The Mystery Maze

Session 2023 - 2027

Submitted by:
Hannan Mushtaq 2023-CS-85

Supervised by:
Dr. Awais Hassan

Course:
CSC-102 Programming Fundamentals

Department of Computer Science
University of Engineering and Technology
Lahore Pakistan

Table Of Contents

1. Desciption & Story Writing

The mystery maze is a game in which the player finds himself trapped inside a maze amidst enemies and little bombs. He has to escape the maze but in order to do so he needs a key. He can only see the key when he has killed all his enemies or collected all the pills inside the maze. With each step, he faces danger, evading enemies and bombs. It's a thrilling adventure that tests his skills and strategic thinking. Can he find the key and make it out of the maze filled with mysteries? Only time will tell!

2. Game Characters Description

There are a total of 4 characters in the game:

1. The Player ([P])

The player can move left, right, up or down and can shoot towards left or right. The movement is controlled by arrow keys and shooting with Z and X keys , respectively.

2. Enemy1 ([1])

Enemy1 moves up and down the maze guarding the left boundary.

3. Enemy2 ([2])

Enemy2 moves up and down the maze guarding the right boundary.

4. Enemy3 ([3])

Enemy3 moves left and right guarding the central area of the maze.

3. Game Objects Description

The game objects are described below:

- An Ordinary Pill (.) which increases score by 1.
- An Unusual Pill (o) which decreases player health by 5.
- A Bonus Pill (*) which increases player health by 5.
- A Key ($) which increases score by 1000 and wins you the game.

4. Rules & Interactions

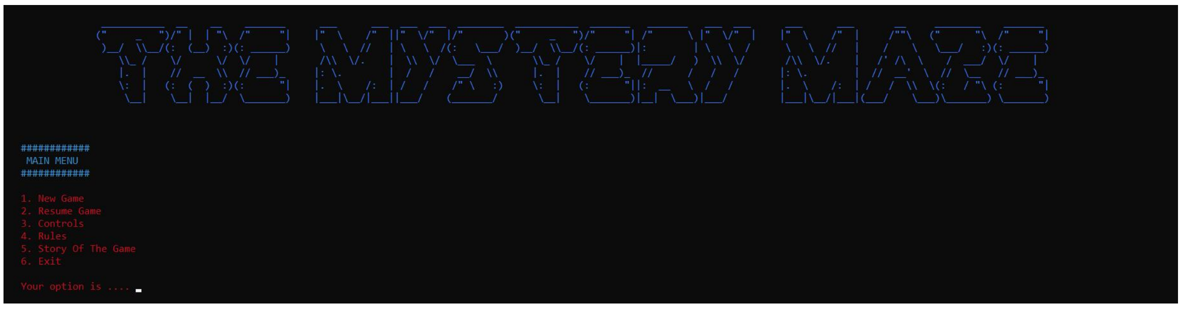
The rules of the game are as follows:

- The player has a total of 50 health points.
- There are 3 enemies in the game.
- Each enemy has 25 health points which totals to 75 enemy health points.
- Collect as many ordinary pills as you can.
- Each ordinary pill increases the score by 1.
- Try to avoid the unusual looking pills.
- Each unusual looking pill decreases the player health by 5.
- A bonus pill appears randomly in the maze.
- Collect the bonus pill to increase player health by 5.
- Do not get close to an enemy.
- Being close to an enemy decreases player health by 5.
- Avoid contact with an enemy.
- Contact with an enemy decreases player health completely thus making you lose the game.
- After an enemy is killed in a position, that specific position becomes a bomb.
- If you go to a bomb area , you will die and lose the game.
- The player can shoot left or right to kill enemies.
- Each hit with an enemy decreases enemy's health by 5 and increases score by 2.
- 5 hits to an enemy kils it.
- If the player health decreases to 0 , you lose the game.
- If you collect all the pills or kill all enemies , a key ($) appears in the maze.
- Collect the key to win the game and get out of the mystery maze.

5. Goal Of The Game

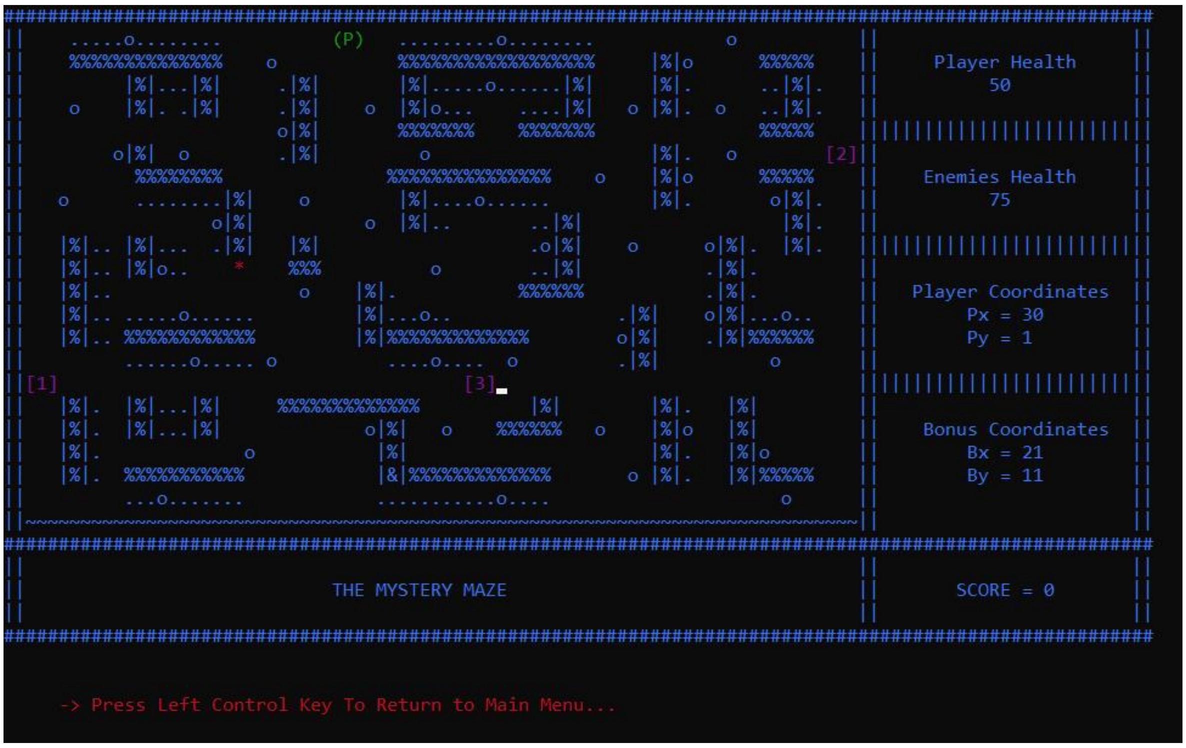
The goal of the game is for the player to find a key to escape the maze. Hecan do so by collecting pills or by killing the enemies.

6. Wireframes



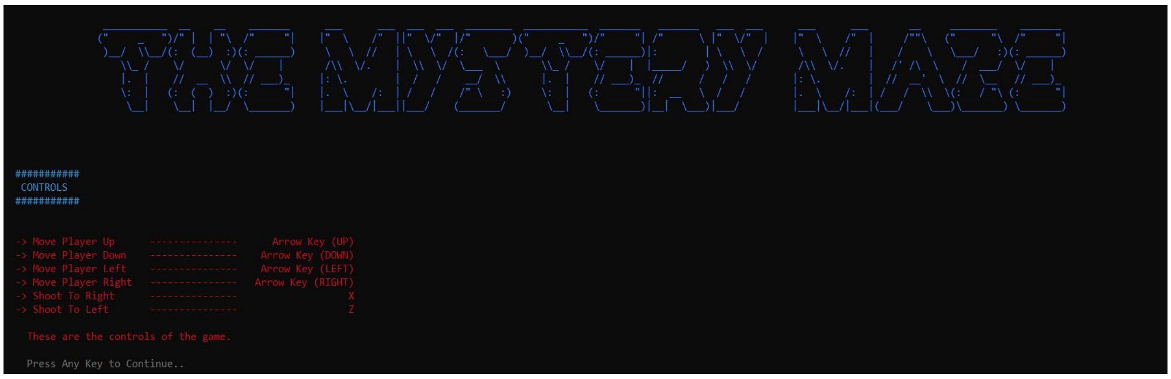
```
############
 MAIN MENU
############

1. New Game
2. Resume Game
3. Controls
4. Rules
5. Story Of The Game
6. Exit

Your option is ....
```

Figure 1: Main Menu



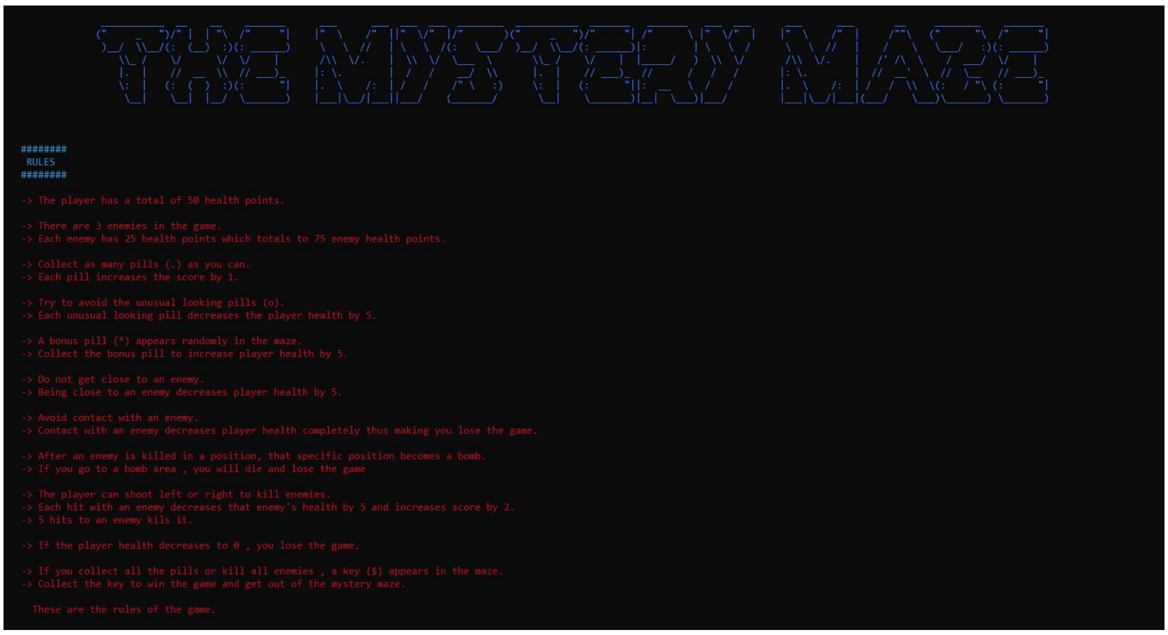
```
Player Health
     50

Enemies Health
     75

Player Coordinates
     Px = 30
     Py = 1

Bonus Coordinates
     Bx = 21
     By = 11

THE MYSTERY MAZE          SCORE = 0

-> Press Left Control Key To Return to Main Menu...
```

Figure 2: Game Interface



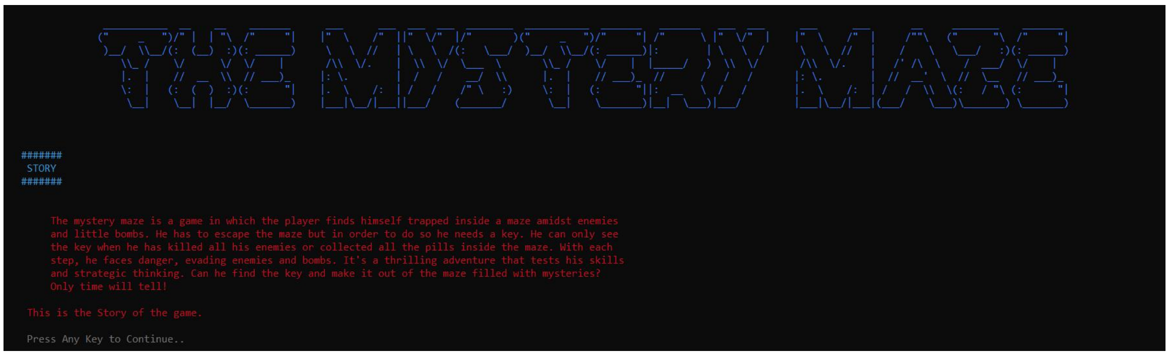
```
###########
 CONTROLS
###########


-> Move Player Up       ---------------      Arrow Key (UP)
-> Move Player Down     ---------------    Arrow Key (DOWN)
-> Move Player Left     ---------------    Arrow Key (LEFT)
-> Move Player Right    ---------------   Arrow Key (RIGHT)
-> Shoot To Right       ---------------                   X
-> Shoot To Left        ---------------                   Z

  These are the controls of the game.

  Press Any Key to Continue..
```

Figure 3: Controls



```
########
 RULES
########

-> The player has a total of 50 health points.

-> There are 3 enemies in the game.
-> Each enemy has 25 health points which totals to 75 enemy health points.

-> Collect as many pills (.) as you can.
-> Each pill increases the score by 1.

-> Try to avoid the unusual looking pills (o).
-> Each unusual looking pill decreases the player health by 5.

-> A bonus pill (*) appears randomly in the maze.
-> Collect the bonus pill to increase player health by 5.

-> Do not get close to an enemy.
-> Being close to an enemy decreases player health by 5.

-> Avoid contact with an enemy.
-> Contact with an enemy decreases player health completely thus making you lose the game.

-> After an enemy is killed in a position, that specific position becomes a bomb.
-> If you go to a bomb area , you will die and lose the game

-> The player can shoot left or right to kill enemies.
-> Each hit with an enemy decreases that enemy's health by 5 and increases score by 2.
-> 5 hits to an enemy kils it.

-> If the player health decreases to 0 , you lose the game.

-> If you collect all the pills or kill all enemies , a key ($) appears in the maze.
-> Collect the key to win the game and get out of the mystery maze.

  These are the rules of the game.
```

Figure 4: Rules



```
#######
 STORY
#######


    The mystery maze is a game in which the player finds himself trapped inside a maze amidst enemies
    and little bombs. He has to escape the maze but in order to do so he needs a key. He can only see
    the key when he has killed all his enemies or collected all the pills inside the maze. With each
    step, he faces danger, evading enemies and bombs. It's a thrilling adventure that tests his skills
    and strategic thinking. Can he find the key and make it out of the maze filled with mysteries?
    Only time will tell!

  This is the Story of the game.

  Press Any Key to Continue..
```

Figure 5: Story

7. Data Structures

```
int px = 30, py = 1, e1x = 2, e1y = 1, e2x = 75, e2y = 21, e3x = 72, e3y = 16;
// Variables for player and enemy coordinates
int score = 0;                // Variable for storing score
int enemyHealth = 75;   // Variable for storing enemies health
int e1 = 0, e2 = 0, e3 = 0; // Variables for counting number of hits to each enemy
int playerHealth = 50;   // Variable for storing player health
int bx, by;          // Variables for bonus pill coordinates
int kx, ky;          // Variables for key coordinates
string direction1 = "down";     // Variable for enemy1 direction
string direction2 = "up";           // Variable for enemy2 direction
string direction3 = "left";       // Variable for enemy3 direction
string filename = "Coordinates.txt";  // Variable for storing coordinates file name
```

8. Function Prototypes

```
void maze();
char getCharAtxy(short int x, short int y);
void gotoxy(int x, int y);
void Enemy1(int &e1x, int &e1y);
void eraseEnemy1(int &e1x, int &e1y);
void Enemy2(int &e2x, int &e2y);
void eraseEnemy2(int &e2x, int &e2y);
void Enemy3(int &e3x, int &e3y);
void eraseEnemy3(int &e3x, int &e3y);
void moveEnemy1(string &direction1, int &e1x, int &e1y, int &e1);
string cdirection1(string &direction1, int &e1y);
void moveEnemy2(string &direction2, int &e2x, int &e2y, int &e2);
string cdirection2(string &direction2, int &e2y);
void moveEnemy3(string &direction3, int &e3x, int &e3y, int &e3);
string cdirection3(string &direction3, int &e3x);
void Player(int &px, int &py);
void erasePlayer(int &px, int &py);
void movePlayerLeft(int &px, int &py, int &score, int &bx, int &by, int
&playerHealth);
void movePlayerRight(int &px, int &py, int &score, int &bx, int &by, int
&playerHealth);
```

```
void movePlayerUp(int &px, int &py, int &score, int &bx, int &by, int
&playerHealth);
void movePlayerDown(int &px, int &py, int &score, int &bx, int &by, int
&playerHealth);
void playerHealthDecrement(int &playerHealth, int &px, int &py, int &e1x, int &e1y,
int &e2x, int &e2y, int &e3x, int &e3y);
void loadCoordinatesFile(string filename, int &px, int &py, int &e1x, int &e1y, int
&e2x, int &e2y, int &e3x, int &e3y, int &score, int &playerHealth, int
&enemyHealth, int &e1, int &e2, int &e3);
void readCoordinatesFile(string filename, int &px, int &py, int &e1x, int &e1y, int
&e2x, int &e2y, int &e3x, int &e3y, int &score, int &playerHealth, int
&enemyHealth, int &e1, int &e2, int &e3);
void gameLoop(string filename, int &px, int &py, int &e1x, int &e1y, int &e2x, int
&e2y, int &e3x, int &e3y, int &score, int &playerHealth, int &enemyHealth, int &bx,
int &by, string &direction1, string &direction2, string &direction3, int &e1, int &e2,
int &e3, int &kx, int &ky);
string getField(string record, int field);
void printScore(int &score);
string setcolor(unsigned short color);
void moveFireRight(int &px, int &py, int &score, int &enemyHealth, int &e1, int
&e2, int &e3);
void moveFireLeft(int &px, int &py, int &score, int &enemyHealth, int &e1, int &e2,
int &e3);
void printFireRight(int &px, int &py);
void printFireLeft(int &px, int &py);
void eraseFire(int &px, int &py);
string controls();
void Header();
string rules();
void bounsPill();
void printEnemyHealth(int enemyHealth);
void printPlayerHealth(int playerHealth);
void printBonusPill(int &bx, int &by);
void printKey(int &kx, int &ky);
void printPlayerCoordinates(int &px, int &py);
void printBonusPillCoordinates(int &bx, int &by);
string mainMenu();
void clearScreen();
void clearHeader();
void wrongOption();
bool pillCheck();
bool keyCheck();
bool bonusCheck();
string about();
```

9. Complete Code

```
// Libraries

#include <iostream>  // Library for input output functons
#include <windows.h> // Library for using getCharAtxy function
#include <conio.h>   // Library for getch function
#include <ctime>     // Library for srand function
#include <iomanip>   // Library for setw function
#include <fstream>   // Library for file handling
#include <stdlib.h>  // Library for abort function
using namespace std;
```

```
// Function Prototypes
```

```
void maze();
char getCharAtxy(short int x, short int y);
void gotoxy(int x, int y);
void Enemy1(int &e1x, int &e1y);
void eraseEnemy1(int &e1x, int &e1y);
void Enemy2(int &e2x, int &e2y);
void eraseEnemy2(int &e2x, int &e2y);
void Enemy3(int &e3x, int &e3y);
void eraseEnemy3(int &e3x, int &e3y);
void moveEnemy1(string &direction1, int &e1x, int &e1y, int &e1);
string cdirection1(string &direction1, int &e1y);
void moveEnemy2(string &direction2, int &e2x, int &e2y, int &e2);
string cdirection2(string &direction2, int &e2y);
void moveEnemy3(string &direction3, int &e3x, int &e3y, int &e3);
string cdirection3(string &direction3, int &e3x);
void Player(int &px, int &py);
void erasePlayer(int &px, int &py);
void movePlayerLeft(int &px, int &py, int &score, int &bx, int &by, int &playerHealth);
void movePlayerRight(int &px, int &py, int &score, int &bx, int &by, int &playerHealth);
void movePlayerUp(int &px, int &py, int &score, int &bx, int &by, int &playerHealth);
void movePlayerDown(int &px, int &py, int &score, int &bx, int &by, int &playerHealth);
void playerHealthDecrement(int &playerHealth, int &px, int &py, int &e1x, int &e1y, int &e2x,
int &e2y, int &e3x, int &e3y);
void loadCoordinatesFile(string filename, int &px, int &py, int &e1x, int &e1y, int &e2x, int
&e2y, int &e3x, int &e3y, int &score, int &playerHealth, int &enemyHealth, int &e1, int &e2,
int &e3);
void readCoordinatesFile(string filename, int &px, int &py, int &e1x, int &e1y, int &e2x, int
&e2y, int &e3x, int &e3y, int &score, int &playerHealth, int &enemyHealth, int &e1, int &e2,
int &e3);
void gameLoop(string filename, int &px, int &py, int &e1x, int &e1y, int &e2x, int &e2y, int
&e3x, int &e3y, int &score, int &playerHealth, int &enemyHealth, int &bx, int &by, string
&direction1, string &direction2, string &direction3, int &e1, int &e2, int &e3, int &kx, int
&ky);
string getField(string record, int field);
void printScore(int &score);
string setcolor(unsigned short color);
void moveFireRight(int &px, int &py, int &score, int &enemyHealth, int &e1, int &e2, int &e3);
void moveFireLeft(int &px, int &py, int &score, int &enemyHealth, int &e1, int &e2, int &e3);
```

```cpp
void printFireRight(int &px, int &py);
void printFireLeft(int &px, int &py);
void eraseFire(int &px, int &py);
string controls();
void Header();
string rules();
void bounsPill();
void printEnemyHealth(int enemyHealth);
void printPlayerHealth(int playerHealth);
void printBonusPill(int &bx, int &by);
void printKey(int &kx, int &ky);
void printPlayerCoordinates(int &px, int &py);
void printBonusPillCoordinates(int &bx, int &by);
string mainMenu();
void clearScreen();
void clearHeader();
void wrongOption();
bool pillCheck();
bool keyCheck();
bool bonusCheck();
string about();
```

```cpp
// Main Function
```

```cpp
main()
{
```

```cpp
    int px = 30, py = 1, e1x = 2, e1y = 1, e2x = 75, e2y = 21, e3x = 72, e3y = 16; //
Variables for player and enemy coordinates
    int score = 0;                                                                 // Variable
for storing score
    int enemyHealth = 75;                                                          // Variable
for storing enemies health
    int e1 = 0, e2 = 0, e3 = 0;                                                    //
Variables for counting number of hits to each enemy
    int playerHealth = 50;                                                         // Variable
for storing player health
    int bx, by;                                                                    //
Variables for bonus pill coordinates
    int kx, ky;                                                                    //
Variables for key coordinates
    string direction1 = "down";                                                    // Variable
for enemy1 direction
    string direction2 = "up";                                                      // Variable
for enemy2 direction
    string direction3 = "left";                                                    // Variable
for enemy3 direction
    string filename = "Coordinates.txt";                                           // Variable
for storing coordinates file name
```

```cpp
    // While loop for running whole system
```

```cpp
    system("cls"); // Clearing Screen
    while (true)
    {
```

```cpp
        Header();                   // Prints header
        string option = mainMenu(); // Storing option returned by user from main menu
```

```cpp
        if (option == "1") // If option is '1'
        {
            // Resetting all coordinates, health, score and directions
            px = 30, py = 1, e1x = 2, e1y = 1, e2x = 75, e2y = 21, e3x = 72, e3y = 16;
            score = 0;
            enemyHealth = 75, e1 = 0, e2 = 0, e3 = 0;
            playerHealth = 50;
            direction1 = "down";
            direction2 = "up";
            direction3 = "left";
```

```
            // Calling of gameLoop function to run the game
            gameLoop(filename, px, py, e1x, e1y, e2x, e2y, e3x, e3y, score, playerHealth,
enemyHealth, bx, by, direction1, direction2, direction3, e1, e2, e3, kx, ky);
        }
        else if (option == "2") // If option is '2'
        {
            // Returning previous game data from file into variables
            readCoordinatesFile(filename, px, py, e1x, e1y, e2x, e2y, e3x, e3y, score,
playerHealth, enemyHealth, e1, e2, e3);
```

```
            // Calling of gameLoop function to run the game
            gameLoop(filename, px, py, e1x, e1y, e2x, e2y, e3x, e3y, score, playerHealth,
enemyHealth, bx, by, direction1, direction2, direction3, e1, e2, e3, kx, ky);
        }
        else if (option == "3") // If option is '3'
        {
            // Clearing screen and printing header
            clearHeader();
```

```
            // Printing controls
            string control = controls();
            cout << control << endl;
        }
        else if (option == "4") // If option is '4'
        {
            // Clearing screen and printing header
            clearHeader();
```

```
            // Printing rules
            string rule = rules();
            cout << endl
                 << rule << endl;
        }
        else if (option == "5") // If option is '5'
        {
            // Clearing screen and printing header
            clearHeader();
```

```
            // Printing story
            string story = about();
            cout << story << endl;
        }
        else if (option == "6") // If option is '5'
        {
            return 0; // Terminating the program
        }
        else // If user enters invalid option
        {
            cout << endl
                 << "     Wrong Option Entered! \nEnter Option Again ... " << endl;
        }
        clearScreen(); // Taking any key and clearing screen
```

```
    }
}
```

```
// Function for running the whole game
```

```
void gameLoop(string filename, int &px, int &py, int &e1x, int &e1y, int &e2x, int &e2y, int
&e3x, int &e3y, int &score, int &playerHealth, int &enemyHealth, int &bx, int &by, string
&direction1, string &direction2, string &direction3, int &e1, int &e2, int &e3, int &kx, int
&ky)
{
```

```
    clearScreen();          // Taking any key and clearing screen
    maze();                 // Printing the maze
    Enemy1(e1x, e1y);       // Printing enemy1
    Enemy2(e2x, e2y);       // Printing enemy2
    Enemy3(e3x, e3y);       // Printing enemy3
    Player(px, py);         // Printing player
    printBonusPill(bx, by); // Printing bonus pill
```

```
    // While loop for running game
```

```
    while (true)
    {
        if (GetAsyncKeyState(VK_LEFT)) // If player presses left arrow key
        {
            movePlayerLeft(px, py, score, bx, by, playerHealth);
        }
        if (GetAsyncKeyState(VK_RIGHT)) // If player presses right arrow key
        {
            movePlayerRight(px, py, score, bx, by, playerHealth);
        }
        if (GetAsyncKeyState(VK_UP)) // If player presses up arrow key
        {
            movePlayerUp(px, py, score, bx, by, playerHealth);
        }
        if (GetAsyncKeyState(VK_DOWN)) // If player presses down arrow key
        {
            movePlayerDown(px, py, score, bx, by, playerHealth);
        }
        if (GetAsyncKeyState('X')) // If player presses x
        {
            printFireRight(px, py);
            moveFireRight(px, py, score, enemyHealth, e1, e2, e3);
        }
        if (GetAsyncKeyState('Z')) // If player presses z
        {
            printFireLeft(px, py);
            moveFireLeft(px, py, score, enemyHealth, e1, e2, e3);
        }
        if (GetAsyncKeyState(VK_LCONTROL)) // If player presses left contol key
        {
            gotoxy(1, 31);
            break;
        }
        printScore(score);                                                  // Printing
score
        printEnemyHealth(enemyHealth);                                      // Printing
Enemies health
        printPlayerHealth(playerHealth);                                    // Printing
player health
```

```
        printPlayerCoordinates(px, py);                                    // Printing
player coordinates
        printBonusPillCoordinates(bx, by);                                 // Printing
bonus pill coordinates
        playerHealthDecrement(playerHealth, px, py, e1x, e1y, e2x, e2y, e3x, e3y); //
Decreasing player health
        direction1 = cdirection1(direction1, e1y);                         // Changing
enemy1 direction
        moveEnemy1(direction1, e1x, e1y, e1);                              // Moving
enemy1
        direction2 = cdirection2(direction2, e2y);                         // Changing
enemy2 direction
        moveEnemy2(direction2, e2x, e2y, e2);                              // Moving
enemy2
        direction3 = cdirection3(direction3, e3x);                         // Changing
enemy3 direction
        moveEnemy3(direction3, e3x, e3y, e3);                              // Moving
enemy3
        Sleep(250);                                                        //
Controlling pace of the game
```

```
        if (playerHealth == 00) // If player health becomes 0
        {
            clearHeader();
            gotoxy(0, 15); // Go to the given coordinates
            setcolor(4);
            cout << "You Lose! Better Luck Next Time." << endl;
            abort(); // Terminating program
        }
        if (enemyHealth == 00 || pillCheck()) // If enemies health becomes zero or there are
no pills left
        {
            if (keyCheck()) // If there is no key in the maze
            {
                printKey(kx, ky); // Printing key
            }
        }
        if (bonusCheck()) // If there is no bonus pill in the maze
        {
            printBonusPill(bx, by); // Printing bouns pill
        }
        if (score > 1000) // If player grabs the key
        {
            clearHeader();
            gotoxy(0, 15); // Go to the given coordinates
            setcolor(4);
            cout << "You Win! You Have Successfully Escaped The Mystery Maze." << endl;
            abort(); // Terminating program
        }
```

```
        // Loading current game data into a file
        loadCoordinatesFile(filename, px, py, e1x, e1y, e2x, e2y, e3x, e3y, score,
playerHealth, enemyHealth, e1, e2, e3);
    }
}
```

```
void maze() // Function for printing maze
{
```

```
    setcolor(9);
    cout <<
"#############################################################################################
############                " << endl;
    cout <<
"||    .....o........                ..........o........             o          ||
          ||                 " << endl;
    cout <<
"||    %%%%%%%%%%%%%%    o           %%%%%%%%%%%%%%%%%%     |%|o      %%%%%    ||
          ||                  " << endl;
    cout <<
"||         |%|...|%|     .|%|       |%|.....o......|%|     |%|.      ..|%|.   ||
          ||                   " << endl;
    cout << "||    o    |%|. .|%|     .|%|    o  |%|o...    ....|%|   o
|%|.  o   ..|%|.   ||                       ||               " << endl;
    cout <<
"||                       o|%|       %%%%%%%    %%%%%%%               %%%%%    |||||||||||||||
||||||||||||                  " << endl;
    cout <<
"||        o|%|  o        .|%|         o                    |%|.   o           ||
          ||                 " << endl;
    cout <<
"||          %%%%%%%%               %%%%%%%%%%%%%%%    o    |%|o      %%%%%    ||
          ||         " << endl;
    cout <<
"||   o      ........|%|    o        |%|....o......         |%|.       o|%|.   ||
          ||   " << endl;
    cout <<
"||                 o|%|          o  |%|..       ..|%|                  |%|.   ||
          ||   " << endl;
    cout << "||   |%|..
|%|...  .|%|   |%|                   .o|%|    o      o|%|.  |%|.   |||||||||||||||||||||||||||
      " << endl;
    cout << "||   |%|..
|%|o..         %%%          o        ..|%|           .|%|.         ||                       ||
       " << endl;
    cout <<
"||   |%|..                 o    |%|.           %%%%%%           .|%|.         ||
          ||         " << endl;
    cout <<
"||   |%|.. .....o......         |%|...o..               .|%|    o|%|...o..    ||
          ||       " << endl;
    cout <<
"||   |%|.. %%%%%%%%%%%%         |%|%%%%%%%%%%%%%        o|%|    .|%|%%%%%%    ||
          ||         " << endl;
    cout << "||         ......o.....
o          ....o....  o         .|%|          o       ||                       ||           "
<< endl;
    cout <<
"||                                                                             |||||||||||||||
||||||||||||              " << endl;
    cout <<
"||   |%|.  |%|...|%|     %%%%%%%%%%%%%          |%|        |%|.   |%|         ||
          ||            " << endl;
    cout <<
"||   |%|.  |%|...|%|             o|%|   o    %%%%%%   o    |%|o   |%|         ||
          ||             " << endl;
    cout <<
"||   |%|.             o           |%|                      |%|.   |%|o        ||
          ||           " << endl;
    cout << "||   |%|.  %%%%%%%%%%%            |&|%%%%%%%%%%%%%       o
|%|.   |%|%%%%%    ||                       ||          " << endl;
```

```
    cout <<
"||         ...o.......            ..........o....                     o      ||
          ||            " << endl;
    cout <<
"||~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~||
          ||               " << endl;
    cout <<
"#############################################################################################
############               " << endl;
    cout <<
"||                                                                             ||
          ||       " << endl;
    cout << "||                            THE MYSTERY
MAZE                                 ||                         ||       " << endl;
    cout <<
"||                                                                             ||
          ||       " << endl;
    cout <<
"#############################################################################################
############        " << endl;
    setcolor(4);
    cout << endl
         << endl
         << "     -> Press Left Control Key To Return to Main Menu...";
}
```

```
void Header() // Function for printing header
{
```

```
    setcolor(9);
    cout << R"(
                 ___________  __    __    _______     ___      ___  ___  ___  ________  _____
______  _______   _______   ___  ___     ___      ___       __     ________    _______
                (" _   ")/" |  |
"\  /"     "|    |"  \    /"  ||"  \/"  |/"       )("     _   ")/"     "| /"      \
|"  \/"  |    |"  \    /"  |     /""\   ("       "\  /"     "|
                 )__/  \\__/(:  (__)  :)(: ______)     \   \  //   |
\   \  /(:   \___/  )__/  \\__/(: ______)|:        |
\   \  /      \   \  //   |    /    \   \___/   :)(: ______)
                    \\_
/    \/      \/  \/    |       /\\  \/.    |  \\  \/  \___  \       \\_
/    \/    |  |_____/    )  \\  \/       /\\  \/.    |   /' /\  \    /  ___/  \/    |
                    |.  |    //  __  \\  // ___)_     |:
\.         |  /   /    __/  \\      |.  |    // ___)_  //      /   /   /       |:
\.         |  //  __'  \  //  \__   // ___)_
                    \:  |   (:  (  )  :)(:      "|    |.  \    /:  | /   /    /"
\   :)     \:  |   (:      "||:  __   \  /   /        |.  \    /:  | /   /  \\  \(:   / "\
(:      "|
                     \__|    \__|  |__/  \_______)    |___|\__/|___||___/    (_______/       \
__|    \_______)|__|  \___)|___/         |___|\__/|___|(___/    \___)\_______) \_______)


```

```
    )" << endl;
}
```

```
string mainMenu() // Function for printing main menu
{
    setcolor(3);
    cout << "   ############";
    cout << endl
```

```cpp
         << "    MAIN MENU " << endl;
    cout << "   ############" << endl;
    string option;
    setcolor(4);
    cout << endl;
    cout << "   1. New Game" << endl;
    cout << "   2. Resume Game" << endl;
    cout << "   3. Controls" << endl;
    cout << "   4. Rules" << endl;
    cout << "   5. Story Of The Game" << endl;
    cout << "   6. Exit" << endl
         << endl;
    cout << "   Your option is .... ";
    getline(cin >> ws, option);
    return option;
}
```

```cpp
// Function for loading game data into the coordinates file
```

```cpp
void loadCoordinatesFile(string filename, int &px, int &py, int &e1x, int &e1y, int &e2x, int
&e2y, int &e3x, int &e3y, int &score, int &playerHealth, int &enemyHealth, int &e1, int &e2,
int &e3)
{
```

```cpp
    fstream file;
    file.open(filename, ios::out);
```

```cpp
    file << px << "," << py << "," << e1x << "," << e1y << "," << e2x << "," << e2y << "," <<
e3x << "," << e3y << "," << score << "," << playerHealth << "," << enemyHealth << "," << e1 <<
"," << e2 << "," << e3;
    file.close();
}
```

```cpp
// Function for resetting game data from a file
```

```cpp
void readCoordinatesFile(string filename, int &px, int &py, int &e1x, int &e1y, int &e2x, int
&e2y, int &e3x, int &e3y, int &score, int &playerHealth, int &enemyHealth, int &e1, int &e2,
int &e3)
{
```

```cpp
    fstream file;
    file.open(filename, ios::in);
    string line;
```

```cpp
    while (getline(file, line))
    {
```

```cpp
        if (line != "")
        {
```

```cpp
            px = stoi(getField(line, 1));
            py = stoi(getField(line, 2));
            e1x = stoi(getField(line, 3));
            e1y = stoi(getField(line, 4));
            e2x = stoi(getField(line, 5));
            e2y = stoi(getField(line, 6));
            e3x = stoi(getField(line, 7));
            e3y = stoi(getField(line, 8));
            score = stoi(getField(line, 9));
            playerHealth = stoi(getField(line, 10));
            enemyHealth = stoi(getField(line, 11));
            e1 = stoi(getField(line, 12));
            e2 = stoi(getField(line, 13));
            e3 = stoi(getField(line, 14));
        }
    }
    file.close();
}
```

```
string getField(string record, int field) // Function for returning the required data from a
file depending upon the comma number
{
```

```
    int commaCount = 1;
    string item;
    for (int x = 0; x < record.length(); x++)
    {
```

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

```
            commaCount = commaCount + 1;
        }
        else if (commaCount == field)
        {
```

```
            item = item + record[x];
        }
    }
    return item;
}
```

```
void clearScreen()
{ // Function for Clearing screen
```

```
    setcolor(8);
    cout << endl
         << "    Press Any Key to Continue.." << endl;
    getch();
    system("cls");
}
```

```
void clearHeader()
{ // Function for clearing screen and printing header
```

```
    system("cls");
    Header();
}
```

```
void wrongOption()
{ // Function for clearing screen and printing header if user enters wrong option
```

```
    cout << "    Wrong Option Entered..." << endl;
    clearScreen();
    Header();
}
```

```
char getCharAtxy(short int x, short int y)
{ // Function for checking character at given coordinates
```

```
    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 gotoxy(int x, int y)
{ // Function for going to given coordinates
```

```
    COORD coordinates;
    coordinates.X = x;
    coordinates.Y = y;
    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), coordinates);
}
```

```
void Enemy1(int &e1x, int &e1y)
{ // Function for printing enemy1
```

```
    gotoxy(e1x, e1y);
    setcolor(5);
    cout << "[1]";
}
```

```
void eraseEnemy1(int &e1x, int &e1y)
{ // Function for erasing enemy1
```

```
    gotoxy(e1x, e1y);
    cout << "   ";
}
```

```
void moveEnemy1(string &direction1, int &e1x, int &e1y, int &e1)
{ // Function for moving enemy1
    if (e1 < 5)
    {
        eraseEnemy1(e1x, e1y);
        if (direction1 == "down")
        {
            e1y = e1y + 1;
        }
        if (direction1 == "up")
        {
            e1y = e1y - 1;
        }
        Enemy1(e1x, e1y);
    }
    else
    {
        eraseEnemy1(e1x, e1y);
    }
}
```

```
string cdirection1(string &direction1, int &e1y)
{ // Function for changing enemy1 direction
```

```
    if (direction1 == "down" && e1y >= 21)
    {
        direction1 = "up";
    }
    if (direction1 == "up" && e1y <= 1)
```

```cpp
    {
        direction1 = "down";
    }
    return direction1;
}
```

```cpp
void Enemy2(int &e2x, int &e2y)
{ // Function for printing enemy2
```

```cpp
    gotoxy(e2x, e2y);
    setcolor(5);
    cout << "[2]";
}
```

```cpp
void eraseEnemy2(int &e2x, int &e2y)
{ // Function for erasing enemy2
```

```cpp
    gotoxy(e2x, e2y);
    cout << "   ";
}
```

```cpp
void moveEnemy2(string &direction2, int &e2x, int &e2y, int &e2)
{ // Function for moving enemy3
    if (e2 < 5)
    {
        eraseEnemy2(e2x, e2y);
        if (direction2 == "up")
        {
            e2y = e2y - 1;
        }
        if (direction2 == "down")
        {
            e2y = e2y + 1;
        }
        Enemy2(e2x, e2y);
    }
    else
    {
        eraseEnemy2(e2x, e2y);
    }
}
```

```cpp
string cdirection2(string &direction2, int &e2y)
{ // Function for changing enemy2 direction
```

```cpp
    if (direction2 == "up" && e2y <= 1)
    {
        direction2 = "down";
    }
    if (direction2 == "down" && e2y >= 21)
    {
        direction2 = "up";
    }
    return direction2;
}
```

```cpp
void Enemy3(int &e3x, int &e3y)
{ // Function for printing enemy3
```

```cpp
    gotoxy(e3x, e3y);
    setcolor(5);
    cout << "[3]";
}
```

```
void eraseEnemy3(int &e3x, int &e3y)
{ // Function for erasing enemy3
```

```
    gotoxy(e3x, e3y);
    cout << "   ";
}
```

```
void moveEnemy3(string &direction3, int &e3x, int &e3y, int &e3)
{ // Function for moving enemy3
    if (e3 < 5)
    {
        eraseEnemy3(e3x, e3y);
        if (direction3 == "left")
        {
            e3x = e3x - 2;
        }
        if (direction3 == "right")
        {
            e3x = e3x + 2;
        }
        Enemy3(e3x, e3y);
    }
    else
    {
        eraseEnemy3(e3x, e3y);
    }
}
```

```
string cdirection3(string &direction3, int &e3x)
{ // Function for changing enemy3 direction
```

```
    if (direction3 == "left" && e3x <= 6)
    {
        direction3 = "right";
    }
    if (direction3 == "right" && e3x >= 72)
    {
        direction3 = "left";
    }
    return direction3;
}
```

```
void Player(int &px, int &py)
{ // Function for printing player
```

```
    gotoxy(px, py);
    setcolor(2);
    cout << "(P)";
}
```

```
void erasePlayer(int &px, int &py)
{ // Function for erasing player
```

```
    gotoxy(px, py);
    cout << "   ";
}
```

```
void movePlayerLeft(int &px, int &py, int &score, int &bx, int &by, int &playerHealth)
```

```
{ // Function for moving player left
```

```
    if (getCharAtxy(px - 1, py) == ' ')
    {
        erasePlayer(px, py);
        px = px - 1;
        Player(px, py);
    }
    else if (getCharAtxy(px - 1, py) == '.')
    {
        erasePlayer(px, py);
        px = px - 1;
        Player(px, py);
        score += 1;
    }
    else if (getCharAtxy(px - 1, py) == '*')
    {
        erasePlayer(px, py);
        px = px - 1;
        Player(px, py);
        playerHealth += 5;
        printBonusPill(bx, by);
    }
    else if (getCharAtxy(px - 1, py) == 'o')
    {
        erasePlayer(px, py);
        px = px - 1;
        Player(px, py);
        playerHealth -= 5;
    }
    else if (getCharAtxy(px - 1, py) == '$')
    {
        erasePlayer(px, py);
        px = px - 1;
        Player(px, py);
        score += 1000;
    }
}
```

```
void movePlayerRight(int &px, int &py, int &score, int &bx, int &by, int &playerHealth)
{ // Function for moving player right
```

```
    if (getCharAtxy(px + 3, py) == ' ')
    {
        erasePlayer(px, py);
        px = px + 1;
        Player(px, py);
    }
    else if (getCharAtxy(px + 3, py) == '.')
    {
        erasePlayer(px, py);
        px = px + 1;
        Player(px, py);
        score += 1;
    }
    else if (getCharAtxy(px + 3, py) == '*')
    {
        erasePlayer(px, py);
        px = px + 1;
        Player(px, py);
        playerHealth += 5;
        printBonusPill(bx, by);
    }
    else if (getCharAtxy(px + 3, py) == 'o')
    {
        erasePlayer(px, py);
        px = px + 1;
        Player(px, py);
        playerHealth -= 5;
    }
    else if (getCharAtxy(px + 3, py) == '$')
    {
        erasePlayer(px, py);
        px = px + 1;
        Player(px, py);
        score += 1000;
    }
}
```

```
void movePlayerUp(int &px, int &py, int &score, int &bx, int &by, int &playerHealth)
{ // Function for moving player up
```

```
    if (getCharAtxy(px, py - 1) == ' ' && getCharAtxy(px + 1, py - 1) == ' ' && getCharAtxy(px
+ 2, py - 1) == ' ')
    {
        erasePlayer(px, py);
        py = py - 1;
        Player(px, py);
    }
    else if (getCharAtxy(px, py - 1) == '.' || getCharAtxy(px + 1, py - 1) == '.' ||
getCharAtxy(px + 2, py - 1) == '.')
    {
        erasePlayer(px, py);
        py = py - 1;
        Player(px, py);
        score += 1;
    }
    else if (getCharAtxy(px, py - 1) == '*' || getCharAtxy(px + 1, py - 1) == '*' ||
getCharAtxy(px + 2, py - 1) == '*')
    {
        erasePlayer(px, py);
        py = py - 1;
        Player(px, py);
        playerHealth += 5;
        printBonusPill(bx, by);
    }
    else if (getCharAtxy(px, py - 1) == 'o' || getCharAtxy(px + 1, py - 1) == 'o' ||
getCharAtxy(px + 2, py - 1) == 'o')
    {
        erasePlayer(px, py);
        py = py - 1;
        Player(px, py);
        playerHealth -= 5;
    }
    else if (getCharAtxy(px, py - 1) == '$' || getCharAtxy(px + 1, py - 1) == '$' ||
getCharAtxy(px + 2, py - 1) == '$')
    {
        erasePlayer(px, py);
        py = py - 1;
        Player(px, py);
        score += 1000;
    }
```

```cpp
}

void movePlayerDown(int &px, int &py, int &score, int &bx, int &by, int &playerHealth)
{ // Function for moving player down

    if (getCharAtxy(px, py + 1) == ' ' && getCharAtxy(px + 1, py + 1) == ' ' && getCharAtxy(px
+ 2, py + 1) == ' ')
    {
        erasePlayer(px, py);
        py = py + 1;
        Player(px, py);
    }
    else if (getCharAtxy(px, py + 1) == '.' || getCharAtxy(px + 1, py + 1) == '.' ||
getCharAtxy(px + 2, py + 1) == '.')
    {
        erasePlayer(px, py);
        py = py + 1;
        Player(px, py);
        score += 1;
    }
    else if (getCharAtxy(px, py + 1) == '*' || getCharAtxy(px + 1, py + 1) == '*' ||
getCharAtxy(px + 2, py + 1) == '*')
    {
        erasePlayer(px, py);
        py = py + 1;
        Player(px, py);
        playerHealth += 5;
        printBonusPill(bx, by);
    }
    else if (getCharAtxy(px, py + 1) == 'o' || getCharAtxy(px + 1, py + 1) == 'o' ||
getCharAtxy(px + 2, py + 1) == 'o')
    {
        erasePlayer(px, py);
        py = py + 1;
        Player(px, py);
        playerHealth -= 5;
    }
    else if (getCharAtxy(px, py + 1) == '$' || getCharAtxy(px + 1, py + 1) == '$' ||
getCharAtxy(px + 2, py + 1) == '$')
    {
        erasePlayer(px, py);
        py = py + 1;
        Player(px, py);
        score += 1000;
    }
}

void playerHealthDecrement(int &playerHealth, int &px, int &py, int &e1x, int &e1y, int &e2x,
int &e2y, int &e3x, int &e3y)
{ // Function for checking conditions and decreasing player health

    if (getCharAtxy(px, py + 1) == '[' || getCharAtxy(px + 1, py + 1) == '[' || getCharAtxy(px
+ 2, py + 1) == '[')
    {

        playerHealth -= 5;
    }
    else if (getCharAtxy(px, py - 1) == ']' || getCharAtxy(px + 1, py - 1) == ']' ||
getCharAtxy(px + 2, py - 1) == ']')
    {
        playerHealth -= 5;
    }
    else if (getCharAtxy(px + 3, py) == '[')
    {
```

```
        playerHealth -= 5;
    }
    else if (getCharAtxy(px - 1, py) == ']')
    {
```

```
        playerHealth -= 5;
    }
    else if (py == e3y)
    {
        if (px == e3x || px + 1 == e3x || px + 2 == e3x || px == e3x + 2 || px + 1 == e3x + 2
|| px + 2 == e3x + 2)
        {
            playerHealth = 0;
        }
    }
    else if (py == e1y)
    {
        if (px == e1x || px + 1 == e1x || px + 2 == e1x || px == e1x + 2 || px + 1 == e1x + 2
|| px + 2 == e1x + 2)
        {
            playerHealth = 0;
        }
    }
    else if (py == e2y)
    {
        if (px == e2x || px + 1 == e2x || px + 2 == e2x || px == e2x + 2 || px + 1 == e2x + 2
|| px + 2 == e2x + 2)
        {
            playerHealth = 0;
        }
    }
}
```

```
void printScore(int &score)
{ // Function for printing score
```

```
    gotoxy(87, 25);
    setcolor(9);
    cout << "SCORE = " << score;
}
```

```
void printEnemyHealth(int enemyHealth)
{ // Function for printing enemy health
```

```
    gotoxy(84, 7);
    setcolor(9);
    cout << "Enemies Health";
    gotoxy(90, 8);
    cout << enemyHealth;
}
```

```
void printPlayerHealth(int playerHealth)
{ // Function for printing player health
```

```
    gotoxy(85, 2);
    setcolor(9);
    cout << "Player Health";
    gotoxy(90, 3);
    cout << playerHealth;
}
```

```
void printPlayerCoordinates(int &px, int &py)
{ // Function for printing player coordinates
```

```
    gotoxy(83, 12);
    setcolor(9);
    cout << "Player Coordinates";
    gotoxy(88, 13);
    cout << "Px = " << px;
    gotoxy(88, 14);
    cout << "Py = " << py;
}
```

```
void printBonusPillCoordinates(int &bx, int &by)
{ // Function for printing bonus pill coordinates
```

```
    gotoxy(84, 18);
    setcolor(9);
    cout << "Bonus Coordinates";
    gotoxy(88, 19);
    cout << "Bx = " << bx;
    gotoxy(88, 20);
    cout << "By = " << by;
}
```

```
string setcolor(unsigned short color)
{ // Function for changing color
```

```
    HANDLE hcon = GetStdHandle(STD_OUTPUT_HANDLE);
    SetConsoleTextAttribute(hcon, color);
    return "";
}
```

```
void bounsPill()
{ // Function for printing bonus pill
```

```
    gotoxy(25, 9);
    setcolor(4);
    cout << "*";
}
```

```
void printBonusPill(int &bx, int &by)
{ // Function for printing bonus pill
```

```
    srand(time(0));
    bx = rand() % 75;
    srand(time(0));
    by = rand() % 20;
    while (!(getCharAtxy(bx, by) == ' ' && bx > 1 && by > 0))
    {
```

```
        srand(time(0));
        bx = rand() % 75;
        srand(time(0));
        by = rand() % 20;
    }
    gotoxy(bx, by);
    setcolor(4);
    cout << '*';
}

void printKey(int &kx, int &ky)
{ // Function for printing key

    srand(time(0));
    kx = rand() % 75;
    srand(time(0));
    ky = rand() % 20;
    while (!(getCharAtxy(kx, ky) == ' ' && kx > 1 && ky > 0))
    {

        srand(time(0));
        kx = rand() % 75;
        srand(time(0));
        ky = rand() % 20;
    }

    gotoxy(kx, ky);
    setcolor(6);
    cout << '$';
}

void printFireRight(int &px, int &py)
{ // Function for printing right side fire

    gotoxy(px + 4, py);
    setcolor(4);
    cout << ">";
}

void printFireLeft(int &px, int &py)
{ // Function for printing left side fire

    gotoxy(px - 1, py);
    setcolor(4);
    cout << ">";
}

void eraseFire(int &px, int &py)
{ // Function for erasing fire

    gotoxy(px + 4, py);
    cout << " ";
}

void moveFireRight(int &px, int &py, int &score, int &enemyHealth, int &e1, int &e2, int &e3)
{ // Function for moving fire right

    for (int i = 2; i < 81; i++)
    {
        for (int j = 1; j < 24; j++)
        {

            if (getCharAtxy(i, j) == '>')

            {
                if (getCharAtxy(i + 1, j) == ' ')
                {
                    gotoxy(i, j);
                    cout << " ";
                    gotoxy(i + 1, j);
                    cout << ">";
                }
                else if (getCharAtxy(i + 1, j) == '[')
                {
                    if (getCharAtxy(i + 2, j) == '1')
                    {
                        score += 2;
                        enemyHealth -= 5;
                        e1++;
                        gotoxy(i, j);
                        cout << " ";
                    }
                    if (getCharAtxy(i + 2, j) == '2')
                    {
                        score += 2;
                        enemyHealth -= 5;
                        e2++;
                        gotoxy(i, j);
                        cout << " ";
                    }
                    if (getCharAtxy(i + 2, j) == '3')
                    {
                        score += 2;
                        enemyHealth -= 5;
                        e3++;
                        gotoxy(i, j);
                        cout << " ";
                    }
                }
                else if (getCharAtxy(i + 1, j) == '#' || getCharAtxy(i + 1, j) == '|' ||
getCharAtxy(i + 1, j) == '%' || getCharAtxy(i + 1, j) == '.' || getCharAtxy(i + 1, j) == 'o'
|| getCharAtxy(i + 1, j) == '*')
                {
                    gotoxy(i, j);
                    cout << " ";
                }
            }
        }
    }
}

void moveFireLeft(int &px, int &py, int &score, int &enemyHealth, int &e1, int &e2, int &e3)
{ // Function for moving fire left

    for (int i = 2; i < 81; i++)
    {
        for (int j = 1; j < 24; j++)
        {

            if (getCharAtxy(i, j) == '>')

            {
                if (getCharAtxy(i - 1, j) == ' ')
                {
                    gotoxy(i, j);
                    cout << " ";
                    gotoxy(i - 1, j);
                    cout << ">";
                }
                else if (getCharAtxy(i - 1, j) == ']')
                {
                    if (getCharAtxy(i - 2, j) == '1')
                    {
                        score += 2;
                        enemyHealth -= 5;
                        e1++;
                        gotoxy(i, j);
                        cout << " ";
                    }
                    if (getCharAtxy(i - 2, j) == '2')
                    {
                        score += 2;
                        enemyHealth -= 5;
                        e2++;
                        gotoxy(i, j);
                        cout << " ";
                    }
                    if (getCharAtxy(i - 2, j) == '3')
                    {
                        score += 2;
                        enemyHealth -= 5;
                        e3++;
                        gotoxy(i, j);
                        cout << " ";
                    }
                }
                else if (getCharAtxy(i - 1, j) == '#' || getCharAtxy(i - 1, j) == '|' ||
getCharAtxy(i - 1, j) == '%' || getCharAtxy(i - 1, j) == '.' || getCharAtxy(i - 1, j) == 'o'
|| getCharAtxy(i - 1, j) == '*')
                {
                    gotoxy(i, j);
                    cout << " ";
                }
            }
        }
    }
}
```

```
string controls()
{ // Function for printing controls
```

```
    setcolor(3);
    cout << endl;
    cout << "  ###########" << endl;
    cout << "   CONTROLS" << endl;
    cout << "  ###########" << endl
         << endl
         << endl;
    setcolor(4);
    cout << "  -> " << setw(20) << left << "Move Player Up"
         << "---------------" << setw(20) << right << "Arrow Key (UP)" << endl;
    cout << "  -> " << setw(20) << left << "Move Player Down"
         << "---------------" << setw(20) << right << "Arrow Key (DOWN)" << endl;
    cout << "  -> " << setw(20) << left << "Move Player Left"
         << "---------------" << setw(20) << right << "Arrow Key (LEFT)" << endl;
    cout << "  -> " << setw(20) << left << "Move Player Right"
         << "---------------" << setw(20) << right << "Arrow Key (RIGHT)" << endl;
    cout << "  -> " << setw(20) << left << "Shoot To Right"
```

```
         << "---------------" << setw(20) << right << "X" << endl;
    cout << "  -> " << setw(20) << left << "Shoot To Left"
         << "---------------" << setw(20) << right << "Z" << endl
         << endl;
```

```
    return "    These are the controls of the game.";
}
```

```
string rules()
{ // Function for printing rules
```

```
    setcolor(3);
    cout << "   ########" << endl;
    cout << "    RULES" << endl;
    cout << "   ########" << endl;
    setcolor(4);
    cout << endl
         << "   -> The player has a total of 50 health points." << endl;
    cout << endl
         << "   -> There are 3 enemies in the game." << endl;
    cout << "   -> Each enemy has 25 health points which totals to 75 enemy health points." <<
endl;
    cout << endl
         << "   -> Collect as many pills (.) as you can. " << endl;
    cout << "   -> Each pill increases the score by 1." << endl;
    cout << endl
         << "   -> Try to avoid the unusual looking pills (o)." << endl;
    cout << "   -> Each unusual looking pill decreases the player health by 5." << endl;
    cout << endl
         << "   -> A bonus pill (*) appears randomly in the maze." << endl;
    cout << "   -> Collect the bonus pill to increase player health by 5." << endl;
    cout << endl
         << "   -> Do not get close to an enemy." << endl;
    cout << "   -> Being close to an enemy decreases player health by 5." << endl;
    cout << endl
         << "   -> Avoid contact with an enemy." << endl;
    cout << "   -> Contact with an enemy decreases player health completely thus making you
lose the game." << endl;
    cout << endl
         << "   -> After an enemy is killed in a position, that specific position becomes a
bomb." << endl;
    cout << "   -> If you go to a bomb area , you will die and lose the game" << endl;
    cout << endl
         << "   -> The player can shoot left or right to kill enemies." << endl;
    cout << "   -> Each hit with an enemy decreases that enemy's health by 5 and increases
score by 2." << endl;
    cout << "   -> 5 hits to an enemy kils it." << endl;
    cout << endl
         << "   -> If the player health decreases to 0 , you lose the game." << endl;
    cout << endl
         << "   -> If you collect all the pills or kill all enemies , a key ($) appears in the
maze." << endl;
    cout << "   -> Collect the key to win the game and get out of the mystery maze." << endl;
```

```
    return "     These are the rules of the game.";
}
```

```
bool pillCheck()
{ // Function for checking pill in the maze
```

```
    for (int x = 2; x < 81; x++)
    {

        for (int y = 1; y < 24; y++)
        {

            if (getCharAtxy(x, y) == '.')
            {

                return false;
            }
        }
    }
    return true;
}

bool keyCheck()
{ // Function for checking key in the maze

    for (int x = 2; x < 81; x++)
    {

        for (int y = 1; y < 24; y++)
        {

            if (getCharAtxy(x, y) == '$')
            {

                return false;
            }
        }
    }
    return true;
}

bool bonusCheck()
{ // Function for checking bonus pill in the maze

    for (int x = 2; x < 81; x++)
    {

        for (int y = 1; y < 24; y++)
        {

            if (getCharAtxy(x, y) == '*')
            {

                return false;
            }
        }
    }
    return true;
}

string about()
{ // Function for printing story

    setcolor(3);
    cout << "   #######" << endl;
    cout << "    STORY" << endl;
    cout << "   #######" << endl;
    setcolor(4);
    cout << endl;
    cout << R"(
    The mystery maze is a game in which the player finds himself trapped inside a maze amidst enemies
    and little bombs. He has to escape the maze but in order to do so he needs a key. He can only see
    the key when he has killed all his enemies or collected all the pills inside the maze. With each
    step, he faces danger, evading enemies and bombs. It's a thrilling adventure that tests his skills
    and strategic thinking. Can he find the key and make it out of the maze filled with mysteries?
    Only time will tell!
    )" << endl;
```

```
    return "    This is the Story of the game.";
}
```