**FIGHTER**



**Session: 2021 – 2024**

**Submitted by:**

M.Abdullah

**Registration No:**

2021-CS-104

**Supervised to:**

MAM MAIDA

**Department of Computer Science**

**University of Engineering and Technology**

**Lahore Pakistan**

**Description and Story of Writing game:**

FIGHTER is a very easy game to play. It was based on TWO FIGHTERS and the player. In fighter player need to kill the opponent and opponent have three life if it will be hited by three shots the game is over and the second player won the game.

**Game Characters Description:**

In FIGHTER, there are three character

1. Player
2. Enemy
3. Opponent

players move toward each other and the player fire on each other.

**Rules and interaction:**

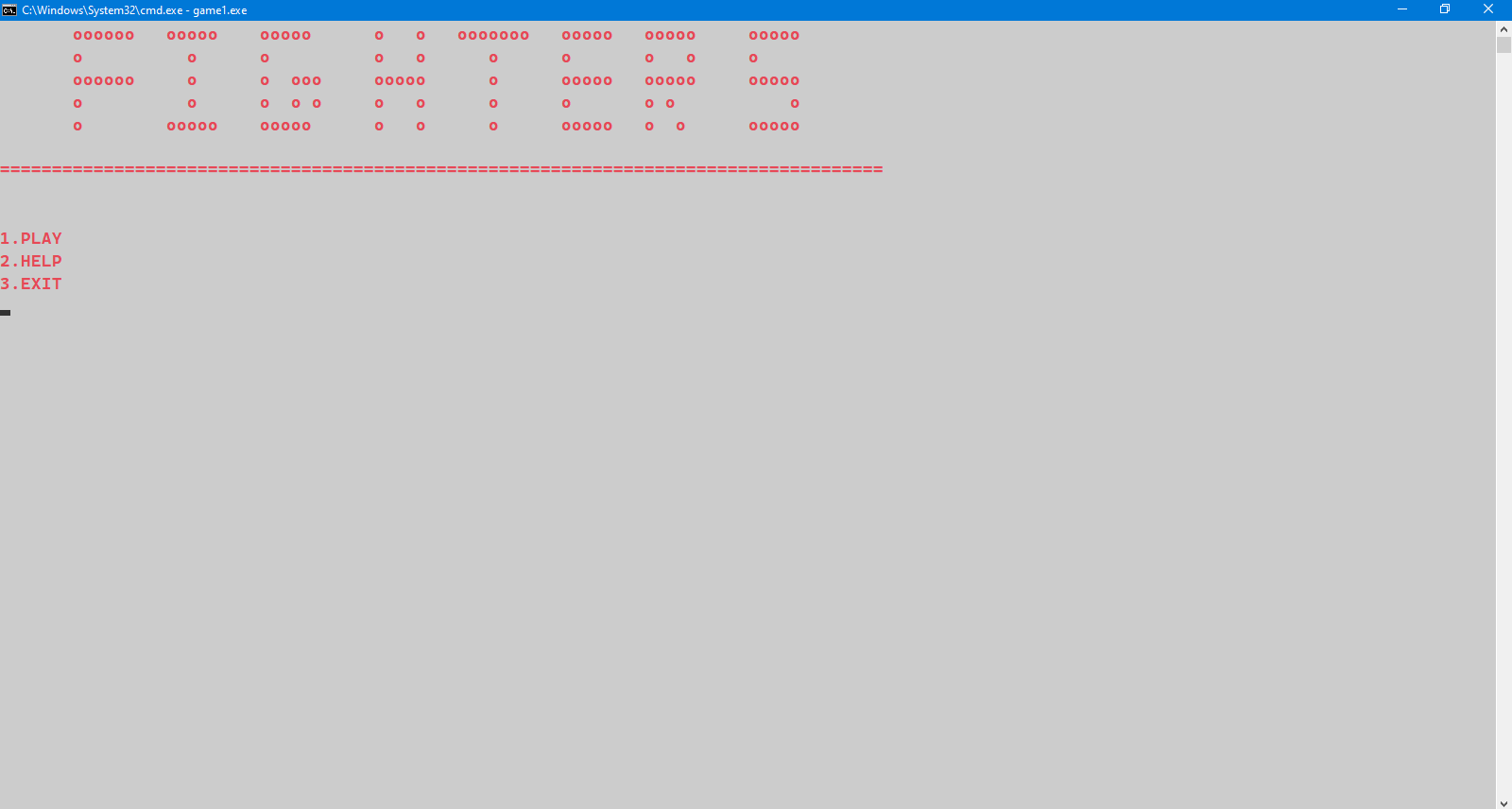
* If fire hit the player it reduces life.
* As fighter killed by player the player won the match.
* Player has three lives to continue the game.
* Use the left or Right arrow key to move left or right.
* No upward downward movement of player.
* Use the A or S key to move left or right.
* No W and X keys for upward and downward movement of player.
* If player kills the other one he will won the game.

**The goal of the Game:**

The goal of the game is to kill 2nd player. If player kill the 2nd one he will be the Sikander of this game. If he could not kill all fighter. 2nd one kill him/her. Player need to kill all enemies and also 2nd player to save his life.

**Wireframes:**

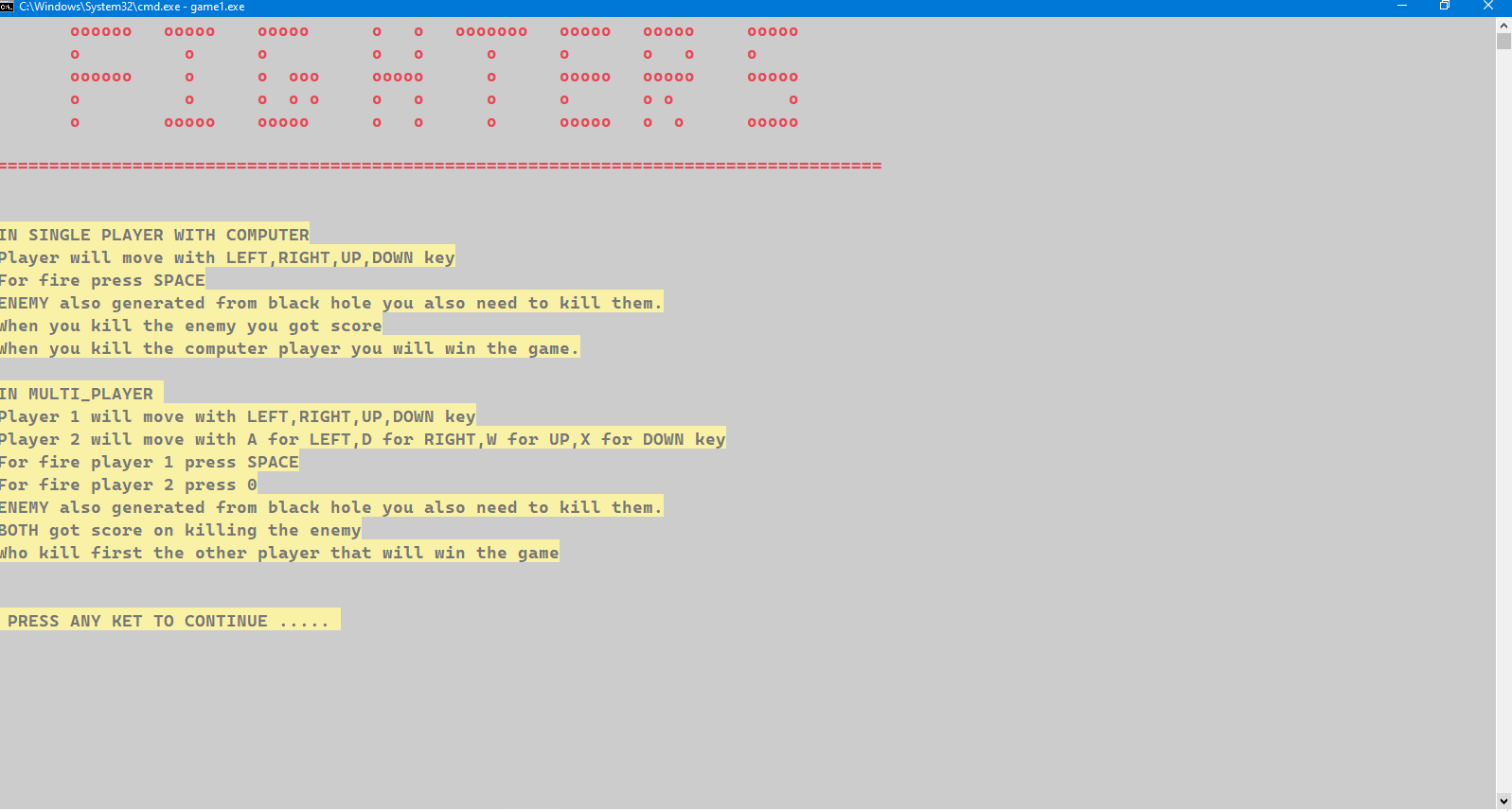
* **Main Menu**



* **PLAY Menu**

****

* **INSTRUCTION**



* **PLAY SCREEN**

**Graphical user interface

Description automatically generated with medium confidence**

**COMPLETE CODE**

#include <iostream>

#include <windows.h>

#include <fstream>

#include <ctime>

#include <conio.h>

#include <math.h>

using namespace std;

void header(HANDLE h);

void clear\_screen();

void help(HANDLE h);

void gotoxy(int x, int y);

void load(char maze[][100], string path, int row, int col);

void print(char maze[][100], int row, int col, HANDLE h);

bool moveLeft(char maze[][100], int px, int py, HANDLE h);

bool moveRight(char maze[][100], int px, int py, HANDLE h);

bool moveup(char maze[][100], int px, int py, HANDLE h);

bool movedown(char maze[][100], int px, int py, HANDLE h);

void fire(char maze[][100], int px, int py, HANDLE h);

void firemove(char maze[][100], int row, int col, HANDLE h);

int enemyDirection();

int enemy0Direction();

main()

{

    system("cls");

    HANDLE h = GetStdHandle(STD\_OUTPUT\_HANDLE);

    char maze[30][100];

    string path = "maze.txt";

    int row = sizeof(maze) / sizeof(maze[0]);

    int col = sizeof(maze[0]) / sizeof(maze[0][0]);

    load(maze, path, row, col);

    bool gameplay = 0;

    string op = " ";

    int px = 15;

    int py = 5;

    int px2 = 15;

    int py2 = 95;

    maze[px2][py2] = '0';

    int life1 = 3, life2 = 3;

    int Ecount = 0;

    int ex = 3;

    int ey = 48;

    while (op != "3")

    {

        header(h);

        cout << "1.PLAY" << endl;

        cout << "2.HELP" << endl;

        cout << "3.EXIT" << endl;

        cin >> op;

        if (op == "1")

        {

            string op2 = " ";

            while (op2 != "3")

            {

                header(h);

                cout << "1.MULTIPLAYER " << endl;

                cout << "2.PLAY WITH COMPUTER" << endl;

                cout << "3.EXIT" << endl;

                cout << life1 << "   " << life2 << endl;

                cin >> op2;

                if (op2 == "1" || op2 == "2")

                {

                    px = 15;

                    py = 5;

                    px2 = 15;

                    py2 = 95;

                    maze[px][py] = 'O';

                    maze[px2][py2] = '0';

                    Ecount = 0;

                    ex = 3;

                    ey = 48;

                    gameplay = 0;

                    print(maze, row, col, h);

                    while (gameplay != 1)

                    {

                        gameplay = 0;

                        Sleep(50);

                        bool check;

                        if (GetAsyncKeyState(0x41))

                        {

                            check = moveLeft(maze, px, py, h);

                            if (check == 1)

                            {

                                SetConsoleTextAttribute(h, 10);

                                maze[px][py] = ' ';

                                gotoxy(py, px);

                                cout << ' ';

                                py = py - 1;

                                maze[px][py] = 'O';

                                gotoxy(py, px);

                                cout << maze[px][py];

                            }

                        }

                        if (GetAsyncKeyState(0x53))

                        {

                            check = moveRight(maze, px, py, h);

                            if (check == 1)

                            {

                                SetConsoleTextAttribute(h, 10);

                                maze[px][py] = ' ';

                                gotoxy(py, px);

                                cout << " ";

                                py = py + 1;

                                maze[px][py] = 'O';

                                gotoxy(py, px);

                                cout << maze[px][py];

                            }

                        }

                        if (GetAsyncKeyState(0x57))

                        {

                            check = moveup(maze, px, py, h);

                            if (check == 1)

                            {

                                SetConsoleTextAttribute(h, 10);

                                maze[px][py] = ' ';

                                gotoxy(py, px);

                                cout << ' ';

                                px = px - 1;

                                maze[px][py] = 'O';

                                gotoxy(py, px);

                                cout << maze[px][py];

                            }

                        }

                        if (GetAsyncKeyState(0x5A))

                        {

                            check = movedown(maze, px, py, h);

                            if (check == 1)

                            {

                                SetConsoleTextAttribute(h, 10);

                                maze[px][py] = ' ';

                                gotoxy(py, px);

                                cout << ' ';

                                px = px + 1;

                                maze[px][py] = 'O';

                                gotoxy(py, px);

                                cout << maze[px][py];

                            }

                        }

                        if (GetAsyncKeyState(0x46))

                        {

                            SetConsoleTextAttribute(h, 2);

                            if (maze[px][py + 1] == ' ')

                            {

                                maze[px][py + 1] = '.';

                                gotoxy(py + 1, px);

                                cout << maze[px][py + 1];

                            }

                        }

                        //=======================P1 FIRE MOVE

                        for (int i = row - 1; i > 0; i--)

                        {

                            for (int j = col; j > 0; j--)

                            {

                                if (maze[i][j] == '.')

                                {

                                    maze[i][j] = ' ';

                                    gotoxy(j, i);

                                    cout << maze[i][j];

                                    if (maze[i][j + 1] == '0' || maze[i][j + 1] == 'E')

                                    {

                                        if (maze[i][j + 1] == '0')

                                        {

                                            life2--;

                                        }

                                        if (maze[i][j + 1] == 'E')

                                        {

                                            maze[ex][ey] = ' ';

                                            ex = 3;

                                            ey = 48;

                                        }

                                    }

                                    if (maze[i][j + 1] == ' ')

                                    {

                                        maze[i][j + 1] = '.';

                                        gotoxy(j + 1, i);

                                        cout << maze[i][j + 1];

                                    }

                                }

                            }

                        }

                        //--------------------------------------------------player 2-------------------------------------------------------------------

                        //---------------------------------------------------left

                        if (op2 == "1")

                        {

                            if (GetAsyncKeyState(VK\_LEFT))

                            {

                                if (maze[px2][py2 - 1] == ' ')

                                {

                                    SetConsoleTextAttribute(h, 4);

                                    maze[px2][py2] = ' ';

                                    gotoxy(py2, px2);

                                    cout << ' ';

                                    py2 = py2 - 1;

                                    maze[px2][py2] = 'O';

                                    gotoxy(py2, px2);

                                    cout << maze[px2][py2];

                                }

                            }

                            if (GetAsyncKeyState(VK\_RIGHT))

                            {

                                if (maze[px2][py2 + 1] == ' ')

                                {

                                    SetConsoleTextAttribute(h, 4);

                                    maze[px2][py2] = ' ';

                                    gotoxy(py2, px2);

                                    cout << " ";

                                    py2 = py2 + 1;

                                    maze[px2][py2] = 'O';

                                    gotoxy(py2, px2);

                                    cout << maze[px2][py2];

                                }

                            }

                            if (GetAsyncKeyState(VK\_UP))

                            {

                                if (maze[px2 - 1][py2] == ' ')

                                {

                                    SetConsoleTextAttribute(h, 4);

                                    maze[px2][py2] = ' ';

                                    gotoxy(py2, px2);

                                    cout << ' ';

                                    px2 = px2 - 1;

                                    maze[px2][py2] = 'O';

                                    gotoxy(py2, px2);

                                    cout << maze[px2][py2];

                                }

                            }

                            if (GetAsyncKeyState(VK\_DOWN))

                            {

                                if (maze[px2 + 1][py2] == ' ')

                                {

                                    SetConsoleTextAttribute(h, 4);

                                    maze[px2][py2] = ' ';

                                    gotoxy(py2, px2);

                                    cout << ' ';

                                    px2 = px2 + 1;

                                    maze[px2][py2] = 'O';

                                    gotoxy(py2, px2);

                                    cout << maze[px2][py2];

                                }

                            }

                            if (GetAsyncKeyState(VK\_SPACE))

                            {

                                SetConsoleTextAttribute(h, 2);

                                if (maze[px2][py2 - 1] == ' ')

                                {

                                    maze[px2][py2 - 1] = '-';

                                    gotoxy(py2 - 1, px2);

                                    cout << maze[px2][py2 - 1];

                                }

                            }

                            //=======================P1 FIRE MOVE

                        }

                        // -------------------------------------------COMPUTER PLAYER

                        if (op2 == "2")

                        {

                            int rom = enemy0Direction();

                            if (rom == 1)

                            {

                                if (maze[px2][py2 - 1] == ' ')

                                {

                                    SetConsoleTextAttribute(h, 14);

                                    maze[px2][py2] = ' ';

                                    gotoxy(py2, px2);

                                    cout << ' ';

                                    py2 = py2 - 1;

                                    maze[px2][py2] = '0';

                                    gotoxy(py2, px2);

                                    cout << maze[px2][py2];

                                }

                            }

                            else if (rom == 2)

                            {

                                if (maze[px2][py2 + 1] == ' ')

                                {

                                    SetConsoleTextAttribute(h, 14);

                                    maze[px2][py2] = ' ';

                                    gotoxy(py2, px2);

                                    cout << ' ';

                                    py2 = py2 + 1;

                                    maze[px2][py2] = '0';

                                    gotoxy(py2, px2);

                                    cout << maze[px2][py2];

                                }

                            }

                            else if (rom == 3)

                            {

                                if (maze[px2 + 1][py2] == ' ')

                                {

                                    SetConsoleTextAttribute(h, 14);

                                    maze[px2][py2] = ' ';

                                    gotoxy(py2, px2);

                                    cout << ' ';

                                    px2 = px2 + 1;

                                    maze[px2][py2] = '0';

                                    gotoxy(py2, px2);

                                    cout << maze[px2][py2];

                                }

                            }

                            else if (rom == 4)

                            {

                                if (maze[px2 - 1][py2] == ' ')

                                {

                                    SetConsoleTextAttribute(h, 14);

                                    maze[px2][py2] = ' ';

                                    gotoxy(py2, px2);

                                    cout << ' ';

                                    px2 = px2 - 1;

                                    maze[px2][py2] = '0';

                                    gotoxy(py2, px2);

                                    cout << maze[px2][py2];

                                }

                            }

                            if (maze[px2][py2 - 5] == 'O' || maze[px2][py2 - 4] == 'O' || maze[px2][py2 - 3] == 'O' || maze[px2][py2 - 2] == 'O' || maze[px2][py2 - 1] == 'O')

                            {

                                SetConsoleTextAttribute(h, 2);

                                maze[px2][py2 - 1] = '-';

                                gotoxy(py2 - 1, px2);

                                cout << maze[px2][py2 - 1];

                            }

                            if (maze[px2][py2 + 5] == 'O' || maze[px2][py2 + 4] == 'O' || maze[px2][py2 + 3] == 'O' || maze[px2][py2 + 2] == 'O' || maze[px2][py2 + 1] == 'O' || maze[px2][py2 + 5] == '0' || maze[px2][py2 + 4] == '0' || maze[px2][py2 + 3] == '0' || maze[px2][py2 + 2] == '0' || maze[px2][py2 + 1] == '0')

                            {

                                maze[px2][py2 + 1] = '.';

                                gotoxy(py2 + 1, px2);

                                cout << maze[px2][py2 + 1];

                            }

                        }

                        //--------------------------fire move--------------------------------------------------------

                        for (int i = 0; i < row - 1; i++)

                        {

                            for (int j = 0; j < col; j++)

                            {

                                if (maze[i][j] == '-')

                                {

                                    maze[i][j] = ' ';

                                    gotoxy(j, i);

                                    cout << maze[i][j];

                                    if (maze[i][j - 1] == 'O')

                                    {

                                        life1--;

                                    }

                                    if (maze[i][j - 1] == 'E')

                                    {

                                        maze[ex][ey] = ' ';

                                        ex = 3;

                                        ey = 48;

                                    }

                                    if (maze[i][j - 1] == '0')

                                    {

                                        life2--;

                                    }

                                    if (maze[i][j - 1] == ' ')

                                    {

                                        maze[i][j - 1] = '-';

                                        gotoxy(j - 1, i);

                                        cout << maze[i][j - 1];

                                    }

                                }

                            }

                        }

                        gotoxy(80, 31);

                        cout << "P2 LIFE : " << life2;

                        gotoxy(0, 31);

                        cout << "P1 LIFE : " << life1;

                        for (int i = 0; i < row; i++)

                        {

                            for (int j = 0; j < col; j++)

                            {

                                if (maze[i][j] == 'E')

                                {

                                    Ecount++;

                                }

                            }

                        }

                        //------------------------ENEMY-------------------------------------------------------------------------------

                        if (Ecount == 0)

                        {

                            maze[ex][ey] = 'E';

                        }

                        int rom = enemyDirection();

                        if (rom == 1)

                        {

                            if (maze[ex][ey - 1] == ' ')

                            {

                                SetConsoleTextAttribute(h, 14);

                                maze[ex][ey] = ' ';

                                gotoxy(ey, ex);

                                cout << ' ';

                                ey = ey - 1;

                                maze[ex][ey] = 'E';

                                gotoxy(ey, ex);

                                cout << maze[ex][ey];

                            }

                        }

                        else if (rom == 2)

                        {

                            if (maze[ex][ey + 1] == ' ')

                            {

                                SetConsoleTextAttribute(h, 14);

                                maze[ex][ey] = ' ';

                                gotoxy(ey, ex);

                                cout << ' ';

                                ey = ey + 1;

                                maze[ex][ey] = 'E';

                                gotoxy(ey, ex);

                                cout << maze[ex][ey];

                            }

                        }

                        else if (rom == 3)

                        {

                            if (maze[ex + 1][ey] == ' ')

                            {

                                SetConsoleTextAttribute(h, 14);

                                maze[ex][ey] = ' ';

                                gotoxy(ey, ex);

                                cout << ' ';

                                ex = ex + 1;

                                maze[ex][ey] = 'E';

                                gotoxy(ey, ex);

                                cout << maze[ex][ey];

                            }

                        }

                        else if (rom == 4)

                        {

                            if (maze[ex - 1][ey] == ' ')

                            {

                                SetConsoleTextAttribute(h, 14);

                                maze[ex][ey] = ' ';

                                gotoxy(ey, ex);

                                cout << ' ';

                                ex = ex - 1;

                                maze[ex][ey] = 'E';

                                gotoxy(ey, ex);

                                cout << maze[ex][ey];

                            }

                        }

                        if (maze[ex][ey - 5] == 'O' || maze[ex][ey - 4] == 'O' || maze[ex][ey - 3] == 'O' || maze[ex][ey - 2] == 'O' || maze[ex][ey - 1] == 'O' || maze[ex][ey - 5] == '0' || maze[ex][ey - 4] == '0' || maze[ex][ey - 3] == '0' || maze[ex][ey - 2] == '0' || maze[ex][ey - 1] == '0')

                        {

                            SetConsoleTextAttribute(h, 2);

                            if (maze[ex][ey - 1] == ' ')

                            {

                                maze[ex][ey - 1] = '-';

                                gotoxy(ey - 1, ex);

                                cout << maze[ex][ey - 1];

                            }

                        }

                        if (maze[ex][ey + 5] == 'O' || maze[ex][ey + 4] == 'O' || maze[ex][ey + 3] == 'O' || maze[ex][ey + 2] == 'O' || maze[ex][ey + 1] == 'O' || maze[ex][ey + 5] == '0' || maze[ex][ey + 4] == '0' || maze[ex][ey + 3] == '0' || maze[ex][ey + 2] == '0' || maze[ex][ey + 1] == '0')

                        {

                            maze[ex][ey + 1] = '.';

                            gotoxy(ey + 1, ex);

                            cout << maze[ex][ey + 1];

                        }

                        if (GetAsyncKeyState(VK\_ESCAPE) || life1 == 0 || life2 == 0)

                        {

                            gameplay = 1; // Stop the game

                        }

                    }

                }

                else if (op2 != "3")

                {

                    header(h);

                    cout << "ENTER THE VALID OPTION" << endl;

                    clear\_screen();

                }

                maze[px2][py2] = ' ';

                maze[px][py] = ' ';

                maze[ex][ey] = ' ';

                Ecount = 0;

                for (int i = 0; i > row - 1; i++)

                {

                    for (int j = 0; j > col; j++)

                    {

                        if (maze[i][j] == '-')

                        {

                            maze[i][j] = ' ';

                        }

                        else if (maze[i][j] == '.')

                        {

                            maze[i][j] = ' ';

                        }

                    }

                }

                header(h);

                clear\_screen();

                if (life1 < 1)

                {

                    cout << "PLAYER 2 WIN THE MATCH" << endl;

                    clear\_screen();

                    getch();

                    life1 = 3;

                    life2 = 3;

                }

                if (life2 < 1)

                {

                    cout << "PLAYER 1 WIN THE MATCH" << endl;

                    clear\_screen();

                    getch();

                    life1 = 3;

                    life2 = 3;

                }

            }

        }

        else if (op == "2")

        {

            header(h);

            help(h);

        }

        else if (op != "3")

        {

            cout << "ENTER THE VALID OPTION" << endl;

            clear\_screen();

        }

    }

}

void gotoxy(int x, int y)

{

    COORD coordinates;

    coordinates.X = x;

    coordinates.Y = y;

    SetConsoleCursorPosition(GetStdHandle(STD\_OUTPUT\_HANDLE), coordinates);

}

void load(char maze[][100], string path, int row, int col)

{

    fstream my;

    string record;

    my.open(path, ios::in);

    for (int x = 0; x < row; x++)

    {

        getline(my, record);

        for (int y = 0; y < col; y++)

        {

            maze[x][y] = record[y];

        }

    }

}

void print(char maze[][100], int row, int col, HANDLE h)

{

    system("cls");

    for (int x = 0; x < row; x++)

    {

        for (int y = 0; y < col; y++)

        {

            if (maze[x][y] == '=')

            {

                SetConsoleTextAttribute(h, 1);

                cout << maze[x][y];

            }

            else if (maze[x][y] == '-')

            {

                SetConsoleTextAttribute(h, 3);

                cout << maze[x][y];

            }

            else

            {

                SetConsoleTextAttribute(h, 5);

                cout << maze[x][y];

            }

        }

        cout << endl;

    }

}

bool moveLeft(char maze[][100], int px, int py, HANDLE h)

{

    if (maze[px][py - 1] == ' ')

    {

        return 1;

    }

    return 0;

}

bool moveRight(char maze[][100], int px, int py, HANDLE h)

{

    if (maze[px][py + 1] == ' ')

    {

        return 1;

    }

    return 0;

}

bool moveup(char maze[][100], int px, int py, HANDLE h)

{

    if (maze[px - 1][py] == ' ')

    {

        return 1;

    }

    return 0;

}

bool movedown(char maze[][100], int px, int py, HANDLE h)

{

    if (maze[px + 1][py] == ' ')

    {

        return 1;

    }

    return 0;

}

void fire(char maze[][100], int px, int py, HANDLE h)

{

}

void firemove(char maze[][100], int row, int col, HANDLE h)

{

}

void clear\_screen()

{

    cout << endl

         << " PRESS ANY KET TO CONTINUE ..... " << endl;

    getch();

    system("cls");

}

void header(HANDLE h)

{

    system("cls");

    SetConsoleTextAttribute(h, 124);

    cout << "       oooooo   ooooo    ooooo      o   o   ooooooo   ooooo   ooooo     ooooo     " << endl;

    cout << "       o          o      o          o   o      o      o       o   o     o         " << endl;

    cout << "       oooooo     o      o  ooo     ooooo      o      ooooo   ooooo     ooooo     " << endl;

    cout << "       o          o      o  o o     o   o      o      o       o o           o     " << endl;

    cout << "       o        ooooo    ooooo      o   o      o      ooooo   o  o      ooooo     " << endl

         << endl;

    cout << "=====================================================================================" << endl

         << endl

         << endl;

}

void help(HANDLE h)

{

    SetConsoleTextAttribute(h, 232);

    cout << "IN SINGLE PLAYER WITH COMPUTER" << endl;

    cout << "Player will move with LEFT,RIGHT,UP,DOWN key" << endl;

    cout << "For fire press SPACE" << endl;

    cout << "ENEMY also generated from black hole you also need to kill them." << endl;

    cout << "When you kill the enemy you got score" << endl;

    cout << "When you kill the computer player you will win the game." << endl

         << endl;

    cout << "IN MULTI\_PLAYER " << endl;

    cout << "Player 1 will move with LEFT,RIGHT,UP,DOWN key" << endl;

    cout << "Player 2 will move with A for LEFT,D for RIGHT,W for UP,X for DOWN key" << endl;

    cout << "For fire player 1 press SPACE" << endl;

    cout << "For fire player 2 press 0" << endl;

    cout << "ENEMY also generated from black hole you also need to kill them." << endl;

    cout << "BOTH got score on killing the enemy" << endl;

    cout << "Who kill first the other player that will win the game" << endl

         << endl;

    clear\_screen();

}

int enemyDirection()

{

    srand(time(0));

    int result = 1 + (rand() % 4);

    return result;

}

int enemy0Direction()

{

    srand(time(0));

    int result = 2 + (rand() % 4);

    return result;

}

THE END