bet,username);
                mysql_query(conn,Query);
                lose("Rock_Paper_Scissors",username,conn);
            }
            else if (computer_choice == 1)
            {
                printf("You win! Scissors beat Paper\n");
                sprintf(Query,"update players set balance = balance +
%d where username = '%s'",bet,username);
                mysql_query(conn,Query);
                win("Rock_Paper_Scissors",username,conn);
            }
            else
            {
                printf("It's a tie!\n");
            }
        }
        total("Rock_Paper_Scissors",username,conn);
        mysql_commit(conn);
    }
    // For invalid input by user
    else
    {
        printf("INVALID INPUT !! Please enter 'R', 'P', or 'S'.\n");
    }
}
void Number_Guessing(char username[], MYSQL *conn){
    printf("\n\n        NUMBER GUESSING GAME\n\n");
    char Query[100];
    //Taking bet from user
    int bet;
    int betPermit = 1;
    while (betPermit == 1){
        printf("RULES :\n");
        printf("You have to guess the number between 0 to 50.\n");
        printf("You get 5 guesses.\n");
        printf("If you win then your bet money will be doubled.\n");
        printf("If you lose then you will lose your bet money.\n\n");

        printf("PLACE YOUR BET : ");
        scanf("%d",&bet);
        betPermit = moneyChecker(bet,username,conn);
    }

    //generating number between 1 and 50
    srand(time(0));
    int r= (rand() % (50))+ 1;

    int b=0;
    int i;

    //loop for bet exhaustion
    for(i=0; i<5; i++)
    {
        printf("ENTER YOUR GUESS (%d): ",i+1);
        scanf("%d", &b);

        if(r>b)
        {
            if((r-b)<=5)
            {
                if(i==4)
                {
                    printf("\nWRONG GUESS !! YOU LOST !!\n");
                    printf("THE NUMBER WAS %d", r);
                    sprintf(Query,"update players set balance = balance
- %d where username = '%s'",bet,username);
                    mysql_query(conn,Query);
                    lose("Number_Guessing",username,conn);
                }
                else
                {
                    printf("\nWRONG GUESS, BUT VERY CLOSE \n");
                    printf("Guess again \n\n");
                    continue;

                }
            }

            else if(((r-b)>5 && (r-b)<=10))
            {
                if(i==4)
                {
                    printf("\nWRONG GUESS !! YOU LOST !!\n");
                    printf("THE NUMBER WAS %d", r);
                    sprintf(Query,"update players set balance = balance
- %d where username = '%s'",bet,username);
                    mysql_query(conn,Query);
                    lose("Number_Guessing",username,conn);
                }
                else
                {
                    printf("\nWRONG GUESS BUT CLOSE \n");
                    printf("Choose a number higher \n");
                    printf("Guess again \n\n");
                    continue;
                }
            }
            else if((r-b)>10)
            {
                if(i==4)
                {
                    printf("\nWRONG GUESS !! YOU LOST !!\n");
                    printf("THE NUMBER WAS %d", r);
                    sprintf(Query,"update players set balance = balance
- %d where username = '%s'",bet,username);
                    mysql_query(conn,Query);
                    lose("Number_Guessing",username,conn);
                }
                else
                {
                    printf("\nWRONG GUESS \n");
                    printf("Choose a number higher \n");
                    printf("Guess again \n\n");
                    continue;
                }
            }
        }
        else if(r<b)
        {
            if((b-r)<=5)
            {
                if(i==4)
                {
                    printf("\nWRONG GUESS !! YOU LOST !!\n");
                    printf("THE NUMBER WAS %d", r);
                    sprintf(Query,"update players set balance = balance
- %d where username = '%s'",bet,username);
                    mysql_query(conn,Query);
                    lose("Number_Guessing",username,conn);
                }
                else
                {
                    printf("WRONG GUESS BUT CLOSE \n");
                    printf("Choose a number lower \n");
                    printf("Guess again \n\n");
                    continue;
                }
            }
            else if ((b-r)>5 && (b-r)<=10)
            {
                if(i==4)
                {
                    printf("\nWRONG GUESS !! YOU LOST !!\n");
                    printf("THE NUMBER WAS %d", r);
                    sprintf(Query,"update players set balance = balance
- %d where username = '%s'",bet,username);
                    mysql_query(conn,Query);
                    lose("Number_Guessing",username,conn);
                }
                else
                {
                    printf("WRONG GUESS BUT CLOSE \n");
                    printf("Choose a number lower \n");
                    printf("Guess again \n\n");
                    continue;
                }
            }

            else if((b-r)>10)
            {
                if(i==4)
                {
                    printf("\nWRONG GUESS !! YOU LOST !!\n");
                    printf("THE NUMBER WAS %d", r);
                    sprintf(Query,"update players set balance = balance
- %d where username = '%s'",bet,username);
                    mysql_query(conn,Query);
                    lose("Number_Guessing",username,conn);
                }
                else{
                    printf("\nWRONG GUESS \n");
                    printf("Choose a number lower \n");
                    printf("Guess again \n\n");
                    continue;
                }
            }
        }
        else
        {
            printf("\nCORRECT GUESS\n");
            printf("YOU WON !!");
            sprintf(Query,"update players set balance = balance + %d
where username = '%s'",bet,username);
            mysql_query(conn,Query);
            win("Number_Guessing",username,conn);
        }
    }
    total("Number_Guessing",username,conn);
    mysql_commit(conn);
}
int moneyChecker(int bet, char username[],MYSQL *conn){
    char Query[100];
    sprintf(Query,"select balance from players where username =
'%s'",username);
    mysql_query(conn,Query);
    MYSQL_RES *rs = mysql_store_result(conn);
    MYSQL_ROW record = mysql_fetch_row(rs);
    if (atoi(record[0])<bet){
        printf("\nAccount Balance Exceeded !! Place your bet again
!!\n");
        printf("Account Balance : %d\n\n",atoi(record[0]));
        return 1;
    }
    return 0;
}
void total(char game[], char username[],MYSQL *conn){
    char Query[100];
    sprintf(Query,"update TGames set %s = %s + 1 where username =
'%s'",game,game,username);
    mysql_query(conn,Query);
}
void win(char game[], char username[],MYSQL *conn){
    char Query[100];
    sprintf(Query,"update WGames set %s = %s + 1 where username =
'%s'",game,game,username);
    mysql_query(conn,Query);
}
void lose(char game[], char username[],MYSQL *conn)
{
    char Query[100];
    sprintf(Query,"update LGames set %s = %s + 1 where username =
'%s'",game,game,username);
    mysql_query(conn,Query);
}
void shuffle_word(char *word) {
    int len = strlen(word);
    for (int i = 0; i < len; i++) {
        int random_index = rand() % len;
        char temp = word[i];
        word[i] = word[random_index];
        word[random_index] = temp;
    }
}
// Function to display the rules of the game CARDS MATCHING
void rule_card_matching() {
    printf("\t\t\t*WELCOME TO CARD MATCHING GAME*\n");
    printf("RULES:\n\tYou have to select two cards out of 16
(represented by numbers in a grid)");
    printf("\n\tIf the cards are matched, they are removed from the
grid.");
    printf("\n\tElse, they are kept back, and the player is given
another chance.");
    printf("\n\tThere are 8 pairs, the more pairs you match, the more
money you will get.");
    printf("\n\tFOR EXAMPLE: If you match 2 pairs then your balance
will be incremented by 25 percent of your bet.");
    printf("\n\tYou can exit the game anytime.");
    printf("\n\tThere is no chance of losing money in this game.");
    printf("\n\n\t\t\t*ALL THE BEST*\n\n");
    printf("\tGrid:\n");
    printf("\t\t| 1  | 2  | 3  | 4  |\n\t\t| 5  | 6  | 7  | 8  |\n\t\t|
9  | 10 | 11 | 12 |\n\t\t| 13 | 14 | 15 | 16 |\n\n");
}
// Function to get the card type as a string
const char* get_card_type(int value) {
    switch (value) {
        case 1: return "Red King";
        case 2: return "Red Queen";
        case 3: return "Red Jack";
        case 4: return "Red Ace";
        case 5: return "Black King";
        case 6: return "Black Queen";
        case 7: return "Black Jack";
        case 8: return "Black Ace";
        default: return "Unknown";
    }
}
// Function to initialize the grid with card values
void initialize_grid(Card *cards, int size) {
    int values[] = {1, 2, 3, 4, 1, 2, 3, 4, 5, 6, 7, 8, 5, 6, 7, 8};
    for (int i = 0; i < size; i++) {
        cards[i].value = values[i];
        cards[i].is_matched = 0;
    }
}
// Function to shuffle the cards
void shuffle_cards(Card *cards, int size) {
    srand(time(0));
    for (int i = 0; i < size; i++) {
        int r = rand() % size;
        Card temp = cards[i];
        cards[i] = cards[r];
        cards[r] = temp;
    }
```

```
}
// Function to display the grid
void display_grid(Card *cards, int size) {
    for (int i = 0; i < size; i++) {
        if (cards[i].is_matched) {
            printf(" XX ");
        } else {
            printf(" %2d ", i + 1);
        }
        if ((i + 1) % 4 == 0) {
            printf("\n");
        }
    }
    printf("\n");
}
// Function to display remaining choices
void display_remaining_choices(Card *cards, int size) {
    printf("\nRemaining choices are: \n");
    for (int i = 0; i < size; i++) {
        if (!cards[i].is_matched) {
            printf("%2d ", i + 1);
        } else {
            printf("XX ");
        }
        if ((i + 1) % 4 == 0) {
            printf("\n");
        }
    }
}
// Function to check if two selected cards match
int check_match(Card *cards, int index1, int index2) {
    return cards[index1].value == cards[index2].value;
}
// Function to remove matched cards from the grid
void remove_matched_cards(Card *cards, int index1, int index2) {
    cards[index1].is_matched = 1;
    cards[index2].is_matched = 1;
}
// Function to calculate the score based on the number of chances taken
int calculate_score(int chances) {
    return 100 - (chances - 1) * 5; // Example scoring formula
}
// Function to calculate the total number of correct choices entered by
the user
int calculate_correct_choices(Card *cards, int size) {
    int correct_choices = 0;
    for (int i = 0; i < size; i++) {
        if (cards[i].is_matched) {
            correct_choices++;
        }
    }
    return correct_choices / 2; // Since each match involves two
choices
}
// Function to display the current state of the word
void display_word(const char *word, const int *revealed) {
    for (int i = 0; word[i] != '\0'; i++) {
        if (revealed[i]) {
            printf("%c", word[i]);
        } else {
            printf("*");
        }
    }
    printf("\n");
}
// Recursive function to generate the maze
void generate_maze(char maze[HEIGHT][WIDTH], int x, int y) {
    int directions[4][2] = {
        {0, 1},   // Right
        {0, -1},  // Left
        {1, 0},   // Down
        {-1, 0}   // Up
    };

    // Shuffle directions to ensure randomness
    for (int i = 0; i < 4; i++) {
        int rand_idx = rand() % 4;
        int temp[2] = {directions[i][0], directions[i][1]};
        directions[i][0] = directions[rand_idx][0];
        directions[i][1] = directions[rand_idx][1];
        directions[rand_idx][0] = temp[0];
        directions[rand_idx][1] = temp[1];
    }

    for (int i = 0; i < 4; i++) {
        int new_x = x + directions[i][0] * 2;
        int new_y = y + directions[i][1] * 2;

        if (new_x > 0 && new_x < HEIGHT - 1 && new_y > 0 && new_y <
WIDTH - 1 && maze[new_x][new_y] == '#') {
            maze[new_x][new_y] = ' ';
            maze[x + directions[i][0]][y + directions[i][1]] = ' ';
            generate_maze(maze, new_x, new_y);
        }
    }
}
// Function to display the maze
void display_maze(char maze[HEIGHT][WIDTH]) {
    for (int i = 0; i < HEIGHT; i++) {
        for (int j = 0; j < WIDTH; j++) {
            printf("%c ", maze[i][j]);
        }
        printf("\n");
    }
}

// Function to navigate through the maze based on user input
int navigate_maze(char maze[HEIGHT][WIDTH], Position *current, char
direction, int steps, Position end) {
    int new_x = current->x;
    int new_y = current->y;

    for (int i = 0; i < steps; i++) {
        if (direction == 'r') {
            new_y++;
        } else if (direction == 'l') {
            new_y--;
        } else if (direction == 'f') {
            new_x++;
        } else if (direction == 'b') {
            new_x--;
        }

        if (new_x <= 0 || new_x >= HEIGHT - 1 || new_y <= 0 || new_y >= 
WIDTH - 1 || maze[new_x][new_y] == '#') {
            printf("No space. Try again.\n");
            return 0;
        }

        // Check if we've reached the end before updating the maze
        if (new_x == end.x && new_y == end.y) {
            maze[current->x][current->y] = ' ';
            current->x = new_x;
            current->y = new_y;
            maze[current->x][current->y] = 'E';
            return 1;
        }
    }

    // Update the maze with the new position
    maze[current->x][current->y] = ' ';
    current->x = new_x;
    current->y = new_y;
    maze[current->x][current->y] = 'P';

    return 0;
}
void T_W_L_Details(char username[],MYSQL *conn){
    char Query[100];
    sprintf(Query,"SELECT Lucky7, Scramble_Word, Maze, Cards_Matching, 
Word_Guessing, Rock_Paper_Scissors, Number_Guessing from TGames where 
username = '%s'",username);
    mysql_query(conn,Query);
    MYSQL_RES *rs = mysql_store_result(conn);
    MYSQL_ROW record = mysql_fetch_row(rs);
    printf("\n\nTOTAL GAMES :");
    printf("\nLucky7  |  Scramble_Word  |  Maze  |  Cards_Matching  | 
Word_Guessing  |  Rock_Paper_Scissors  |  Number_Guessing\n");
    printf("%s       |  %s              |  %s     |  %s                | 
%s              |  %s                     | 
%s\n",record[0],record[1],record[2],record[3],record[4],record[5],recor
d[6]);

    sprintf(Query,"SELECT Lucky7, Scramble_Word, Maze, Cards_Matching, 
Word_Guessing, Rock_Paper_Scissors, Number_Guessing from WGames where 
username = '%s'",username);
    mysql_query(conn,Query);
    rs = mysql_store_result(conn);
    record = mysql_fetch_row(rs);
    printf("\nGAMES WON :");
    printf("\nLucky7  |  Scramble_Word  |  Maze  |  Cards_Matching  | 
Word_Guessing  |  Rock_Paper_Scissors  |  Number_Guessing\n");
    printf("%s       |  %s              |  %s     |  %s                | 
%s              |  %s                     | 
%s\n",record[0],record[1],record[2],record[3],record[4],record[5],recor
d[6]);

    sprintf(Query,"SELECT Lucky7, Scramble_Word, Maze, Cards_Matching,
Word_Guessing, Rock_Paper_Scissors, Number_Guessing from LGames where
username = '%s'",username);
    mysql_query(conn,Query);
    rs = mysql_store_result(conn);
    record = mysql_fetch_row(rs);
    printf("\nGAMES LOST :");
    printf("\nLucky7  |  Scramble_Word  |  Maze  |  Cards_Matching  |
Word_Guessing  |  Rock_Paper_Scissors  |  Number_Guessing\n");
    printf("%s        |  %s              |  %s     |  %s               |
%s              |  %s                    |
%s\n\n",record[0],record[1],record[2],record[3],record[4],record[5],rec
ord[6]);
}
void Account_Details(char username[],MYSQL *conn){
    char Query[100];
    sprintf(Query,"select * from players where username =
'%s'",username);
    mysql_query(conn,Query);
    MYSQL_RES *rs = mysql_store_result(conn);
    MYSQL_ROW record = mysql_fetch_row(rs);
    printf("NAME  |  USERNAME  |  PASSWORD  |  BALANCE\n");
    printf("%s   |   %s     |     %s     |
%s",record[0],record[1],record[2],record[3]);
}
```

OUTPUT

Account Creation

```
********************************************************************************

                              GAMING NEXUS

********************************************************************************

PRESS 1 FOR CREATING ACCOUNT

PRESS 2 FOR LOG IN

Do you want to CREATE ACCOUNT/LOG IN : 1

Enter Name : Anjaan

Enter Username : Any

Enter Password : ghost

ACCOUNT CREATED & LOGGED IN
```

USER LOGIN

```
********************************************************************************

                              GAMING NEXUS

********************************************************************************


PRESS 1 FOR CREATING ACCOUNT
PRESS 2 FOR LOG IN
Do you want to CREATE ACCOUNT/LOG IN : 2

Enter Username : Any
Enter Password : ghost

LOGGED IN SUCCESSFULLY

PRESS 1 to enter in GAME ZONE
PRESS 2 to see GAMEZONE PERFORMANCE
PRESS 3 to see ACCOUNT DETAILS
ENTER CHOICE : 1
```

```
******* GAME ZONE *******

PRESS 1 to play LUCKY 7
PRESS 2 to play SCRAMBLE WORDS
PRESS 3 to play MAZE GAME
PRESS 4 to play CARDS MATCHING GAME
PRESS 5 to play WORD GUESSING GAME
PRESS 6 to play ROCK, PAPER & SCISSORS
PRESS 7 to play NUMBER GUESSING GAME
CHOOSE GAME TO PLAY : 1
WELCOME TO LUCKY 7

RULES :
You have to choose in between LESS THAN 7 , 7 & GREATER THAN 7.
Two dices will be rolled.
If you win then your bet money will be doubled.
If you lose then you will lose your bet money.

PLACE YOUR BET : 20
ENTER ANY NUMBER ACCORDING TO THE RANGE YOU WANT TO CHOOSE :3

DICE ROLLED !!
THE DICE SHOWED 9
YOU LOST !!

Do you want to continue in the GAME ZONE (Y/N): Y

PRESS 1 to play LUCKY 7
PRESS 2 to play SCRAMBLE WORDS
PRESS 3 to play MAZE GAME
PRESS 4 to play CARDS MATCHING GAME
PRESS 5 to play WORD GUESSING GAME
PRESS 6 to play ROCK, PAPER & SCISSORS
PRESS 7 to play NUMBER GUESSING GAME
CHOOSE GAME TO PLAY : 2


     SCRAMBLE WORDS GAME

RULES :
You have to unscramble the given word in 10 attempts.
If you win then your bet money will be doubled.
If you lose then you will lose your bet money.
HINT : The words are related to computer science only.

PLACE YOUR BET : 20
UNSCRAMBLE THE WORD : eanttioir
YOUR GUESS (1): tea
Incorrect! Try again !
```

```
YOUR GUESS (2): iteration
CONGRATULATIONS !! YOU GUESSED IT RIGHT !!

Do you want to continue in the GAME ZONE (Y/N): Y


PRESS 1 to play LUCKY 7
PRESS 2 to play SCRAMBLE WORDS
PRESS 3 to play MAZE GAME
PRESS 4 to play CARDS MATCHING GAME
PRESS 5 to play WORD GUESSING GAME
PRESS 6 to play ROCK, PAPER & SCISSORS
PRESS 7 to play NUMBER GUESSING GAME
CHOOSE GAME TO PLAY : 3


        MAZE GAME

RULES :
S means Starting Point.
P means Present Position.
E means Exit Point.
You have to enter the direction and the number of steps.
You don't have any chance to lose this game
.On winning, your bet money will be doubled.

PLACE YOUR BET : 20
# # # # # # # # # #
# S             # #
#   # # # # #   # #
#       #       # #
#   #   #   # # # #
#   #   #       # #
#   # # # # #   # #
#             E # #
# # # # # # # # # #
# # # # # # # # # #
Enter direction (r - right, l - left, f - front, b - back) and steps: f6
Current position: (7, 1)
# # # # # # # # # #
#               # #
#   # # # # #   # #
#       #       # #
#   #   #   # # # #
#   #   #       # #
#   # # # # #   # #
# P           E # #
# # # # # # # # # #
# # # # # # # # # #
Enter direction (r - right, l - left, f - front, b - back) and steps: r6
```

```
You have reached the exit!

Do you want to continue in the GAME ZONE (Y/N): Y


PRESS 1 to play LUCKY 7
PRESS 2 to play SCRAMBLE WORDS
PRESS 3 to play MAZE GAME
PRESS 4 to play CARDS MATCHING GAME
PRESS 5 to play WORD GUESSING GAME
PRESS 6 to play ROCK, PAPER & SCISSORS
PRESS 7 to play NUMBER GUESSING GAME
CHOOSE GAME TO PLAY : 4
                *WELCOME TO CARD MATCHING GAME*
RULES:
    You have to select two cards out of 16 (represented by numbers in a grid)
    If the cards are matched, they are removed from the grid.
    Else, they are kept back, and the player is given another chance.
    There are 8 pairs, the more pairs you match, the more money you will get.
    FOR EXAMPLE: If you match 2 pairs then your balance will be incremented by 25 percent of
your bet.
    You can exit the game anytime.
    There is no chance of losing money in this game.

                *ALL THE BEST*

    Grid:
        | 1  | 2  | 3  | 4  |
        | 5  | 6  | 7  | 8  |
        | 9  | 10 | 11 | 12 |
        | 13 | 14 | 15 | 16 |

PLACE YOUR BET : 20
  1   2   3   4
  5   6   7   8
  9  10  11  12
 13  14  15  16



Enter the card numbers: 1
14
Card 1: Red Queen
Card 2: Black Ace

No match. Try again.
Do you want to exit the game? (y/n): n

Remaining choices are:
 1  2  3  4
 5  6  7  8
 9 10 11 12
13 14 15 16
  1   2   3   4
  5   6   7   8
  9  10  11  12
 13  14  15  16


Enter the card numbers: 5
12
Card 1: Red King
Card 2: Black Queen

No match. Try again.
Do you want to exit the game? (y/n): n

Remaining choices are:
 1  2  3  4
 5  6  7  8
 9 10 11 12
13 14 15 16
  1   2   3   4
  5   6   7   8
  9  10  11  12
 13  14  15  16


Enter the card numbers: 14
4
Card 1: Black Ace
Card 2: Red Queen

No match. Try again.
Do you want to exit the game? (y/n): n

Remaining choices are:
 1  2  3  4
 5  6  7  8
 9 10 11 12
13 14 15 16
  1   2   3   4
  5   6   7   8
  9  10  11  12
 13  14  15  16


Enter the card numbers: 1
4
Card 1: Red Queen
Card 2: Red Queen

It's a match!
Do you want to exit the game? (y/n): n

Remaining choices are:
XX  2  3 XX
 5  6  7  8
 9 10 11 12
13 14 15 16
 XX   2   3  XX
  5   6   7   8
  9  10  11  12
 13  14  15  16



Enter the card numbers: 10
16
Card 1: Black Jack
Card 2: Red King

No match. Try again.
Do you want to exit the game? (y/n): n

Remaining choices are:
XX  2  3 XX
 5  6  7  8
 9 10 11 12
13 14 15 16
 XX   2   3  XX
  5   6   7   8
  9  10  11  12
 13  14  15  16



Enter the card numbers: 16
10
Card 1: Red King
Card 2: Black Jack

No match. Try again.
Do you want to exit the game? (y/n): y
You have exited the game.
Total correct choices: 1

Do you want to continue in the GAME ZONE (Y/N): Y


PRESS 1 to play LUCKY 7
PRESS 2 to play SCRAMBLE WORDS
PRESS 3 to play MAZE GAME
PRESS 4 to play CARDS MATCHING GAME
PRESS 5 to play WORD GUESSING GAME
PRESS 6 to play ROCK, PAPER & SCISSORS
PRESS 7 to play NUMBER GUESSING GAME
CHOOSE GAME TO PLAY : 5


        WORD GUESSING GAME

RULES :
You have to input a letter everytime to guess a word.
You get 10 guesses.
If you win then your bet money will be doubled.
If you lose then you will lose your bet money.

PLACE YOUR BET : 20
Guess the word by guessing one letter at a time!

Your current word: *********
Enter your guess (1): a
Good guess!

Your current word: a********
Enter your guess (1): e
Incorrect! Try again. Remaining attempts: 9

Your current word: a********
Enter your guess (2): algorithm
Incorrect! Try again. Remaining attempts: 8

Your current word: a********
Enter your guess (3): Good guess!

Your current word: al*******
Enter your guess (3): Good guess!

Your current word: alg******
Enter your guess (3): Good guess!

Your current word: algo*****
Enter your guess (3): Good guess!

Your current word: algor****
Enter your guess (3): Good guess!

Your current word: algori***
Enter your guess (3): Good guess!

Your current word: algorit**
```

```
Enter your guess (3): Good guess!

Your current word: algorith*
Enter your guess (3): Good guess!

Congratulations! You guessed the word: algorithm

Do you want to continue in the GAME ZONE (Y/N): Y


PRESS 1 to play LUCKY 7
PRESS 2 to play SCRAMBLE WORDS
PRESS 3 to play MAZE GAME
PRESS 4 to play CARDS MATCHING GAME
PRESS 5 to play WORD GUESSING GAME
PRESS 6 to play ROCK, PAPER & SCISSORS
PRESS 7 to play NUMBER GUESSING GAME
CHOOSE GAME TO PLAY : 6


        ROCK, PAPER, SCISSORS GAME

RULES :
You Have To Choose from Rock, Paper & Scissors.
The Computer will also choose from Rock, Paper & Scissors.
If you win then your bet money will be doubled.
If you lose then you will lose your bet money.

PLACE YOUR BET : 20
Enter your choice (R for Rock, P for Paper, S for Scissors): S
Computer chose Paper
You win! Scissors beat Paper

Do you want to continue in the GAME ZONE (Y/N): Y


PRESS 1 to play LUCKY 7
PRESS 2 to play SCRAMBLE WORDS
PRESS 3 to play MAZE GAME
PRESS 4 to play CARDS MATCHING GAME
PRESS 5 to play WORD GUESSING GAME
PRESS 6 to play ROCK, PAPER & SCISSORS
PRESS 7 to play NUMBER GUESSING GAME
CHOOSE GAME TO PLAY : 7


        NUMBER GUESSING GAME

RULES :
You have to guess the number between 0 to 50.
You get 5 guesses.
If you win then your bet money will be doubled.
If you lose then you will lose your bet money.

PLACE YOUR BET : 20
ENTER YOUR GUESS (1): 7
WRONG GUESS BUT CLOSE
Choose a number lower
Guess again

ENTER YOUR GUESS (2): 5

WRONG GUESS, BUT VERY CLOSE
Guess again

ENTER YOUR GUESS (3): 3

WRONG GUESS, BUT VERY CLOSE
Guess again

ENTER YOUR GUESS (4): 4

WRONG GUESS, BUT VERY CLOSE
Guess again

ENTER YOUR GUESS (5): 1

WRONG GUESS !! YOU LOST !!
THE NUMBER WAS 6

Do you want to continue in the GAME ZONE (Y/N): N

DO YOU WANT TO CONTINUE THE PROGRAM (Y/N): Y

PRESS 1 to enter in GAME ZONE
PRESS 2 to see GAMEZONE PERFORMANCE
PRESS 3 to see ACCOUNT DETAILS
ENTER CHOICE : 3
NAME | USERNAME | PASSWORD | BALANCE
k |  Any    |    ghost    |    711

DO YOU WANT TO CONTINUE THE PROGRAM (Y/N): Y

PRESS 1 to enter in GAME ZONE
PRESS 2 to see GAMEZONE PERFORMANCE
PRESS 3 to see ACCOUNT DETAILS
ENTER CHOICE : 2
```

```
TOTAL GAMES :
Lucky7 | Scramble_Word | Maze | Cards_Matching | Word_Guessing | Rock_Paper_Scissors |
Number_Guessing
4      | 3             | 3    | 3              | 3             | 3                   | 3

GAMES WON :
Lucky7 | Scramble_Word | Maze | Cards_Matching | Word_Guessing | Rock_Paper_Scissors |
Number_Guessing
2      | 3             | 3    | 3              | 3             | 3                   | 1

GAMES LOST :
Lucky7 | Scramble_Word | Maze | Cards_Matching | Word_Guessing | Rock_Paper_Scissors |
Number_Guessing
2      | 0             | 0    | 0              | 0             | 0                   | 2

DO YOU WANT TO CONTINUE THE PROGRAM (Y/N): N

*********************************************************************************
                          YOU HAVE BEEN LOGGED OUT
                          THANK YOU FOR CHECKING IN
*********************************************************************************
```

REFERENCE

https://youtu.be/cBJiVypvz-U?si=TtFPjxPQtyQ-Hv8z

https://www.geeksforgeeks.org/time-h-header-file-in-c-with-examples/

https://chatgpt.com/share/672b69d3-c0c0-8000-a323-37664ff5af74

https://www.geeksforgeeks.org/

https://www.hackerrank.com/dashboard

https://www.w3schools.com/sql/