# 

**EveryOne is Imposter!!!**

A logo of a university of engineering and technology

Description automatically generated

# Session 2023 – 2027

# Submitted by:

Saad Qaiser 2023-CS-37

# Supervised by:

Dr. Muhammad Awais Hassan

# Course:

CSC-102 Programming Fundamentals

Department of Computer Science

# University of Engineering and Technology

# Lahore Pakistan

Table of Contents

[ Description 3](#_Toc153824134)

[ Characters 3](#_Toc153824135)

[ Game Objects 3](#_Toc153824136)

[ Rules & Interactions 3](#_Toc153824137)

[ Goal of The Game 4](#_Toc153824138)

[ Wireframes of the Game 4](#_Toc153824139)

[ Function Prototypes 8](#_Toc153824140)

[ Complete Code 10](#_Toc153824141)

# EveryOne is Imposter!!!

" EveryOne is Imposter!!!" is an action-packed game where players step into the shoes of a resilient hero aiming to reclaim their village from three different small but formidable enemies. Navigate through the intricacies of the maze-like village, overcoming challenges posed by each unique adversary. As the hero prevails against these smaller foes, they will face the ultimate test in a final showdown with the powerful and deceptive Boss. With intuitive controls and strategic gameplay, " EveryOne is Imposter!!!" promises an engaging adventure as players strive to liberate their village and emerge victorious against the odds.

# Characters

This following are the characters of the game:

* Hero: The hero can move throughout the maze and can also shoot up and down.
* Enemy 1: The enemy is present on the top left corner of the maze and moves diagnal whilst also shooting bullets.
* Enemy 2: This is one moves up and down on the top right corner of the maze while also fires bullets.
* Enemy 3: This enemy is present near the spawn point of the hero and shoots bullets upwards.
* Final Boss : It is larger in size than other enemies and shoots bullets whilst moving random. It appears in the final level.

# Game Objects

The object in the game is:

* + A health boost is given at final stage which hero can collect to change his health from a max to 400 to make it easier to fight the final villain. It is present in the center of the maze..

# Rules & Interactions

The rules of the game are:

* The hero must kill all the enemies in the first level and the final level.
* The hero can utilize the health boost given in the first level of the game.
* The hero’s health starts at 300 and with every hit it decreases by 50. If the health goes below zero, then the hero dies, and the game is over.
* The enemy dies with three direct hits of bullets from the hero.
* The hero and enemies health are given at the top of the maze.

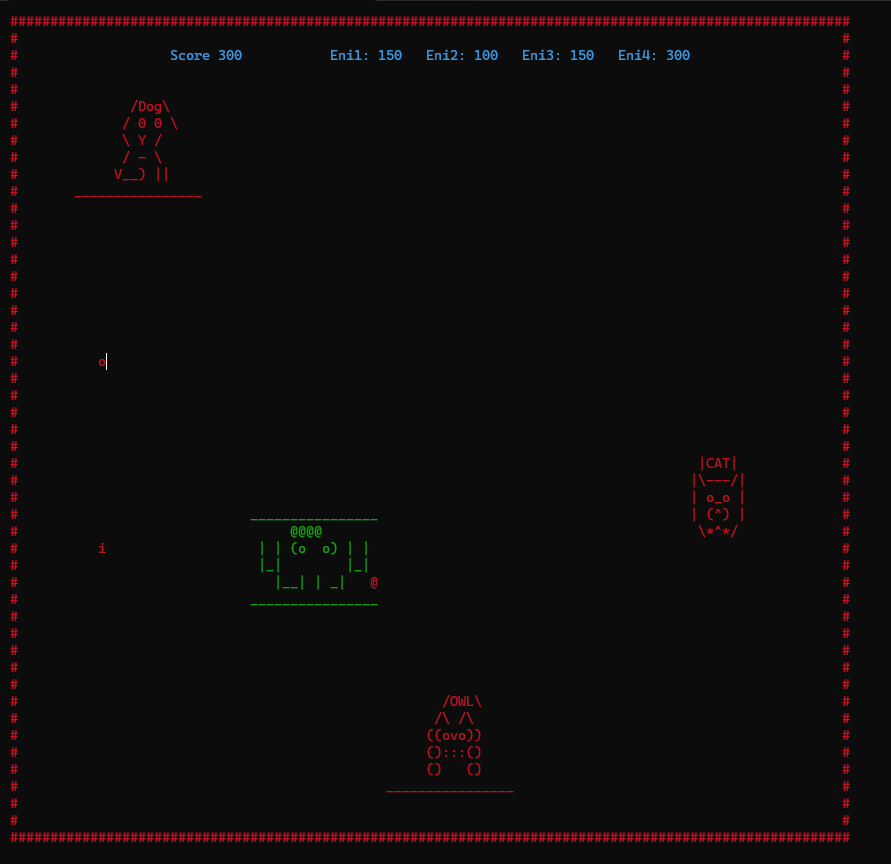
# Goal of The Game

The goal of the game is that the hero must kill all enemies while keeping himself alive to reach the final level. Where the hero will face boss villain. The hero will eliminate them and win the game.

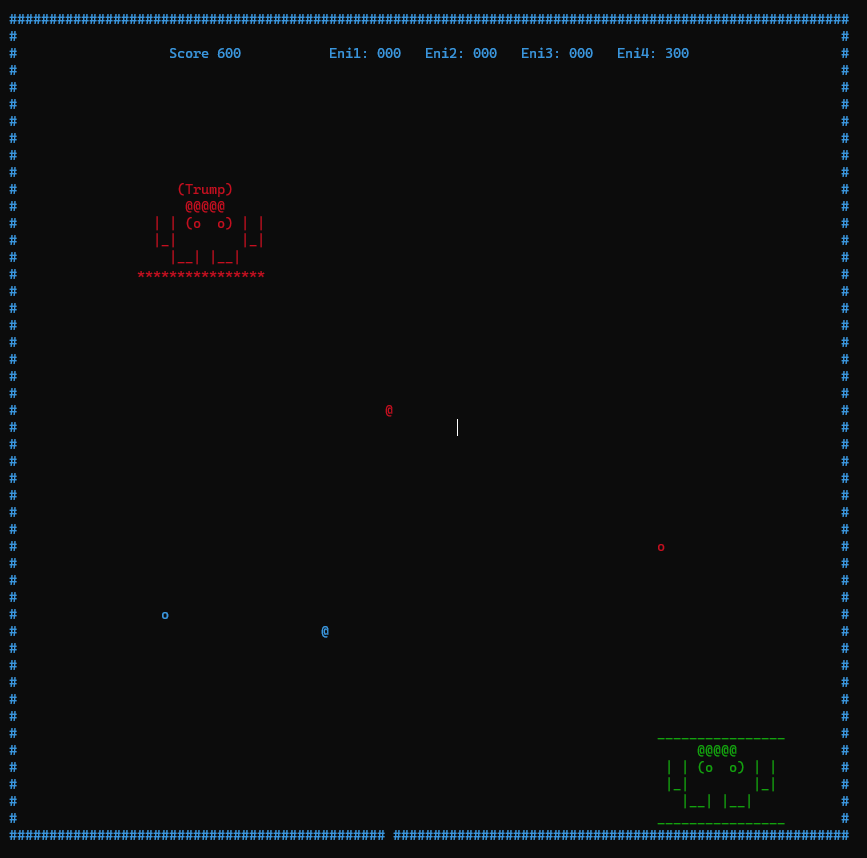
# Wireframes of the Game



**Figure 1: Starting Msg**



**Figure 2: First Level**



**Figure 3: Final Level**



**Figure 4: Losing Msg**



**Figure 5: Winning Msg**

# Function Prototypes

* void gotoxy(int x, int y);
* void printEnemy1();
* void eraseEnemy1();
* void printEnemy2();
* void eraseEnemy2();
* void printEnemy3();
* void eraseEnemy3();
* void printEnemy4();
* void eraseEnemy4();
* void printPlayer();
* void erasePlayer();
* void moveLeft();
* void moveRight();
* void moveUp();
* void moveDown();
* void moveEnemy1();
* void moveEnemy2();
* void moveEnemy3();
* void moveEnemy4(int digit);
* void printMaze();
* void printscore();
* void erase();
* void eraseR();
* void health();
* void header();
* void fire();
* void fireR();
* void bullet();
* void bulletR();
* void healthErase();
* void moveBullet();
* void moveBulletR();
* void printEnemybullets();
* void moveEnemyBullet();
* void checkCon();
* void enemyfire();
* void lose();
* void win();
* int random(int min, int max);
* char getCharAtxy(short int x, short int y);

# Complete Code

#include <iostream>

#include <windows.h>

#include <conio.h>

using namespace std;

void gotoxy(int x, int y);

void printEnemy1();

void eraseEnemy1();

void printEnemy2();

void eraseEnemy2();

void printEnemy3();

void eraseEnemy3();

void printEnemy4();

void eraseEnemy4();

void printPlayer();

void erasePlayer();

void moveLeft();

void moveRight();

void moveUp();

void moveDown();

void moveEnemy1();

void moveEnemy2();

void moveEnemy3();

void moveEnemy4(int digit);

void printMaze();

void printscore();

void erase();

void eraseR();

void health();

void header();

void fire();

void fireR();

void bullet();

void bulletR();

void healthErase();

void moveBullet();

void moveBulletR();

void printEnemybullets();

void moveEnemyBullet();

void checkCon();

void enemyfire();

void lose();

void win();

int random(int min, int max);

char getCharAtxy(short int x, short int y);

int eX1 = 2, eY1 = 2;

int eX2 = 80, eY2 = 4;

int eX3 = 40, eY3 = 40;

int eX4 = 10, eY4 = 10;

int hit1 = 0, hit2 = 0, hit3 = 0, hit4 = 0;

int pX = 30, pY = 30;

int hX = 40, hY = 20;

int bulletX, bulletY;

int bulletRX, bulletRY;

int enemy1bulletX, enemy1bulletY;

int enemy2bulletX, enemy2bulletY;

int enemy3bulletX, enemy3bulletY;

int enemy4bulletX, enemy4bulletY;

bool bulletActive = false;

bool bulletActiveR = false;

bool firstEnemybullet = false, secondEnemybullet = false, thirdEnemybullet = false, forthEnemybullet = false;

int score = 300;

int e1score = 150, e2score = 150, e3score = 150, e4score = 300;

main()

{

system("cls");

header();

system("cls");

printMaze();

printEnemy1();

printEnemy2();

printEnemy3();

printPlayer();

//health();

while (true)

{

printscore();

if (GetAsyncKeyState(VK\_LEFT))

{

moveLeft();

}

if (GetAsyncKeyState(VK\_RIGHT))

{

moveRight();

}

if (GetAsyncKeyState(VK\_UP))

{

moveUp();

}

if (GetAsyncKeyState(VK\_DOWN))

{

moveDown();

}

if (GetAsyncKeyState(VK\_SPACE))

{

fire();

}

if (GetAsyncKeyState(VK\_SHIFT))

{

fireR();

}

moveEnemy1();

moveEnemy2();

moveEnemy3();

enemyfire();

moveEnemyBullet();

printEnemybullets();

if (bulletActive)

{

moveBullet();

checkCon();

}

if (bulletActiveR)

{

moveBulletR();

checkCon();

}

if (e1score == 0 || e1score < 0)

{

eraseEnemy1();

firstEnemybullet = false;

}

if (e2score == 0 || e2score < 0)

{

eraseEnemy2();

secondEnemybullet = false;

}

if (e3score == 0 || e3score < 0)

{

eraseEnemy3();

thirdEnemybullet = false;

}

if (e1score == 0 && e2score == 0 && e3score == 0)

{

int a = random(0, 1);

printEnemy4();

moveEnemy4(a);

if (pX == hX && pY >= hY && pY <= hY + 2)

{

healthErase();

score = 300;

}

health();

}

if (e4score == 0)

{

eraseEnemy4();

forthEnemybullet = false;

win();

Sleep(200);

break;

}

if (score == 0)

{

erasePlayer();

lose();

Sleep(200);

break;

}

Sleep(50);

}

// header();

}

//this funtion is used to print enemy1

void printEnemy1()

{

SetConsoleTextAttribute(GetStdHandle(STD\_OUTPUT\_HANDLE),4);

gotoxy(eX1, eY1);

cout << " /Dog\\ ";

gotoxy(eX1, eY1 + 1);

cout << " / 0 0 \\ ";

gotoxy(eX1, eY1 + 2);

cout << " \\ Y / ";

gotoxy(eX1, eY1 + 3);

cout << " / - \\ ";

gotoxy(eX1, eY1 + 4);

cout << " V\_\_) || ";

gotoxy(eX1, eY1 + 5);

cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_";

}

//this funtion is used to print enemy2

void printEnemy2()

{

SetConsoleTextAttribute(GetStdHandle(STD\_OUTPUT\_HANDLE),4);

gotoxy(eX2, eY2);

cout << " |CAT| ";

gotoxy(eX2, eY2 + 1);

cout << " |\\---/| ";

gotoxy(eX2, eY2 + 2);

cout << " | o\_o | ";

gotoxy(eX2, eY2 + 3);

cout << " | (^) | ";

gotoxy(eX2, eY2 + 4);

cout << " \\\*^\*/ ";

}

//this funtion is used to erase enemy1

void eraseEnemy1()

{

gotoxy(eX1, eY1);

cout << " ";

gotoxy(eX1, eY1 + 1);

cout << " ";

gotoxy(eX1, eY1 + 2);

cout << " ";

gotoxy(eX1, eY1 + 3);

cout << " ";

gotoxy(eX1, eY1 + 4);

cout << " ";

gotoxy(eX1, eY1 + 5);

cout << " ";

}

//this funtion is used to erase enemy2

void eraseEnemy2()

{

gotoxy(eX2, eY2);

cout << " ";

gotoxy(eX2, eY2 + 1);

cout << " ";

gotoxy(eX2, eY2 + 2);

cout << " ";

gotoxy(eX2, eY2 + 3);

cout << " ";

gotoxy(eX2, eY2 + 4);

cout << " ";

}

//this funtion is used to print enemy3

void printEnemy3()

{

SetConsoleTextAttribute(GetStdHandle(STD\_OUTPUT\_HANDLE),4);

gotoxy(eX3, eY3);

cout << " /OWL\\ ";

gotoxy(eX3, eY3 + 1);

cout << " /\\ /\\ ";

gotoxy(eX3, eY3 + 2);

cout << " ((ovo)) ";

gotoxy(eX3, eY3 + 3);

cout << " ():::() ";

gotoxy(eX3, eY3 + 4);

cout << " () () ";

gotoxy(eX3, eY3 + 5);

cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_";

}

//this funtion is used to print enemy4

void printEnemy4()

{

SetConsoleTextAttribute(GetStdHandle(STD\_OUTPUT\_HANDLE),4);

gotoxy(eX4, eY4);

cout << " (Trump) ";

gotoxy(eX4, eY4 + 1);

cout << " @@@@@ ";

gotoxy(eX4, eY4 + 2);

cout << " | | (o o) | |";

gotoxy(eX4, eY4 + 3);

cout << " |\_| |\_|";

gotoxy(eX4, eY4 + 4);

cout << " |\_\_| |\_\_| ";

gotoxy(eX4, eY4 + 5);

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";

}

//this funtion is used to erase enemy3

void eraseEnemy3()

{

gotoxy(eX3, eY3);

cout << " ";

gotoxy(eX3, eY3 + 1);

cout << " ";

gotoxy(eX3, eY3 + 2);

cout << " ";

gotoxy(eX3, eY3 + 3);

cout << " ";

gotoxy(eX3, eY3 + 4);

cout << " ";

gotoxy(eX3, eY3 + 5);

cout << " ";

}

//this funtion is used to erase enemy4

void eraseEnemy4()

{

gotoxy(eX4, eY4);

cout << " ";

gotoxy(eX4, eY4 + 1);

cout << " ";

gotoxy(eX4, eY4 + 2);

cout << " ";

gotoxy(eX4, eY4 + 3);

cout << " ";

gotoxy(eX4, eY4 + 4);

cout << " ";

gotoxy(eX4, eY4 + 5);

cout << " ";

}

//this funtion is used to move enemy1

void moveEnemy1()

{

if (firstEnemybullet)

{

eraseEnemy1();

if (hit1 == 0)

{

eX1 = eX1 + 2;

eY1 = eY1 + 1;

}

if (eX1 == 29 || eY1 == 16)

{

hit1 = 1;

}

if (hit1 == 1)

{

eX1 = eX1 - 2;

eY1 = eY1 - 1;

}

if (eX1 == 2 || eY1 == 2)

{

hit1 = 0;

}

printEnemy1();

}

}

//this funtion is used to move enemy2

void moveEnemy2()

{

if (secondEnemybullet)

{

eraseEnemy2();

if (hit2 == 0)

{

eY2 = eY2 + 1;

}

if (eY2 == 30)

{

hit2 = 1;

}

if (hit2 == 1)

{

eY2 = eY2 - 1;

}

if (eY2 == 4)

{

hit2 = 0;

}

printEnemy2();

}

}

//this funtion is used to move enemy3

void moveEnemy3()

{

if (thirdEnemybullet)

{

eraseEnemy3();

if (hit3 == 0)

{

eX3 = eX3 + 1;

}

if (eX3 == 60)

{

hit3 = 1;

}

if (hit3 == 1)

{

eX3 = eX3 - 1;

}

if (eX3 == 40)

{

hit3 = 0;

}

printEnemy3();

}

}

//this funtion is used to move enemy4 randomly

void moveEnemy4(int digit)

{

eraseEnemy4();

if (digit == 0)

{

eX4 = eX4 + 1;

}

if(eX4>2)

{

if(digit!=0)

{

eX4 = eX4 - 1;

}

}

if (eX4 == 80)

{

hit4 = 1;

}

if (hit4 == 1)

{

if (eX4 == 20)

{

hit4 = 0;

}

}

printEnemy4();

}

//this funtion is used to have randomness in movement

int random(int min, int max)

{

return rand() % (max - min + 1) + min;

}

//this funtion is used to print maze

void printMaze()

{

cout << "#########################################################################################################" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "# #" << endl;

cout << "#########################################################################################################" << endl;

}

//this funtion is used to print player

void printPlayer()

{

SetConsoleTextAttribute(GetStdHandle(STD\_OUTPUT\_HANDLE),2);

gotoxy(pX, pY);

cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_";

gotoxy(pX, pY + 1);

cout << " @@@@@ ";

gotoxy(pX, pY + 2);

cout << " | | (o o) | | ";

gotoxy(pX, pY + 3);

cout << " |\_| |\_| ";

gotoxy(pX, pY + 4);

cout << " |\_\_| |\_\_| ";

gotoxy(pX, pY + 5);

cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_";

}

//this funtion is used to erase player

void erasePlayer()

{

gotoxy(pX, pY);

cout << " ";

gotoxy(pX, pY + 1);

cout << " ";

gotoxy(pX, pY + 2);

cout << " ";

gotoxy(pX, pY + 3);

cout << " ";

gotoxy(pX, pY + 4);

cout << " ";

gotoxy(pX, pY + 5);

cout << " ";

}

//this funtion is used to move player to left

void moveLeft()

{

if (getCharAtxy(pX - 1, pY) == ' ')

{

erasePlayer();

pX = pX - 1;

printPlayer();

}

}

//this funtion is used to move player to right

void moveRight()

{

if (getCharAtxy(pX + 16, pY) == ' ')

{

erasePlayer();

pX = pX + 1;

printPlayer();

}

}

//this funtion is used to move player to down

void moveDown()

{

if (getCharAtxy(pX, pY + 6) == ' ' )

{

erasePlayer();

pY = pY + 1;

printPlayer();

}

}

//this funtion is used to move player up

void moveUp()

{

if (getCharAtxy(pX, pY - 1) == ' ')

{

erasePlayer();

pY = pY - 1;

printPlayer();

}

}

//this funtion is used to get charachter

char getCharAtxy(short int x, short int y)

{

CHAR\_INFO ci;

COORD xy = {0, 0};

SMALL\_RECT rect = {x, y, x, y};

COORD coordBufSize;

coordBufSize.X = 1;

coordBufSize.Y = 1;

return ReadConsoleOutput(GetStdHandle(STD\_OUTPUT\_HANDLE), &ci, coordBufSize, xy, &rect) ? ci.Char.AsciiChar : ' ';

}

//this funtion is used to print score on board

void printscore()

{

SetConsoleTextAttribute(GetStdHandle(STD\_OUTPUT\_HANDLE),3);

gotoxy(20, 2);

cout << "Score " << score;

gotoxy(40, 2);

cout << "Eni1: " << e1score;

gotoxy(52, 2);

cout << "Eni2: " << e2score;

gotoxy(64, 2);

cout << "Eni3: " << e3score;

gotoxy(76, 2);

cout << "Eni4: " << e4score;

}

//this funtion is used to go to different axis

void gotoxy(int x, int y)

{

COORD coordinates;

coordinates.X = x;

coordinates.Y = y;

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

}

//this funtion is used to give player a health pill

void health()

{

gotoxy(hX, hY);

cout << " ";

gotoxy(hX, hY + 1);

cout << " ";

gotoxy(hX, hY + 2);

cout << " ";

gotoxy(hX, hY + 3);

cout << " @ ";

gotoxy(hX, hY + 4);

cout << " ";

}

//this funtion is used to erase health pill after touching it

void healthErase()

{

gotoxy(hX, hY);

cout << " ";

gotoxy(hX, hY + 1);

cout << " ";

gotoxy(hX, hY + 2);

cout << " ";

gotoxy(hX, hY + 3);

cout << " ";

gotoxy(hX, hY + 4);

cout << " ";

}

//this funtion is used to set fire position of player

void fire()

{

if (!bulletActive)

{

bulletX = pX + 7;

bulletY = pY - 1;

bulletActive = true;

}

}

//this funtion is used to set fire position of player

void fireR()

{

if (!bulletActiveR)

{

bulletRX = pX - 7;

bulletRY = pY + 1;

bulletActiveR = true;

}

}

//this funtion is used to move fire downwords

void moveBulletR()

{

gotoxy(bulletRX, bulletRY);

cout << " ";

bulletRY++;

if (bulletRY > 50)

{

bulletActiveR = false;

gotoxy(bulletRX, bulletRY);

cout << " ";

}

else

{

char hitChar = getCharAtxy(bulletRX, bulletRY);

if (hitChar == '#')

{

bulletActiveR = false;

gotoxy(bulletRX, bulletRY);

cout << " ";

}

else

{

gotoxy(bulletRX, bulletRY);

cout << "|";

}

}

}

//this funtion is used to move fire upwards

void moveBullet()

{

gotoxy(bulletX, bulletY);

cout << " ";

bulletY--;

if (bulletY < 2)

{

bulletActive = false;

gotoxy(bulletX, bulletY);

cout << " ";

}

else

{

char hitChar = getCharAtxy(bulletX, bulletY);

if (hitChar == '#')

{

bulletActive = false;

gotoxy(bulletX, bulletY);

cout << " ";

}

else

{

gotoxy(bulletX, bulletY);

cout << "^";

}

}

}

//this funtion is used to check collision

void checkCon()

{

if (getCharAtxy(bulletX, bulletY - 1) == '\_')

{

gotoxy(bulletX, bulletY);

cout << " ";

// cout << "Muje baja fire";

e1score -= 50;

bulletActive = false;

score += 50;

}

if (getCharAtxy(bulletX, bulletY - 1) == '\\' || getCharAtxy(bulletX, bulletY - 1) == '\*' || getCharAtxy(bulletX, bulletY - 1) == '^' || getCharAtxy(bulletX, bulletY - 1) == '/')

{

gotoxy(bulletX, bulletY);

cout << " ";

// cout << "Muje baja fire";

e2score -= 50;

bulletActive = false;

score += 50;

}

if (getCharAtxy(bulletRX, bulletRY + 1) == '\\' || getCharAtxy(bulletRX, bulletRY + 1) == 'O' || getCharAtxy(bulletRX, bulletRY + 1) == 'W' || getCharAtxy(bulletRX, bulletRY + 1) == 'L' || getCharAtxy(bulletRX, bulletRY + 1) == '/')

{

gotoxy(bulletRX, bulletRY);

cout << " ";

// cout << "Muje baja fire";

e3score -= 50;

bulletActiveR = false;

score += 50;

}

if (getCharAtxy(bulletX, bulletY - 1) == '\*')

{

gotoxy(bulletX, bulletY);

cout << " ";

// cout << "Muje baja fire";

e4score -= 50;

bulletActive = false;

score += 50;

}

}

//this funtion is used to print bullet of player

void bullet()

{

gotoxy(bulletX, bulletY);

cout << "|";

// Sleep(200);

}

//this funtion is used to print bullet of player

void bulletR()

{

gotoxy(bulletRX, bulletRY);

cout << "0";

// Sleep(200);

}

//this funtion is used to print enemy bullets

void printEnemybullets()

{

if (firstEnemybullet)

{

gotoxy(enemy1bulletX, enemy1bulletY);

cout << "i";

}

if (secondEnemybullet)

{

gotoxy(enemy2bulletX, enemy2bulletY);

cout << "o";

}

if (thirdEnemybullet)

{

gotoxy(enemy3bulletX, enemy3bulletY);

cout << "@";

}

if (forthEnemybullet)

{

gotoxy(enemy4bulletX, enemy4bulletY);

cout << "o";

}

}

//this funtion is used to set enemy bullets position

void enemyfire()

{

if (!firstEnemybullet)

{

enemy1bulletX = eX1 + 1;

enemy1bulletY = eY1 + 1;

firstEnemybullet = true;

}

if (!secondEnemybullet)

{

enemy2bulletX = eX2 + 1;

enemy2bulletY = eY2 + 1;

secondEnemybullet = true;

}

if (!thirdEnemybullet)

{

enemy3bulletX = eX3 - 1;

enemy3bulletY = eY3 - 1;

thirdEnemybullet = true;

}

if (!forthEnemybullet)

{

enemy4bulletX = eX4 + 1;

enemy4bulletY = eY4 + 1;

forthEnemybullet = true;

}

}

//this funtion is used to move enemy bullets

void moveEnemyBullet()

{

if (firstEnemybullet)

{

gotoxy(enemy1bulletX, enemy1bulletY);

cout << " ";

enemy1bulletY += 3;

if (enemy1bulletY > 40 || getCharAtxy(enemy1bulletX, enemy1bulletY) == '#' || getCharAtxy(enemy1bulletX, enemy1bulletY) == '#')

{

firstEnemybullet = false;

}

if (getCharAtxy(enemy1bulletX, enemy1bulletY) == '\_')

{

score -= 50;

}

}

if (secondEnemybullet)

{

gotoxy(enemy2bulletX, enemy2bulletY);

cout << " ";

enemy2bulletY += 3;

if (enemy2bulletY > 40 || getCharAtxy(enemy2bulletX, enemy2bulletY) == '#' || getCharAtxy(enemy2bulletX, enemy2bulletY) == '#')

{

secondEnemybullet = false;

}

if (getCharAtxy(enemy2bulletX, enemy2bulletY) - 1 == '\_')

{

score -= 50;

}

}

if (thirdEnemybullet)

{

gotoxy(enemy3bulletX, enemy3bulletY);

cout << " ";

enemy3bulletY -= 3;

if (enemy3bulletY < 2 || getCharAtxy(enemy3bulletX, enemy3bulletY) == '#' || getCharAtxy(enemy3bulletX, enemy3bulletY) == '#')

{

thirdEnemybullet = false;

}

if (getCharAtxy(enemy3bulletX, enemy3bulletY) == '\_')

{

score -= 50;

}

}

if (forthEnemybullet)

{

gotoxy(enemy4bulletX, enemy4bulletY);

cout << " ";

enemy4bulletY += 3;

if (enemy4bulletY > 40 || getCharAtxy(enemy4bulletX, enemy4bulletY) == '#' || getCharAtxy(enemy4bulletX, enemy4bulletY) == '#')

{

forthEnemybullet = false;

}

if (getCharAtxy(enemy4bulletX, enemy4bulletY) == '\_')

{

score -= 50;

}

}

}

//this funtion is used to print header of game

void header()

{

cout << R"(

\_\_\_\_\_ \_\_\_ \_

| \_\_\_\_|\_ \_\_\_\_\_ \_ \_\_ \_ \_ / \_ \ \_ \_\_ \_\_\_ (\_)\_\_\_

| \_| \ \ / / \_ \ '\_\_| | | | | | | '\_ \ / \_ \ | / \_\_|

| |\_\_\_ \ V / \_\_/ | | |\_| | |\_| | | | | \_\_/ | \\_\_ \

|\_\_\_\_\_| \\_/ \\_\_\_|\_| \\_\_, |\\_\_\_/|\_| |\_|\\_\_\_| |\_|\_\_\_/

|\_ \_|\_ \_\_ \_\_\_ \_ \_\_ |\_\_\_/\_\_\_| |\_ \_\_\_ \_ \_\_| | | |

| || '\_ ` \_ \| '\_ \ / \_ \/ \_\_| \_\_/ \_ \ '\_\_| | | |

| || | | | | | |\_) | (\_) \\_\_ \ || \_\_/ | |\_|\_|\_|

|\_\_\_|\_| |\_| |\_| .\_\_/ \\_\_\_/|\_\_\_/\\_\_\\_\_\_|\_| (\_|\_|\_)

|\_| )"

<< endl;

cout << "Press Any key to Start: ";

getch();

}

//this funtion is used to erase player up bullet

void erase()

{

gotoxy(bulletX, bulletY);

cout << " ";

}

//this funtion is used to erase players down bullet

void eraseR()

{

gotoxy(bulletRX, bulletRY);

cout << " ";

}

//this funtion is used to print msg after losing game

void lose()

{

system("cls");

cout << R"(

\_\_\_\_\_ \_ \_ \_ \_ \_

|\_ \_| \_\_ \_ \_ / \ \_\_ \_ \_\_ \_(\_)\_ \_\_ | | | |

| || '\_\_| | | | / \_ \ / \_` |/ \_` | | '\_ \| | | |

| || | | |\_| | / \_\_\_ \ (\_| | (\_| | | | | |\_|\_|\_|

\_\_|\_||\_| \\_\_, | /\_/ \\_\\_\_, |\\_\_,\_|\_|\_| |\_(\_|\_|\_)

\ \ / /\_\_ |\_\_\_/ | | \_|\_\_\_/\_\_ \_\_\_| | | |

\ V / \_ \| | | | | | / \_ \/ \_\_|/ \_ \ | | |

| | (\_) | |\_| | | |\_\_| (\_) \\_\_ \ \_\_/\_|\_|\_|

|\_|\\_\_\_/ \\_\_,\_| |\_\_\_\_\_\\_\_\_/|\_\_\_/\\_\_\_(\_|\_|\_) )";

}

//this funtion is used to print msg after winning game

void win()

{

system("cls");

cout << R"(

\_\_\_\_ \_ \_ \_ \_ \_ \_

/ \_\_\_|\_\_\_ \_ \_\_ \_\_ \_ \_ \_\_ \_\_ \_| |\_ \_ \_| | \_\_ \_| |\_(\_) \_\_\_ \_ \_\_ \_\_\_| | |

| | / \_ \| '\_ \ / \_` | '\_\_/ \_` | \_\_| | | | |/ \_` | \_\_| |/ \_ \| '\_ \/ \_\_| | |

| |\_\_| (\_) | | | | (\_| | | | (\_| | |\_| |\_| | | (\_| | |\_| | (\_) | | | \\_\_ \\_|\_|

\\_\_\_\_\\_\_\_/|\_| |\_|\\_\_, |\_| \\_\_,\_|\\_\_|\\_\_,\_|\_|\\_\_,\_|\\_\_|\_|\\_\_\_/|\_| |\_|\_\_\_(\_|\_)

\_\_ \_\_ |\_\_\_/ \_\_\_ \_ \_ \_

\ \ / /\_\_ \_ \_ \ \ / (\_)\_ \_\_ | | | |

\ V / \_ \| | | | \ \ /\ / /| | '\_ \| | | |

| | (\_) | |\_| | \ V V / | | | | |\_|\_|\_|

|\_|\\_\_\_/ \\_\_,\_| \\_/\\_/ |\_|\_| |\_(\_|\_|\_)

)";

}