**The Pirates**



Session: 2022 – 2026

**Submitted by:**

Ayesha Khalid 2022-CS-153

**Supervised by:**

PROF.AWAIS

Department of Computer Science

**University of Engineering and Technology**

**Lahore Pakistan**

Table of Contents

[ Short Description and Story Writing of your Game 3](#_Toc130229191)

[ Game Characters Description 3](#_Toc130229194)

[Player 3](#_Toc130229195)

[Enemies 3](#_Toc130229196)

[ Game Objects Description 4](#_Toc130229197)

[PEOPLE: 4](#_Toc130229198)

[WALLS: 4](#_Toc130229199)

[ Rules & Interactions 4](#_Toc130229200)

[ Goal of the Game 4](#_Toc130229201)

[ The goal of the game is to hit as many people as Thomas can to get the scores while avoiding the police and cross the city. 4](#_Toc130229202)

[ Wireframes of the Game (Screenshots of the Menus of the Game) 4](#_Toc130229203)

[ Data Structures (2D Arrays) 4](#_Toc130229204)

[ Function Prototypes 4](#_Toc130229205)

[ Complete Code 4](#_Toc130229206)

# Short Description and Story Writing of your Game

## Thomas is thin looking fragile boy who is very optimistic and has an old mother whom he has to feed. He used to do a job in a small coffee shop. Unfortunately, he lost his job because customers complained against him that he is never clean. Now he is jobless and living from hand to mouth. So now Thomas has to rob a bank. Thomas will get success in his plan but it will be his bad luck that a random guy who will be watching all this scene from a distance, will call the police.

## Then Thomas will be running to save his life. He won't even surrender because he knows that if police took him away then they will put him in lock for days and his old mother will die with hunger. So at any cost he will not want to get caught. On his way he will collide with many other people which are shown by asterisks in game and steal their money as well in order to increase his money which is shown as score. There will be three police cars which will chase Thomas. Thomas is a clever boy and a fast runner so he knows that he will protect himself and the money from the police. Thomas has a gun and he will be able to fire at the police as well. He always loved to do the adventures and finally he will be fulfilling his desire by taking more and more risks. His mission will be to cross all the police cars in order to run away from the city. There will be a main gate at the other end of the from where he will escape. There will be only hundred healths for Thomas and after that Thomas will die and game will be over.

# Game Characters Description

## Player

There is one human player in the Game.

**Thomas:**

Thomas is the main character of this game and is known for his smartness. He is adventurous and loves to take risks. Thomas is a brave boy who got distracted because of poverty. Thomas is a brave, determined, and has a never-say-die-spirit.

## Enemies

There are 3 enemies in the game.

**Highway Police:**

Highway police is capable of moving in four directions. It will be moving randomly and then at the end will start to roam aroung the main door of the city in order to stop theh Thomas to escape.

**Motor Police:**

Motor police can chase thomas. If it hits the Thomas then Thomas will lose his one health cell.

**Local Police:**

Local police can only move in vertical direction and can fire only from the vertical sides. If the fire hits the Thomas then Thomas will lose his one health cell.

# Game Objects Description

Following are the Objects in the Game

## PEOPLE:

Thomas will hit the people passing by and will get a score because hitting people means Thomas has robbed them too.

## WALLS:

Walls are the barriers in the game which the Thomas and Police cannot cross.

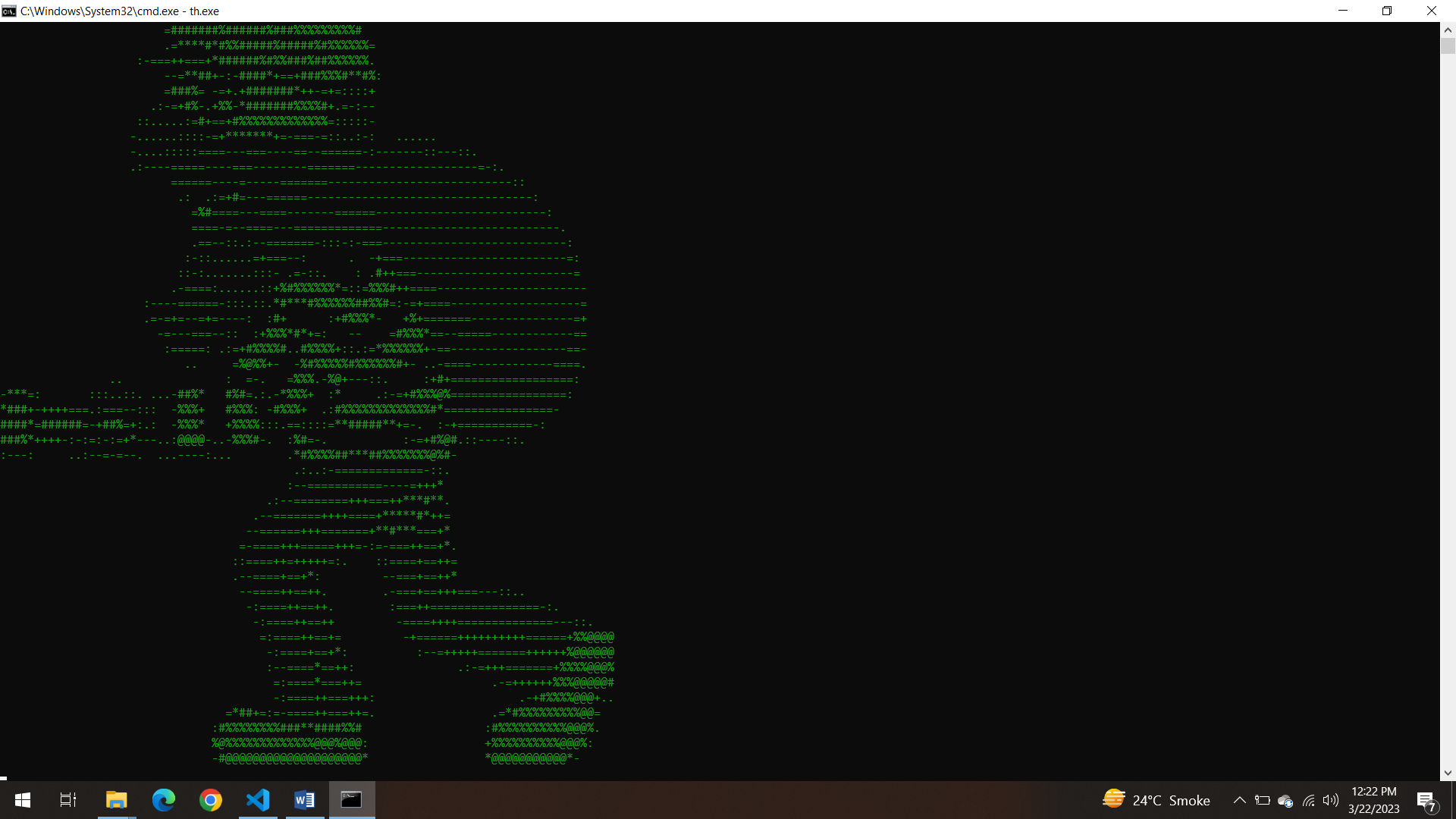
# Rules & Interactions

Thomas can hit the people in order to get the score .Thomas loses his health when got fired by the police. If Thomas hits the police he can kill them in 3 hits.

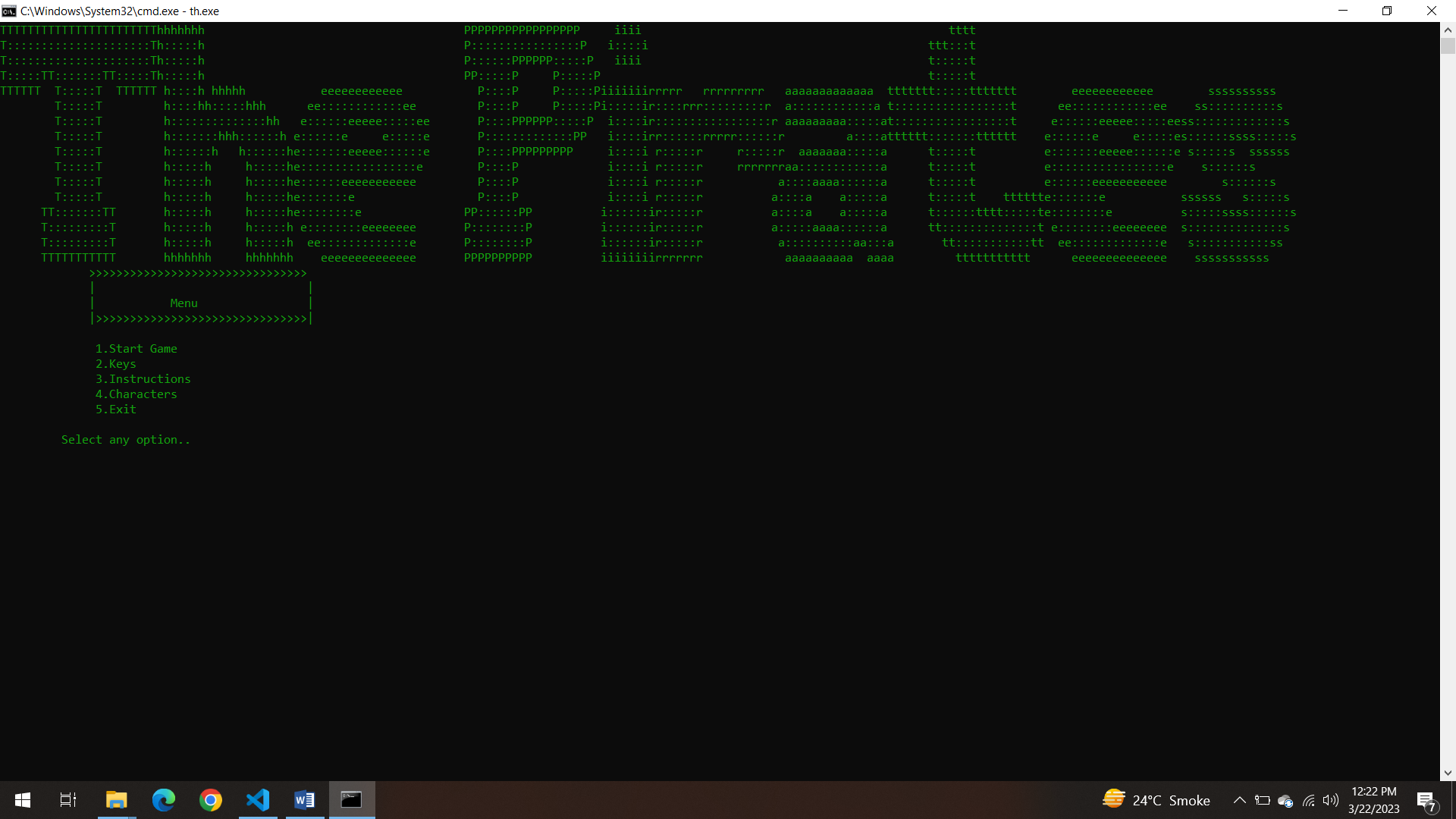
# Goal of the Game

# The goal of the game is to hit as many people as Thomas can to get the scores while avoiding the police and cross the city.

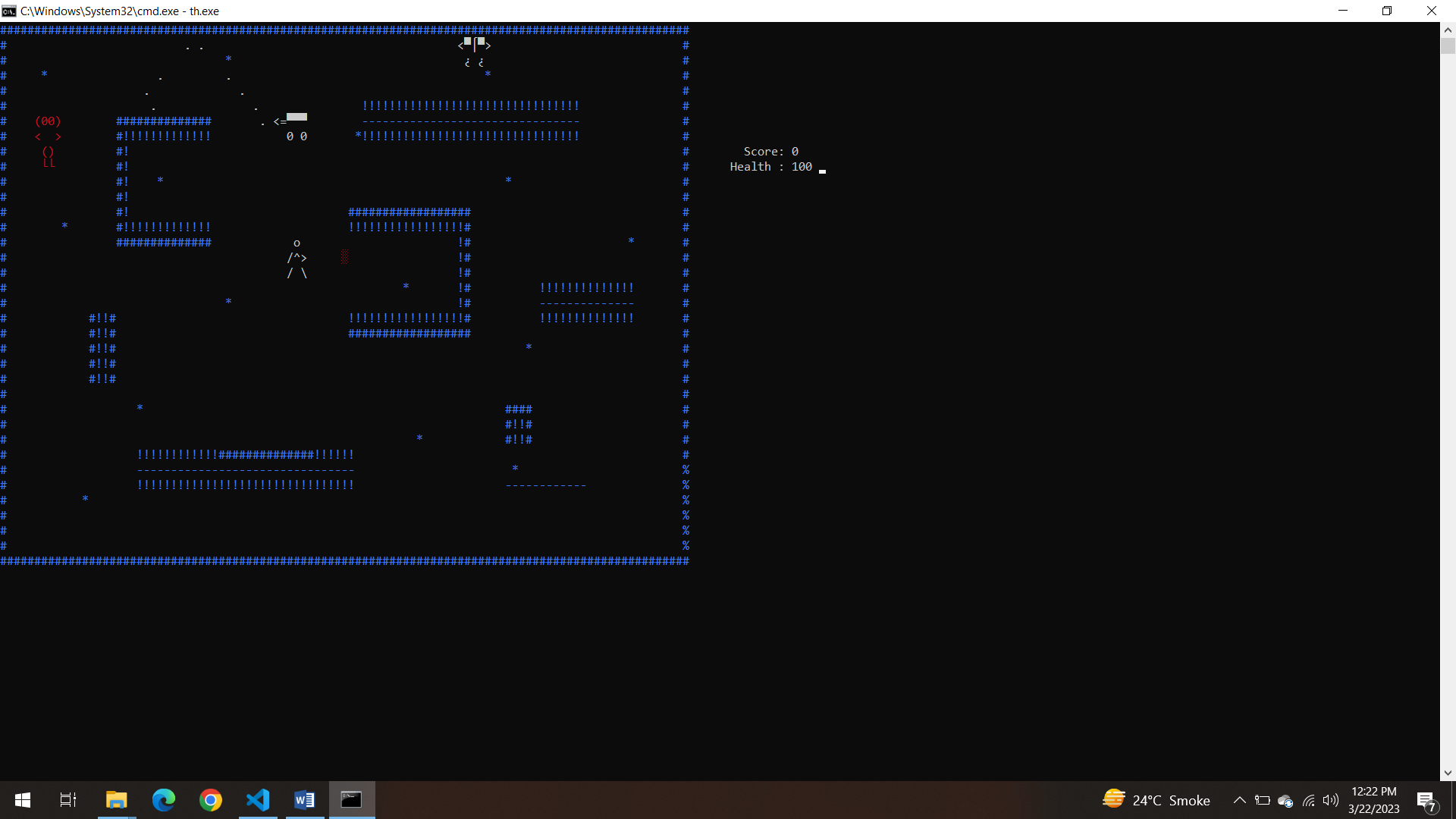
# Wireframes of the Game (Screenshots of the Menus of the Game)



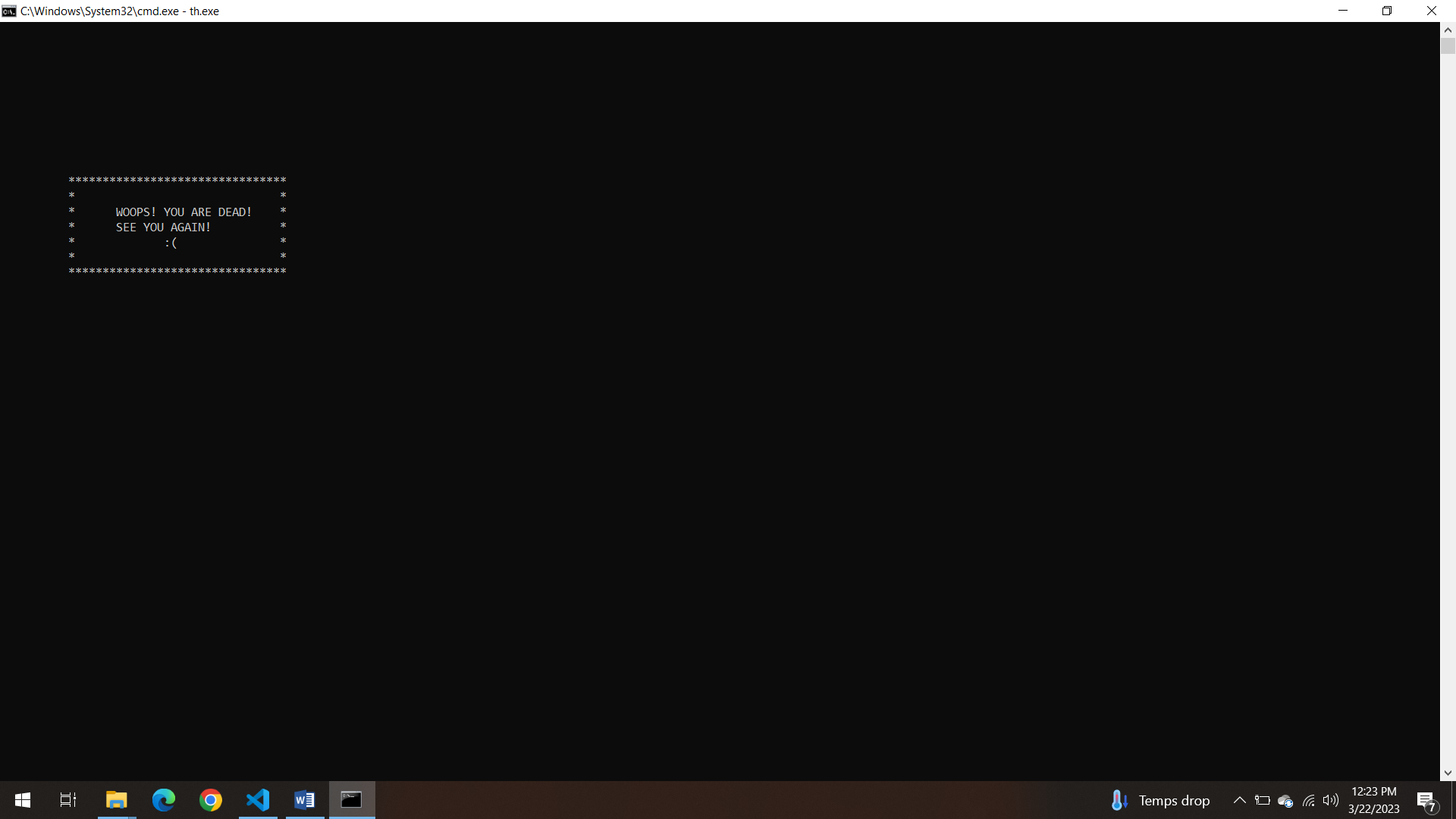
# Figure1:logo



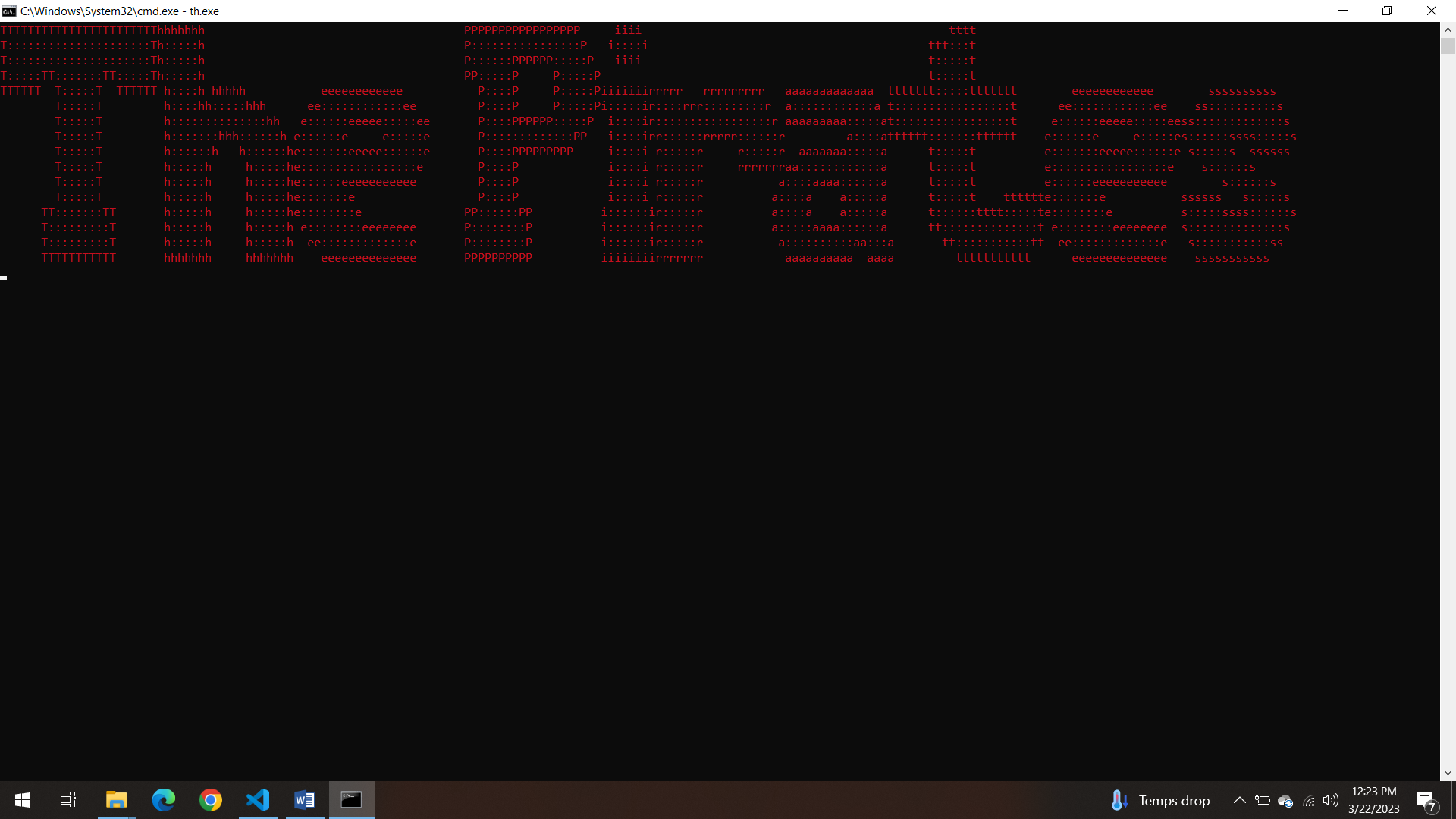
# Figure2:main menu

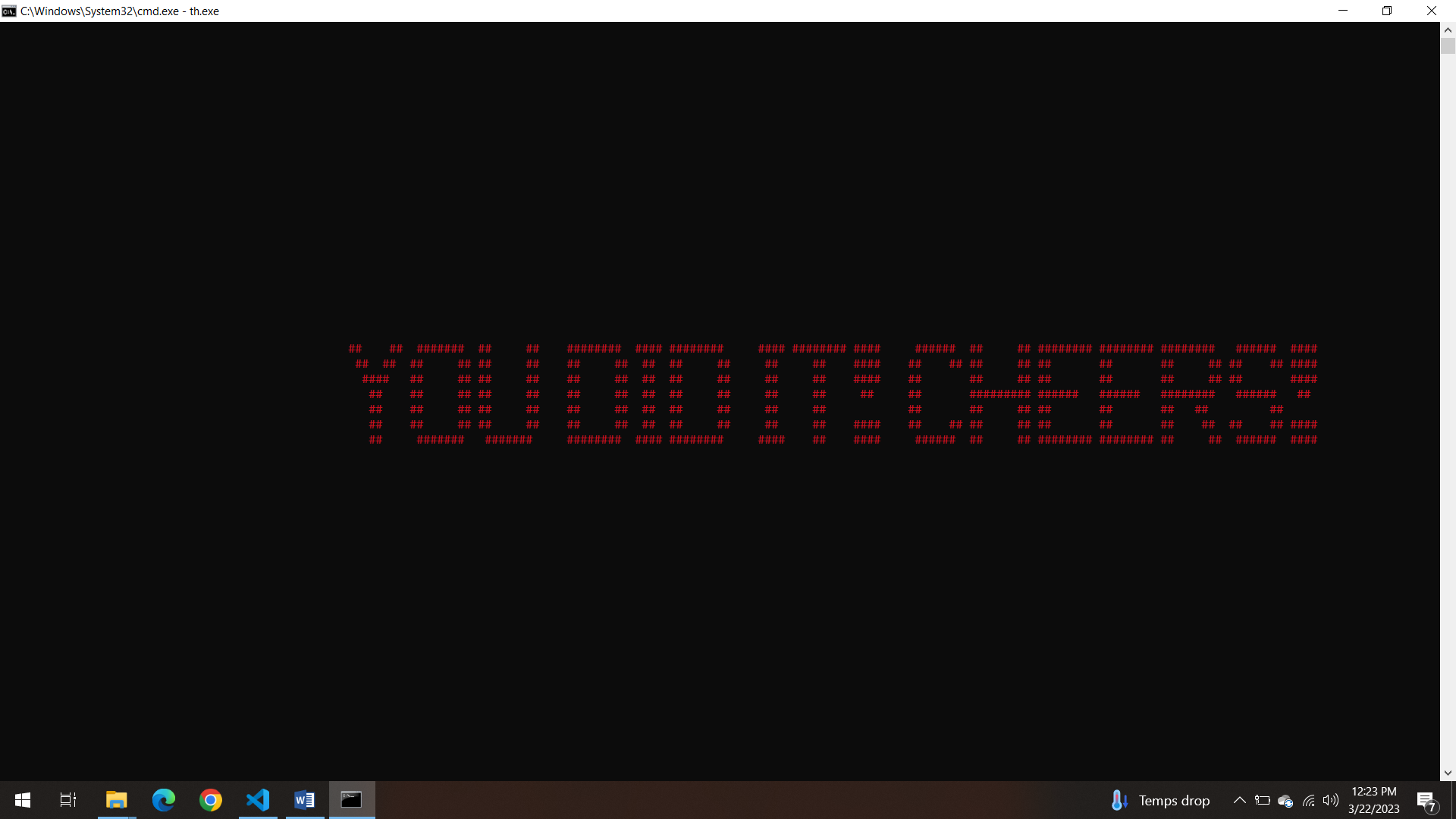


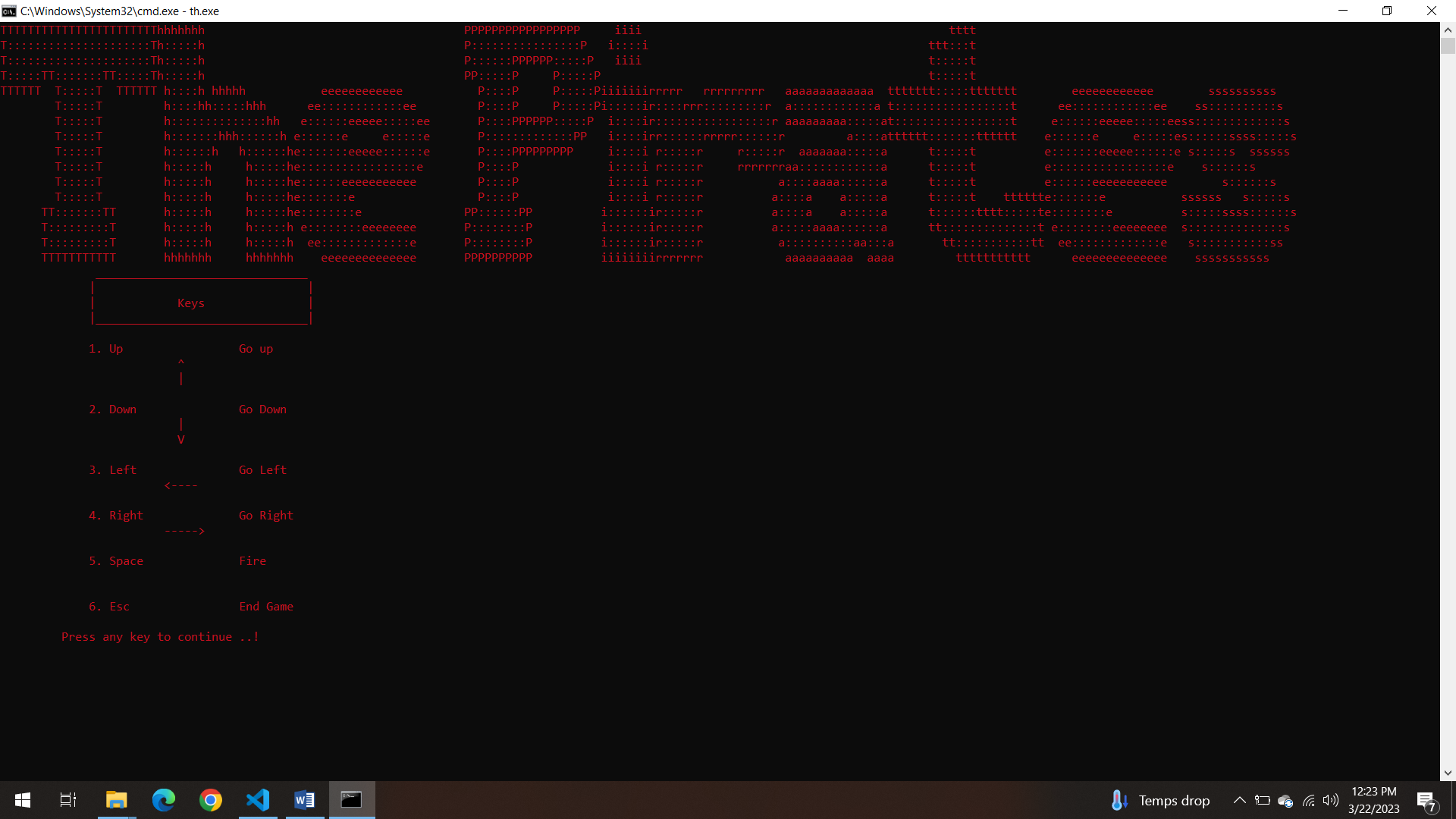
# Figure3:Start game



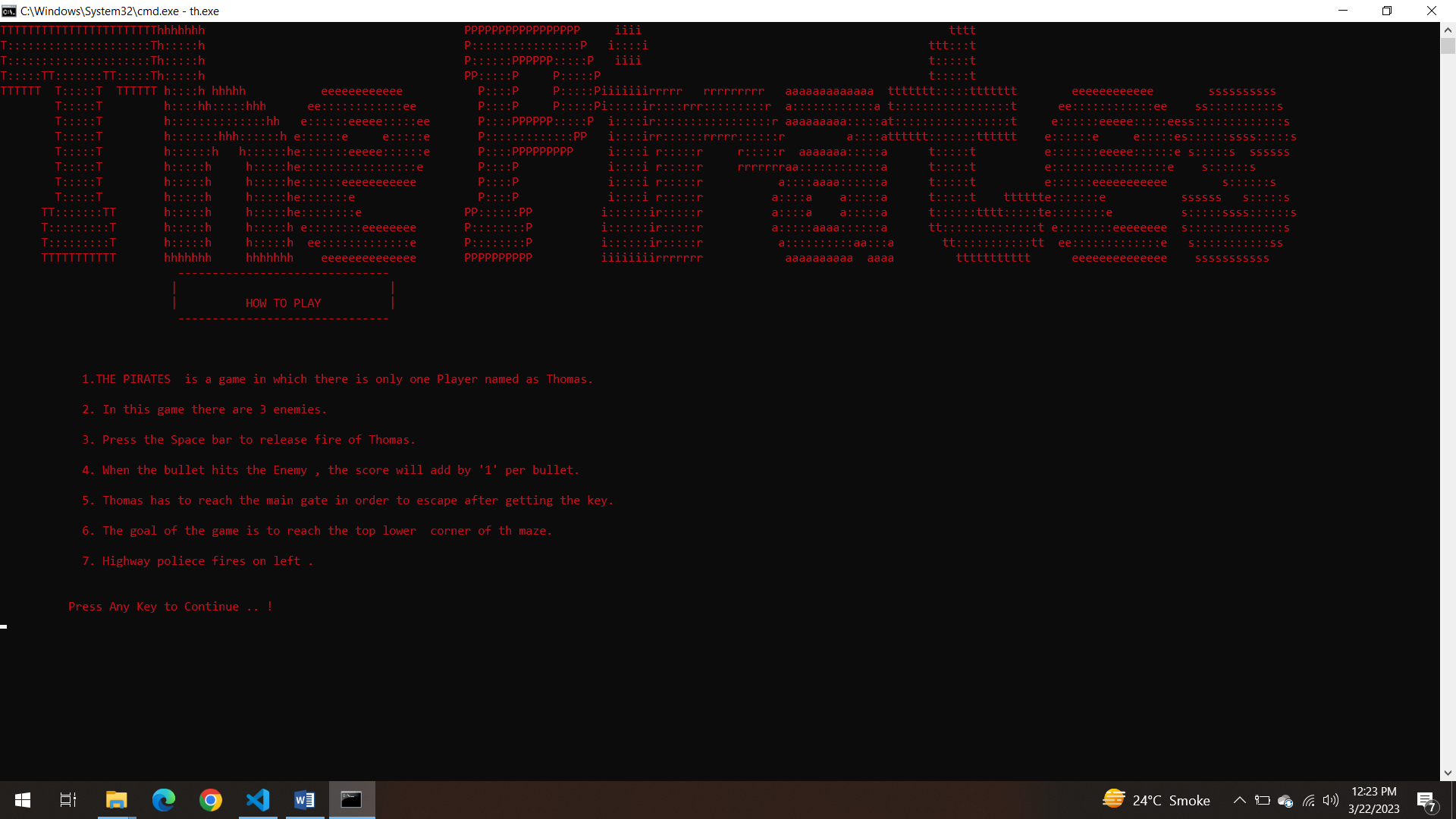
# Figure4:Game Over







# Figure5:Keys



# Figure6: Instruction

# Data Structures (2D Arrays)

char printMaze[36][103]; // array to print the maze

char mainCharacter1[4][4] = {{'(', '0', '0', ')'},

{'<', ' ', ' ', '>'},

{' ', '(', ')', ' '},

{' ', char(192), char(192), ' '}}; // Array to print the main character

int bulletX[100]; // X coordinates of bullet of main character

int bulletY[100]; // Y coordinates of bullet of main character

bool isBulletActive[100]; // ftn to make bullet active of main character

char Enemy1[2][5] = {{'<', '=', char(boxx), char(boxx), char(boxx)},

{' ', ' ', '0', ' ', '0'}}; // array to print 1st enemy

int bulletXenemy[100]; // X coordinates of bullet of 1st enemy

int bulletYenemy[100]; // Y coordinates of bullet of 1st enemy

char Enemy2[3][3] = {{' ', 'o', ' '},

{'/', '^', '>'},

{'/', ' ', '\\'}}; // array to print enemy 2

char Enemy3[2][5] = {{'<', char(boxx), char(p), char(boxx), '>'},

{' ', char(questionMark), ' ', char(questionMark), ' '}}; // array to print enemy 3

char escapeKey1[1][1] = {{char(176)}};

# Function Prototypes

void gotoxy(int x, int y);

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

void frontPage();//ftn to print front page

void keys(); //ftn to ptint ketys

int menu();//ftn to print menu

void characters();//ftn to print characters

void clearScreen();//ftn to clear screen

void instruction();////ftn for instructions of the gamew

void loadMaze(); // ftn to load the maze from file

void printMainCharacter(); // ftn to print main character

void eraseMainCharacter(); // ftn to erase main character

void moveMainCharDown(); // ftn to move main character down

void moveMainCharUp(); // ftn to move main character up

void moveMainCharRight(); // ftn to move main character right

void moveMainCharLeft(); // ftn to move main character left

void moveBullet(); // ftn to move bullet of main character

void generateBullet(); // ftn to generate bullet of main character

void removeBulletFromArray(int index); // ftn to remove bullet of main character from main

void printBullet(int x, int y); // ftn to print bullet of main character

void eraseBullet(int x, int y); // ftn to erase bullet of main character

// details of first enemy

void printEnemy1(); // array to print first enemy

void eraseEnemy1();//ftn to print enemy1

void moveEnemy1(); // ftn to move enemy 1

void moveEnemyDown(); // ftn to move 1st enemy down

void moveEnemyUp(); // ftn to move 1st enemy up

void generateBulletenemy(); // ftn to generate bullet of 1st enemy

void printBulletenemy(int x, int y); // ftn to print bullet of 1st enemy

void eraseBulletenemy(int x, int y); // ftn to erase bullet of 1st enemy

void removeBulletFromArrayenemy(int index); // ftn to remove bullet of 1st enemy

void moveBulletEnemy(); // ftn to move bullet of 1st enemy

void eraseEnemy2(); // ftn to erase enemy 2

void chaseEnemy2(); // ftn to chase enemy 2

void eraseEnemy2(); // ftn to erase enemy 2

void chaseEnemy2(); // ftn to chase enemy 2

void printEnemy3(); // ftn to print enemy 3

void eraseEnemy3(); // ftn to erase enemy 3

void moveMainEnemyLeft(); // ftn to move main enemy down

void moveMainEnemyRight(); // ftn to move main enemy up

void moveMainEnemyUp(); // ftn to move main enemy left

void moveMainEnemyDown(); // ftn to move main enemy right

void moveEnemy3(); // ftn to move enemy 3

void printScore(); // ftn to print score of main character

void start(); //ftn to start rthe game

void topHeader(); //ftn to print the top header

void storeData();//ftn to store the data

void loadData();//ftn to load the data

string getField(string record, int field);

void printdoor(); // ftn to print the exit door

# Complete Code

#include <iostream>

#include <windows.h>

#include <sstream>

#include <fstream>

#include <string>

#include <conio.h>

using namespace std;

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

void gotoxy(int x, int y);

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

void frontPage(); // ftn to print front page

void keys(); // ftn to ptint ketys

int menu(); // ftn to print menu

void characters(); // ftn to print characters

void clearScreen(); // ftn to clear screen

void instruction(); ////ftn for instructions of the gamew

char printMaze[36][103]; // array to print the maze

void loadMaze(); // ftn to load the maze from file

void printMainCharacter(); // ftn to print main character

void eraseMainCharacter(); // ftn to erase main character

char mainCharacter1[4][4] = {{'(', '0', '0', ')'},

{'<', ' ', ' ', '>'},

{' ', '(', ')', ' '},

{' ', char(192), char(192), ' '}}; // Array to print the main character

int mainX = 5; // X coordinate of main character

int mainY = 6; // Y coordinate of main character

void moveMainCharDown(); // ftn to move main character down

void moveMainCharUp(); // ftn to move main character up

void moveMainCharRight(); // ftn to move main character right

void moveMainCharLeft(); // ftn to move main character left

void moveBullet(); // ftn to move bullet of main character

void generateBullet(); // ftn to generate bullet of main character

void removeBulletFromArray(int index); // ftn to remove bullet of main character from main

void printBullet(int x, int y); // ftn to print bullet of main character

void eraseBullet(int x, int y); // ftn to erase bullet of main character

int bulletX[100]; // X coordinates of bullet of main character

int bulletY[100]; // Y coordinates of bullet of main character

bool isBulletActive[100]; // ftn to make bullet active of main character

int bulletCount = 0; // bullet counter for main character

// details of first enemy

void printEnemy1(); // array to print first enemy

void eraseEnemy1(); // ftn to print enemy1

int boxx = 223;

char Enemy1[2][5] = {{'<', '=', char(boxx), char(boxx), char(boxx)},

{' ', ' ', '0', ' ', '0'}}; // array to print 1st enemy

int enemy1X = 40; // x coordinates of enemy 1

int enemy1Y = 23; // y coordinate of enemy 1

string enemyDirection = "right"; // a variable to declare the direction

string dir = "up"; // a variable to declare the direction

void moveEnemy1(); // ftn to move enemy 1

void moveEnemyDown(); // ftn to move 1st enemy down

void moveEnemyUp(); // ftn to move 1st enemy up

int bulletXenemy[100]; // X coordinates of bullet of 1st enemy

int bulletYenemy[100]; // Y coordinates of bullet of 1st enemy

int bulletCountenemy = 0; // bullet counter for 1st enemy

void generateBulletenemy(); // ftn to generate bullet of 1st enemy

void printBulletenemy(int x, int y); // ftn to print bullet of 1st enemy

void eraseBulletenemy(int x, int y); // ftn to erase bullet of 1st enemy

void removeBulletFromArrayenemy(int index); // ftn to remove bullet of 1st enemy

void moveBulletEnemy(); // ftn to move bullet of 1st enemy

// enemy2 detail

void printEnemy2(int x, int y); // ftn to print enemy 2

char Enemy2[3][3] = {{' ', 'o', ' '},

{'/', '^', '>'},

{'/', ' ', '\\'}}; // array to print enemy 2

int enemy2X = 70; // x coordinates of enemy 2

int enemy2Y = 31; // y coordinates of enemy 2

void eraseEnemy2(); // ftn to erase enemy 2

void chaseEnemy2(); // ftn to chase enemy 2

string destination = "up"; // a variable to declare the direction

void printEnemy3(); // ftn to print enemy 3

void eraseEnemy3(); // ftn to erase enemy 3

int p = 244;

int questionMark = 168;

char Enemy3[2][5] = {{'<', char(boxx), char(p), char(boxx), '>'},

{' ', char(questionMark), ' ', char(questionMark), ' '}}; // array to print enemy 3

int enemy3X = 90; // x coordinates of enemy 3

int enemy3Y = 5; // y coordinates of enemy 3

void moveMainEnemyLeft(); // ftn to move main enemy down

void moveMainEnemyRight(); // ftn to move main enemy up

void moveMainEnemyUp(); // ftn to move main enemy left

void moveMainEnemyDown(); // ftn to move main enemy right

void moveEnemy3(); // ftn to move enemy 3

int timer = 0; // timer to move all the three enemies

int score = 0; // variable to add the score

int health = 100; // a variable for health

void addScore(); // ftn to add the score

void printScore(); // ftn to print score of main character

void start(); // ftn to start rthe game

void topHeader(); // ftn to print the top header

bool gameRunning = true;

int escapeKeyX = 50;

int escapeKeyY = 15;

void Exit(); // ftn to exit the game

char escapeKey1[1][1] = {{char(176)}};

void storeData(); // ftn to store the data

void loadData(); // ftn to load the data

string getField(string record, int field);

void printdoor(); // ftn to print the exit door

int printdoorX = 50;

int printdoorY = 15;

main()

{

// loadData();

// storeData();

system("cls");

frontPage();

Sleep(1000);

getch();

system("cls");

int option = 0;

while (option != 5)

{

system("cls");

topHeader();

option = menu();

if (option == 1)

{

clearScreen();

topHeader();

system("cls");

gameRunning = true;

health = 100;

score = 0;

start();

}

else if (option == 2)

{

clearScreen();

topHeader();

keys();

}

else if (option == 3)

{

clearScreen();

topHeader();

instruction();

}

else if (option == 4)

{

clearScreen();

topHeader();

characters();

}

}

}

void start()

{

system("cls");

SetConsoleTextAttribute(acolor, 3);

loadMaze();

printMainCharacter();

SetConsoleTextAttribute(acolor, 4);

printEnemy1();

printEnemy2(enemy2X, enemy2Y);

printEnemy3();

gotoxy(escapeKeyX, escapeKeyY);

cout << char(176);

while (gameRunning)

{

generateBulletenemy();

printScore();

if (GetAsyncKeyState(VK\_LEFT))

{

moveMainCharLeft();

}

if (GetAsyncKeyState(VK\_RIGHT))

{

moveMainCharRight();

}

if (GetAsyncKeyState(VK\_UP))

{

moveMainCharUp();

}

if (GetAsyncKeyState(VK\_DOWN))

{

moveMainCharDown();

}

if (GetAsyncKeyState(VK\_SPACE))

{

generateBullet();

}

if ((mainX == escapeKeyX || mainX + 1 == escapeKeyX || mainX - 1 == escapeKeyX) && (mainY == escapeKeyY || mainY + 1 == escapeKeyY || mainY + 2 == escapeKeyY || mainY + 3 == escapeKeyY))

{

for (int idx = 0; idx < 3; idx++)

{

gotoxy(94, 30 + idx);

for (int idx = 0; idx < 3; idx++)

{

cout << char(178);

}

}

}

if ((mainX == 94 || mainX + 1 == 94 || mainX + 2 == 94) || (mainY == 94 || mainY + 1 == 94 || mainY + 2 == 94 || mainY + 3 == 94))

{

system("cls");

gotoxy(50, 21);

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

gotoxy(50, 22);

Sleep(300);

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

gotoxy(50, 23);

Sleep(300);

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

gotoxy(50, 24);

Sleep(300);

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

gotoxy(50, 25);

Sleep(300);

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

gotoxy(50, 26);

Sleep(300);

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

gotoxy(50, 27);

Sleep(300);

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

Sleep(300);

Sleep(1000);

break;

}

if (timer == 2)

{

moveEnemy1();

chaseEnemy2();

moveEnemy3();

timer = 0;

}

moveBullet();

moveBulletEnemy();

gotoxy(107, 9);

cout << "Health : " << health << " ";

if (health <= 0)

{

system("cls");

gotoxy(10, 10);

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

Sleep(100);

gotoxy(10, 11);

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

Sleep(100);

gotoxy(10, 12);

cout << "\* WOOPS! YOU ARE DEAD! \*" << endl;

Sleep(100);

gotoxy(10, 13);

cout << "\* SEE YOU AGAIN! \*" << endl;

Sleep(100);

gotoxy(10, 14);

cout << "\* :( \*" << endl;

Sleep(100);

gotoxy(10, 15);

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

Sleep(100);

gotoxy(10, 16);

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

storeData();

gameRunning = false;

Sleep(5000);

// break;

}

timer++;

Sleep(30);

}

}

void topHeader()

{

cout << "TTTTTTTTTTTTTTTTTTTTTTThhhhhhh PPPPPPPPPPPPPPPPP iiii tttt " << endl;

cout << "T:::::::::::::::::::::Th:::::h P::::::::::::::::P i::::i ttt:::t " << endl;

cout << "T:::::::::::::::::::::Th:::::h P::::::PPPPPP:::::P iiii t:::::t " << endl;

cout << "T:::::TT:::::::TT:::::Th:::::h PP:::::P P:::::P t:::::t " << endl;

cout << "TTTTTT T:::::T TTTTTT h::::h hhhhh eeeeeeeeeeee P::::P P:::::Piiiiiiirrrrr rrrrrrrrr aaaaaaaaaaaaa ttttttt:::::ttttttt eeeeeeeeeeee ssssssssss " << endl;

cout << " T:::::T h::::hh:::::hhh ee::::::::::::ee P::::P P:::::Pi:::::ir::::rrr:::::::::r a::::::::::::a t:::::::::::::::::t ee::::::::::::ee ss::::::::::s " << endl;

cout << " T:::::T h::::::::::::::hh e::::::eeeee:::::ee P::::PPPPPP:::::P i::::ir:::::::::::::::::r aaaaaaaaa:::::at:::::::::::::::::t e::::::eeeee:::::eess:::::::::::::s " << endl;

cout << " T:::::T h:::::::hhh::::::h e::::::e e:::::e P:::::::::::::PP i::::irr::::::rrrrr::::::r a::::atttttt:::::::tttttt e::::::e e:::::es::::::ssss:::::s" << endl;

cout << " T:::::T h::::::h h::::::he:::::::eeeee::::::e P::::PPPPPPPPP i::::i r:::::r r:::::r aaaaaaa:::::a t:::::t e:::::::eeeee::::::e s:::::s ssssss " << endl;

cout << " T:::::T h:::::h h:::::he:::::::::::::::::e P::::P i::::i r:::::r rrrrrrraa::::::::::::a t:::::t e:::::::::::::::::e s::::::s " << endl;

cout << " T:::::T h:::::h h:::::he::::::eeeeeeeeeee P::::P i::::i r:::::r a::::aaaa::::::a t:::::t e::::::eeeeeeeeeee s::::::s " << endl;

cout << " T:::::T h:::::h h:::::he:::::::e P::::P i::::i r:::::r a::::a a:::::a t:::::t tttttte:::::::e ssssss s:::::s " << endl;

cout << " TT:::::::TT h:::::h h:::::he::::::::e PP::::::PP i::::::ir:::::r a::::a a:::::a t::::::tttt:::::te::::::::e s:::::ssss::::::s" << endl;

cout << " T:::::::::T h:::::h h:::::h e::::::::eeeeeeee P::::::::P i::::::ir:::::r a:::::aaaa::::::a tt::::::::::::::t e::::::::eeeeeeee s::::::::::::::s " << endl;

cout << " T:::::::::T h:::::h h:::::h ee:::::::::::::e P::::::::P i::::::ir:::::r a::::::::::aa:::a tt:::::::::::tt ee:::::::::::::e s:::::::::::ss " << endl;

cout << " TTTTTTTTTTT hhhhhhh hhhhhhh eeeeeeeeeeeeee PPPPPPPPPP iiiiiiiirrrrrrr aaaaaaaaaa aaaa ttttttttttt eeeeeeeeeeeeee sssssssssss " << endl;

Sleep(200);

}

void frontPage()

{

SetConsoleTextAttribute(acolor, 2);

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;

cout << " =\*##+=:=-====++===++=. .=\*#%%%%%%%%%@@= " << endl;

cout << " :#%%%%%%%%###\*\*####%%# :#%%%%%%%%%%@@@%. " << endl;

cout << " %@%%%%%%%%%%%%%@@@%@@@: +%%%%%%%%%%@@@%: " << endl;

cout << " -#@@@@@@@@@@@@@@@@@@@@\* \*@@@@@@@@@@@\*- " << endl;

}

void loadMaze()

{

SetConsoleTextAttribute(acolor, 9);

fstream yourfile;

string line;

int rows = 0;

yourfile.open("print.txt", ios::in);

while (getline(yourfile, line))

{

for (int cols = 0; cols < 102; cols++)

{

printMaze[rows][cols] = line[cols];

}

rows++;

}

yourfile.close();

for (int i = 0; i < 36; i++)

{

for (int j = 0; j < 102; j++)

{

cout << printMaze[i][j];

}

cout << endl;

}

}

void instruction()

{

cout << "\t"

" ------------------------------- "

<< endl;

cout << "\t"

" | |"

<< endl;

cout << "\t"

" | HOW TO PLAY |"

<< endl;

cout << "\t"

" ------------------------------- "

<< endl;

cout << endl

<< endl

<< endl;

cout << "\t"

" 1.THE PIRATES is a game in which there is only one Player named as Thomas. "

<< endl

<< endl;

cout << "\t"

" 2. In this game there are 3 enemies. "

<< endl

<< endl;

cout << "\t"

" 3. Press the Space bar to release fire of Thomas. "

<< endl

<< endl;

cout << "\t"

" 4. When the bullet hits the Enemy , the score will add by '1' per bullet. "

<< endl

<< endl;

cout << "\t"

" 5. Thomas has to reach the main gate in order to escape after getting the key. "

<< endl

<< endl;

cout << "\t"

" 6. The goal of the game is to reach the top lower corner of th maze. "

<< endl

<< endl;

cout << "\t"

" 7. Highway poliece fires on left . "

<< endl

<< endl;

cout << endl;

cout << "\t"

" Press Any Key to Continue .. !"

<< endl;

getch();

}

void characters()

{

system("cls");

cout << "Thomas :)" << endl;

for (int i = 0; i < 4; i++)

{

for (int j = 0; j < 4; j++)

{

cout << mainCharacter1[i][j];

}

cout << endl;

}

cout << endl;

cout << endl;

cout << "Local Police :)>" << endl;

for (int i = 0; i < 2; i++)

{

for (int j = 0; j < 5; j++)

{

cout << Enemy1[i][j];

}

cout << endl;

}

cout << endl;

cout << endl;

cout << "Motor Police: :)>" << endl;

for (int i = 0; i < 3; i++)

{

for (int j = 0; j < 3; j++)

{

cout << Enemy2[i][j];

}

cout << endl;

}

cout << endl;

cout << endl;

cout << "Highway Police: :)>" << endl;

for (int i = 0; i < 2; i++)

{

for (int j = 0; j < 5; j++)

{

cout << Enemy3[i][j];

}

cout << endl;

cout << endl;

getch();

}

}

int menu()

{

int option;

cout << "\t"

" >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>> "

<< endl;

cout << "\t"

" | |"

<< endl;

cout << "\t"

" | Menu |"

<< endl;

cout << "\t"

" |>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>|"

<< endl;

cout << "\t"

" "

<< endl;

cout << "\t"

" 1.Start Game "

<< endl;

cout << "\t"

" 2.Keys "

<< endl;

cout << "\t"

" 3.Instructions "

<< endl;

cout << "\t"

" 4.Characters "

<< endl;

cout << "\t"

" 5.Exit "

<< endl;

cout << endl;

cout << "\t"

" Select any option..";

cin >> option;

return option;

}

void keys()

{

cout << "\t"

" \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ "

<< endl;

cout << "\t"

" | |"

<< endl;

cout << "\t"

" | Keys |"

<< endl;

cout << "\t"

" |\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|"

<< endl;

cout << endl;

cout << "\t"

" 1. Up Go up "

<< endl;

cout << "\t"

" ^ "

<< endl;

cout << "\t"

" | "

<< endl;

cout << endl;

cout << "\t"

" 2. Down Go Down "

<< endl;

cout << "\t"

" | "

<< endl;

cout << "\t"

" V "

<< endl;

cout << endl;

cout << "\t"

" 3. Left Go Left "

<< endl;

cout << "\t"

" <---- "

<< endl;

cout << endl;

cout << "\t"

" 4. Right Go Right "

<< endl;

cout << "\t"

" -----> "

<< endl;

cout << endl;

cout << "\t"

" 5. Space Fire "

<< endl;

cout << endl

<< endl;

cout << "\t"

" 6. Esc End Game "

<< endl;

cout << endl;

cout << "\t"

" Press any key to continue ..!"

<< endl;

getch();

}

void clearScreen()

{

cout << " Press any Key to Continue ! " << endl;

getch();

system("cls");

}

void printMainCharacter()

{

SetConsoleTextAttribute(acolor, 4);

for (int i = 0; i < 4; i++)

{

gotoxy(mainX, mainY + i);

for (int j = 0; j < 4; j++)

{

cout << mainCharacter1[i][j];

}

cout << endl;

}

}

void eraseMainCharacter()

{

for (int i = 0; i < 4; i++)

{

gotoxy(mainX, mainY + i);

for (int j = 0; j < 4; j++)

{

cout << " ";

}

cout << endl;

}

}

void moveMainCharLeft()

{

char next = printMaze[mainY][mainX - 1];

char next1 = printMaze[mainY + 1][mainX - 1];

char next2 = printMaze[mainY + 2][mainX - 1];

char next3 = printMaze[mainY + 3][mainX - 1];

if (next == '\*' || next1 == '\*' || next2 == '\*' || next3 == '\*')

{

score += 5;

if (next == '\*')

printMaze[mainY][mainX - 1] = ' ';

if (next1 == '\*')

printMaze[mainY + 1][mainX - 1] = ' ';

if (next2 == '\*')

printMaze[mainY + 2][mainX - 1] = ' ';

if (next3 == '\*')

printMaze[mainY + 3][mainX - 1] = ' ';

}

if ((next != '%' && next1 != '%' && next2 != '%' && next3 != '%') && (next != '#' && next1 != '#' && next2 != '#' && next3 != '#') && (next != '!' && next1 != '!' && next2 != '!' && next3 != '!') && (next != '-' && next1 != '-' && next2 != '-' && next3 != '-') && (next != '<' && next1 != '<' && next2 != '<' && next3 != '<') && (next != char(boxx) && next1 != char(boxx) && next2 != char(boxx) && next3 != char(boxx)) && (next != '0' && next1 != '0' && next2 != '0' && next3 != '0'))

{

eraseMainCharacter();

mainX = mainX - 1;

printMainCharacter();

}

else if ((next == '<' || next1 == '<' || next2 == '<' || next3 == '<') || (next == '>' || next1 == '>' || next2 == '>' || next3 == '>') || (next == '0' || next1 == '0' || next2 == '0' || next3 == '0') || (next == char(questionMark) || next1 == char(questionMark) || next2 == char(questionMark) || next3 == char(questionMark)) || (next == '.' || next1 == '.' || next2 == '.' || next3 == '.'))

{

health--;

}

}

void moveMainCharRight()

{

char next = printMaze[mainY][mainX + 4];

char next1 = printMaze[mainY + 1][mainX + 4];

char next2 = printMaze[mainY + 2][mainX + 4];

char next3 = printMaze[mainY + 3][mainX + 4];

if (next == '\*' || next1 == '\*' || next2 == '\*' || next3 == '\*')

{

score += 5;

if (next == '\*')

printMaze[mainY][mainX - 1] = ' ';

if (next1 == '\*')

printMaze[mainY + 1][mainX - 1] = ' ';

if (next2 == '\*')

printMaze[mainY + 2][mainX - 1] = ' ';

if (next3 == '\*')

printMaze[mainY + 3][mainX - 1] = ' ';

}

if ((next != '%' && next1 != '%' && next2 != '%' && next3 != '%') && (next != '#' && next1 != '#' && next2 != '#' && next3 != '#') && (next != '!' && next1 != '!' && next2 != '!' && next3 != '!') && (next != '-' && next1 != '-' && next2 != '-' && next3 != '-') && (next != '<' && next1 != '<' && next2 != '<' && next3 != '<') && (next != char(boxx) && next1 != char(boxx) && next2 != char(boxx) && next3 != char(boxx)) && (next != '0' && next1 != '0' && next2 != '0' && next3 != '0'))

{

eraseMainCharacter();

mainX = mainX + 1;

printMainCharacter();

}

else if ((next == '<' || next1 == '<' || next2 == '<' || next3 == '<') || (next == '>' || next1 == '>' || next2 == '>' || next3 == '>') || (next == '0' || next1 == '0' || next2 == '0' || next3 == '0') || (next == char(questionMark) || next1 == char(questionMark) || next2 == char(questionMark) || next3 == char(questionMark)) || (next == '.' || next1 == '.' || next2 == '.' || next3 == '.'))

{

health--;

}

// if (next == char(178) || next1 == char(176) || next2 == char(178) || next3 == char(178))

// {

// system("cls");

// cout<<"you have won the game";

// // break;

// }

}

void printdoor()

{

for (int idx = 0; idx < 3; idx++)

{

gotoxy(50, 5 + idx);

for (int idx = 0; idx < 3; idx++)

{

cout << char(178);

}

cout << endl;

}

}

void moveMainCharUp()

{

char next = printMaze[mainY - 1][mainX];

char next1 = printMaze[mainY - 1][mainX + 1];

char next2 = printMaze[mainY - 1][mainX + 2];

char next3 = printMaze[mainY - 1][mainX + 3];

if (next == '\*' || next1 == '\*' || next2 == '\*' || next3 == '\*')

{

score += 5;

if (next == '\*')

printMaze[mainY][mainX - 1] = ' ';

if (next1 == '\*')

printMaze[mainY + 1][mainX - 1] = ' ';

if (next2 == '\*')

printMaze[mainY + 2][mainX - 1] = ' ';

if (next3 == '\*')

printMaze[mainY + 3][mainX - 1] = ' ';

}

if ((next != '%' && next1 != '%' && next2 != '%' && next3 != '%') && (next != '#' && next1 != '#' && next2 != '#' && next3 != '#') && (next != '!' && next1 != '!' && next2 != '!' && next3 != '!') && (next != '-' && next1 != '-' && next2 != '-' && next3 != '-') && (next != '<' && next1 != '<' && next2 != '<' && next3 != '<') && (next != char(boxx) && next1 != char(boxx) && next2 != char(boxx) && next3 != char(boxx)) && (next != '0' && next1 != '0' && next2 != '0' && next3 != '0'))

{

eraseMainCharacter();

mainY = mainY - 1;

printMainCharacter();

}

else if ((next == '<' && next1 == '<' && next2 == '<' && next3 == '<') || (next == '>' && next1 == '>' && next2 == '>' && next3 == '>') || (next == '0' && next1 == '0' && next2 == '0' && next3 == '0') || (next == char(questionMark) && next1 == char(questionMark) && next2 == char(questionMark) && next3 == char(questionMark)) || (next == '.' && next1 == '.' && next2 == '.' && next3 == '.'))

{

health--;

}

}

void moveMainCharDown()

{

char next = printMaze[mainY + 4][mainX];

char next1 = printMaze[mainY + 4][mainX + 1];

char next2 = printMaze[mainY + 4][mainX + 2];

char next3 = printMaze[mainY + 4][mainX + 3];

if (next == '\*' || next1 == '\*' || next2 == '\*' || next3 == '\*')

{

score += 5;

if (next == '\*')

printMaze[mainY][mainX - 1] = ' ';

if (next1 == '\*')

printMaze[mainY + 1][mainX - 1] = ' ';

if (next2 == '\*')

printMaze[mainY + 2][mainX - 1] = ' ';

if (next3 == '\*')

printMaze[mainY + 3][mainX - 1] = ' ';

}

if ((next != '%' && next1 != '%' && next2 != '%' && next3 != '%') && (next != '#' && next1 != '#' && next2 != '#' && next3 != '#') && (next != '!' && next1 != '!' && next2 != '!' && next3 != '!') && (next != '-' && next1 != '-' && next2 != '-' && next3 != '-') && (next != '<' && next1 != '<' && next2 != '<' && next3 != '<') && (next != char(boxx) && next1 != char(boxx) && next2 != char(boxx) && next3 != char(boxx)) && (next != '0' && next1 != '0' && next2 != '0' && next3 != '0'))

{

eraseMainCharacter();

mainY = mainY + 1;

printMainCharacter();

}

else if ((next == '<' && next1 == '<' && next2 == '<' && next3 == '<') || (next == '>' && next1 == '>' && next2 == '>' && next3 == '>') || (next == '0' && next1 == '0' && next2 == '0' && next3 == '0') || (next == char(questionMark) && next1 == char(questionMark) && next2 == char(questionMark) && next3 == char(questionMark)) || (next == '.' && next1 == '.' && next2 == '.' && next3 == '.'))

{

health--;

}

}

void generateBullet()

{

char next;

bulletX[bulletCount] = mainX + 4;

bulletY[bulletCount] = mainY;

isBulletActive[bulletCount] = true;

gotoxy(mainX + 4, mainY);

cout << "-";

bulletCount++;

}

void generateBulletenemy()

{

char next;

bulletXenemy[bulletCountenemy] = enemy1X - 1;

bulletYenemy[bulletCountenemy] = enemy1Y;

gotoxy(enemy1X - 1, enemy1Y);

for (next = 0; next < 100; next++)

{

if (next == ' ')

{

cout << ".";

}

}

bulletCountenemy++;

}

void removeBulletFromArray(int index)

{

for (int x = index; x < bulletCount - 1; x++)

{

bulletX[x] = bulletX[x + 1];

bulletY[x] = bulletY[x + 1];

}

bulletCount--;

}

void moveBullet()

{

SetConsoleTextAttribute(acolor, 7);

for (int x = 0; x < bulletCount; x++)

{

if (isBulletActive[x] == true)

{

char next = getCharAtxy(bulletX[x] + 1, bulletY[x] + 1);

if (next != ' ')

{

eraseBullet(bulletX[x], bulletY[x]);

removeBulletFromArray(x);

}

else

{

eraseBullet(bulletX[x], bulletY[x]);

bulletX[x] = bulletX[x] + 1;

printBullet(bulletX[x], bulletY[x]);

}

}

}

}

void printBullet(int x, int y)

{

gotoxy(x, y);

cout << "-";

}

void eraseBullet(int x, int y)

{

gotoxy(x, y);

cout << " ";

}

void printEnemy1()

{

for (int i = 0; i < 2; i++)

{

gotoxy(enemy1X, enemy1Y + i);

for (int j = 0; j < 5; j++)

{

cout << Enemy1[i][j];

}

cout << endl;

}

}

void eraseEnemy1()

{

for (int i = 0; i < 2; i++)

{

gotoxy(enemy1X, enemy1Y + i);

for (int j = 0; j < 5; j++)

{

cout << " ";

}

cout << endl;

}

}

void moveEnemy1()

{

{

if (dir == "up")

{

moveEnemyUp();

}

else if (dir == "down")

{

moveEnemyDown();

}

}

}

void moveEnemyUp()

{

char next = getCharAtxy(enemy1X, enemy1Y - 1);

char next1 = getCharAtxy(enemy1X + 1, enemy1Y - 1);

if ((next != '%' && next1 != '%') && (next != '#' && next1 != '#') && (next != '!' && next1 != '!') && (next != '-' && next1 != '-') && (next != '<' && next1 != '<') && (next != char(boxx) && next1 != char(boxx)) && (next != '0' && next1 != '0'))

{

eraseEnemy1();

enemy1Y = enemy1Y - 1;

printEnemy1();

}

else

{

dir = "down";

}

if (next == '0' || next == '>' || next == '<' || next == '(' || next == ')' || next1 == '0' || next1 == '>' || next1 == '<' || next1 == '(' || next1 == ')')

{

health--;

}

}

void moveEnemyDown()

{

char next = getCharAtxy(enemy1X, enemy1Y + 3);

char next1 = getCharAtxy(enemy1X + 1, enemy1Y + 3);

if (next == '0' || next == '>' || next == '<' || next == '(' || next == ')' || next1 == '0' || next1 == '>' || next1 == '<' || next1 == '(' || next1 == ')')

{

health--;

}

if ((next != '%' && next1 != '%') && (next != '#' && next1 != '#') && (next != '!' && next1 != '!') && (next != '-' && next1 != '-') && (next != '<' && next1 != '<') && (next != char(boxx) && next1 != char(boxx)) && (next != '0' && next1 != '0'))

{

eraseEnemy1();

enemy1Y = enemy1Y + 1;

printEnemy1();

}

if (next == '#' || next1 == '#')

{

dir = "up";

}

}

void printBulletenemy(int x, int y)

{

gotoxy(x, y);

cout << ".";

}

void eraseBulletenemy(int x, int y)

{

gotoxy(x, y);

cout << " ";

}

void removeBulletFromArrayenemy(int index)

{

for (int x = index; x < bulletCountenemy - 1; x++)

{

bulletXenemy[x] = bulletXenemy[x + 1];

bulletYenemy[x] = bulletYenemy[x + 1];

}

bulletCountenemy--;

}

void moveBulletEnemy()

{

for (int x = 0; x < bulletCountenemy; x++)

{

char next = getCharAtxy(bulletXenemy[x] - 1, bulletYenemy[x]);

if (next != ' ')

{

eraseBulletenemy(bulletXenemy[x], bulletYenemy[x]);

removeBulletFromArrayenemy(x);

}

else if (next == ' ')

{

eraseBulletenemy(bulletXenemy[x], bulletYenemy[x]);

bulletXenemy[x] = bulletXenemy[x] - 1;

printBulletenemy(bulletXenemy[x], bulletYenemy[x]);

}

if ((next == '0' || next == '<' || next == '>' || next == '(' || next == ')'))

{

health--;

}

}

}

void printEnemy2(int enemy2X, int enemy2Y)

{

for (int i = 0; i < 3; i++)

{

gotoxy(enemy2X, enemy2Y + i);

for (int j = 0; j < 3; j++)

{

cout << Enemy2[i][j];

}

cout << endl;

}

}

void eraseEnemy2(int enemy2X, int enemy2Y)

{

for (int i = 0; i < 3; i++)

{

gotoxy(enemy2X, enemy2Y + i);

for (int j = 0; j < 3; j++)

{

cout << " ";

}

cout << endl;

}

}

void chaseEnemy2()

{

int chaseX = enemy2X - mainX; // difference for chase

int chaseY = enemy2Y - mainY;

if (chaseX > 0) // left

{

char nextlocation = getCharAtxy(enemy2X - 1, enemy2Y);

char nextlocation1 = getCharAtxy(enemy2X - 1, enemy2Y + 1);

char nextlocation2 = getCharAtxy(enemy2X - 1, enemy2Y + 2);

if (nextlocation == '0' || nextlocation == '>' || nextlocation == '<' || nextlocation == '(' || nextlocation == ')' || nextlocation1 == '0' || nextlocation1 == '>' || nextlocation1 == '<' || nextlocation1 == '(' || nextlocation1 == ')' || nextlocation2 == '0' || nextlocation2 == '<' || nextlocation2 == '>' || nextlocation2 == '(' || nextlocation2 == ')')

{

health--;

}

if (nextlocation == ' ' && nextlocation1 == ' ' && nextlocation2 == ' ')

{

eraseEnemy2(enemy2X, enemy2Y);

enemy2X = enemy2X - 1;

printEnemy2(enemy2X, enemy2Y);

}

}

if (chaseX < 0) // right

{

char nextlocation = getCharAtxy(enemy2X + 3, enemy2Y);

char nextlocation1 = getCharAtxy(enemy2X + 3, enemy2Y + 1);

char nextlocation2 = getCharAtxy(enemy2X + 3, enemy2Y + 2);

if (nextlocation == '0' || nextlocation == '>' || nextlocation == '<' || nextlocation == '(' || nextlocation == ')' || nextlocation1 == '0' || nextlocation1 == '>' || nextlocation1 == '<' || nextlocation1 == '(' || nextlocation1 == ')' || nextlocation2 == '0' || nextlocation2 == '<' || nextlocation2 == '>' || nextlocation2 == '(' || nextlocation2 == ')')

{

health--;

}

if (nextlocation == ' ' && nextlocation1 == ' ' && nextlocation2 == ' ')

{

eraseEnemy2(enemy2X, enemy2Y);

enemy2X = enemy2X + 1;

printEnemy2(enemy2X, enemy2Y);

}

}

if (chaseY < 0) // down

{

char nextlocation = getCharAtxy(enemy2X, enemy2Y + 3);

char nextlocation1 = getCharAtxy(enemy2X + 1, enemy2Y + 3);

char nextlocation2 = getCharAtxy(enemy2X + 2, enemy2Y + 3);

if (nextlocation == '0' || nextlocation == '>' || nextlocation == '<' || nextlocation == '(' || nextlocation == ')' || nextlocation1 == '0' || nextlocation1 == '>' || nextlocation1 == '<' || nextlocation1 == '(' || nextlocation1 == ')' || nextlocation2 == '0' || nextlocation2 == '<' || nextlocation2 == '>' || nextlocation2 == '(' || nextlocation2 == ')')

{

health--;

}

if (nextlocation == ' ' && nextlocation1 == ' ' && nextlocation2 == ' ')

{

eraseEnemy2(enemy2X, enemy2Y);

enemy2Y = enemy2Y + 1;

printEnemy2(enemy2X, enemy2Y);

}

}

if (chaseY > 0) // up

{

char nextlocation = getCharAtxy(enemy2X, enemy2Y - 1);

char nextlocation1 = getCharAtxy(enemy2X + 1, enemy2Y - 1);

char nextlocation2 = getCharAtxy(enemy2X + 2, enemy2Y - 1);

if (nextlocation == '0' || nextlocation == '>' || nextlocation == '<' || nextlocation == '(' || nextlocation == ')' || nextlocation1 == '0' || nextlocation1 == '>' || nextlocation1 == '<' || nextlocation1 == '(' || nextlocation1 == ')' || nextlocation2 == '0' || nextlocation2 == '<' || nextlocation2 == '>' || nextlocation2 == '(' || nextlocation2 == ')')

{

health--;

}

if (nextlocation == ' ' && nextlocation1 == ' ' && nextlocation2 == ' ')

{

eraseEnemy2(enemy2X, enemy2Y);

enemy2Y = enemy2Y - 1;

printEnemy2(enemy2X, enemy2Y);

}

}

}

void printEnemy3()

{

for (int i = 0; i < 2; i++)

{

gotoxy(enemy3X, enemy3Y + i);

for (int j = 0; j < 5; j++)

{

cout << Enemy3[i][j];

}

cout << endl;

}

}

void eraseEnemy3()

{

for (int i = 0; i < 2; i++)

{

gotoxy(enemy3X, enemy3Y + i);

for (int j = 0; j < 5; j++)

{

cout << " ";

}

cout << endl;

}

}

void moveMainEnemyLeft()

{

char next = getCharAtxy(enemy3X - 1, enemy3Y);

char next1 = getCharAtxy(enemy3X - 1, enemy3Y + 1);

// char next2 = printMaze[enemy3Y + 2][enemy3X - 1];

// char next3 = printMaze[enemy3Y + 3][enemy3X - 1];

// char next4 = printMaze[enemy3Y + 4][enemy3X - 1];

if ((next != '%' && next1 != '%') && (next != '#' && next1 != '#') && (next != '!' && next1 != '!') && (next != '-' && next1 != '-') && (next != '<' && next1 != '<') && (next != char(boxx) && next1 != char(boxx)) && (next != '0' && next1 != '0'))

// if ((next != '%' && next1 != '%' && next2 != '%' && next3 != '%' && next4 != '%') && (next != '#' && next1 != '#' && next2 != '#' && next3 != '#' && next4 != '#') && (next != '!' && next1 != '!' && next2 != '!' && next3 != '!'&& next4 != '!') && (next != '-' && next1 != '-' && next2 != '-' && next3 != '-'&& next4 != '-') && (next != '<' && next1 != '<' && next2 != '<' && next3 != '<'&& next4 != '<') && (next != char(boxx) && next1 != char(boxx) && next2 != char(boxx) && next3 != char(boxx) && next4 != char(boxx)) && (next != '0' && next1 != '0' && next2 != '0' && next3 != '0' && next4 != '0'))

{

eraseEnemy3();

enemy3X = enemy3X - 1;

printEnemy3();

}

else

{

destination = "down";

}

if (next == '0' || next == '>' || next == '<' || next == '(' || next == ')' || next1 == '0' || next1 == '>' || next1 == '<' || next1 == '(' || next1 == ')')

{

health--;

}

}

void moveMainEnemyRight()

{

char next = getCharAtxy(enemy3X + 5, enemy3Y);

char next1 = getCharAtxy(enemy3X + 5, enemy3Y + 1);

// char next2= printMaze[enemy3Y + 2][enemy3X + 4];

// char next3 = printMaze[enemy3Y+ 3][enemy3X + 4];

// char next4 = printMaze[enemy3Y+ 4][enemy3X + 4];

if ((next != '%' && next1 != '%') && (next != '#' && next1 != '#') && (next != '!' && next1 != '!') && (next != '-' && next1 != '-') && (next != '<' && next1 != '<') && (next != char(boxx) && next1 != char(boxx)) && (next != '0' && next1 != '0'))

// if ((next != '%' && next1 != '%' && next2 != '%' && next3 != '%' && next4 != '%') && (next != '#' && next1 != '#' && next2 != '#' && next3 != '#' && next4 != '#') && (next != '!' && next1 != '!' && next2 != '!' && next3 != '!'&& next4 != '!') && (next != '-' && next1 != '-' && next2 != '-' && next3 != '-'&& next4 != '-') && (next != '<' && next1 != '<' && next2 != '<' && next3 != '<'&& next4 != '<') && (next != char(boxx) && next1 != char(boxx) && next2 != char(boxx) && next3 != char(boxx) && next4 != char(boxx)) && (next != '0' && next1 != '0' && next2 != '0' && next3 != '0' && next4 != '0'))

{

eraseEnemy3();

enemy3X = enemy3X + 1;

printEnemy3();

}

else if ((next == '#' && next1 == '#') || (next == '!' && next1 == '!') || (next == '%' && next1 == '%'))

{

destination = "up";

}

if (next == '0' || next == '>' || next == '<' || next == '(' || next == ')' || next1 == '0' || next1 == '>' || next1 == '<' || next1 == '(' || next1 == ')')

{

health--;

}

}

void moveMainEnemyUp()

{

char next = getCharAtxy(enemy3X, enemy3Y - 1);

char next1 = getCharAtxy(enemy3X + 1, enemy3Y - 1);

// char next2 = printMaze[enemy3Y - 1][enemy3X + 2];

// char next3 = printMaze[enemy3Y - 1][enemy3X + 3];

// char next4 = printMaze[enemy3Y - 1][enemy3X + 4];

if ((next != '%' && next1 != '%') && (next != '#' && next1 != '#') && (next != '!' && next1 != '!') && (next != '-' && next1 != '-') && (next != '<' && next1 != '<') && (next != char(boxx) && next1 != char(boxx)) && (next != '0' && next1 != '0'))

{

eraseEnemy3();

enemy3Y = enemy3Y - 1;

printEnemy3();

}

else if ((next == '#' && next1 == '#') || (next == '!' && next1 == '!'))

{

destination = "left";

}

if (next == '0' || next == '>' || next == '<' || next == '(' || next == ')' || next1 == '0' || next1 == '>' || next1 == '<' || next1 == '(' || next1 == ')')

{

health--;

}

}

void moveMainEnemyDown()

{

char next = getCharAtxy(enemy3X, enemy3Y + 4);

char next1 = getCharAtxy(enemy3X + 1, enemy3Y + 4);

// char next2 = printMaze[enemy3Y + 4][enemy3X + 2];

// char next3 = printMaze[enemy3Y + 4][enemy3X + 3];

// char next4 = printMaze[enemy3Y + 4][enemy3X + 4];

if ((next != '%' && next1 != '%') && (next != '#' && next1 != '#') && (next != '!' && next1 != '!') && (next != '-' && next1 != '-') && (next != '<' && next1 != '<') && (next != char(boxx) && next1 != char(boxx)) && (next != '0' && next1 != '0'))

{

eraseEnemy3();

enemy3Y = enemy3Y + 1;

printEnemy3();

}

else if ((next == '#' && next1 == '#') || (next == '!' && next1 == '!'))

{

destination = "right";

}

if (next == '0' || next == '>' || next == '<' || next == '(' || next == ')' || next1 == '0' || next1 == '>' || next1 == '<' || next1 == '(' || next1 == ')')

{

health--;

}

}

void moveEnemy3()

{

{

if (destination == "up")

{

moveMainEnemyUp();

}

else if (destination == "down")

{

moveMainEnemyDown();

}

else if (destination == "left")

{

moveMainEnemyLeft();

}

else if (destination == "right")

{

moveMainEnemyRight();

}

}

}

void bulletCollisionWithEnemy1()

{

for (int x = 0; x < bulletCount; x++)

{

if (bulletX[x] + 1 == enemy1X && (bulletY[x] == enemy1Y || bulletY[x] == enemy1Y + 1 || bulletY[x] == enemy1Y + 2 || bulletY[x] == enemy1Y + 3))

{

addScore();

eraseBulletenemy(bulletX[x], bulletY[x]);

removeBulletFromArrayenemy(x);

health--;

}

}

}

void printScore()

{

gotoxy(109, 8);

cout << "Score: " << score << " ";

}

void addScore()

{

score += 5;

}

// void Exit()

// {

// }

void gotoxy(int x, int y)

{

COORD coordinates;

coordinates.X = x;

coordinates.Y = y;

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

}

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 : ' ';

}

void storeData()

{

fstream yourfile;

yourfile.open("data.txt", ios::out);

yourfile << score << ",";

yourfile << health << ",";

yourfile << mainX << ",";

yourfile << mainY << ",";

yourfile << enemy1X << ",";

yourfile << enemy1Y << ",";

yourfile << enemy2X << ",";

yourfile << enemy2Y << ",";

yourfile << enemy3X << ",";

yourfile << enemy3Y << ",";

yourfile << escapeKeyX << ",";

yourfile << escapeKeyY << endl;

yourfile.close();

}

void loadData()

{

fstream yourfile;

string record;

yourfile.open("data.txt", ios::in);

while (getline(yourfile, record))

{

stringstream t0(getField(record, 1));

t0 >> score;

stringstream t1(getField(record, 2));

t1 >> health;

stringstream t2(getField(record, 3));

t2 >> mainX;

stringstream t3(getField(record, 4));

t3 >> mainY;

stringstream t4(getField(record, 5));

t4 >> enemy1X;

stringstream t5(getField(record, 6));

t5 >> enemy1Y;

stringstream t6(getField(record, 7));

t6 >> enemy2X;

stringstream t7(getField(record, 8));

t7 >> enemy2Y;

stringstream t8(getField(record, 9));

t8 >> enemy3X;

stringstream t9(getField(record, 10));

t9 >> enemy3Y;

stringstream t10(getField(record, 11));

t10 >> escapeKeyX;

stringstream t11(getField(record, 12));

t11 >> escapeKeyY;

}

yourfile.close();

}

string getField(string record, int field)

{

int commaCount = 0;

string item;

for (int x = 0; x < record.length(); x++)

{

if (record[x] == ',')

{

commaCount++;

}

else if (commaCount == field)

{

item = item + record[x];

}

}

return item;

}