Final FOP Project

Matin Amirpanah far[*]

student NO:40212358003

Nima Makhmali[†]

student NO:40212358035

February 29, 2024

---

[*]email: matin.a0789@gmail.com

[†]email: nmkhmly790@gmail.com

Contents

Part I
Introducing the game

1   How to Play

The game has a menu, You can choose between the options
1.Continue game If you choose this, You can continue the saved game
2.New game If you choose this, a new game will be start. you have to set
ground size then tou have to set your end point then you see the game ground.
3.Game settings If you choose this, you see the option menu. In option menu
you can change the spaceships character or enemys character.
4.Quit game If you choose this, you can exit the game.

The playing ground is a size*size matrix.The character of the ship is in
the lowest row of the matrix and enemies attack from the top of the matrix
with the bottom so you have to destroyed enemies with down row in the
keyboard or you can move right or left in the ground with right row or left
row.you can pause the game and see the menu if you click on the up row key.

If you destoried the enemy you receive the enemys point but if the if the
enemy reaches the last line, your health will decrease.

If your score reaches the limit, a "You win" message will be printed but
if your health drops to zero, a "game over" message will be printed.

2   Errors and Warnings

If you enter an undefined character for move the spaceship, nothing special
will happen.
If the character leaves the playing ground, it enters the playing grand from
the other side.

Part II
Code description

1 libraries

We have used the following libraries in this code

1.1 iostream

This library is a requirement for writing any c++ code.

1.2 stdlib.h

We used this library so that we can use "rand()" and "srand()" functions.

1.3 ctime

By adding this library, we were able to use the "sleep()" function for interrupts.

1.4 fstream

Adding this library is so that we can create a file and use its commands like "fstream" or "ios::binary".

1.5 vector

We used this library to be able to use vector and its functions in the code, such as "push_back".

2 Defines

In this code, we have used defines for colors such as blue, green, black, etc., to color and also to bold our output text, whose ascii codes we have obtained by searching the internet, which of course We also used it in the mini project and we have also put 4 defines for the keys of our game in the code.

3   Enumertions

3.1   condition

This enum determines what is included in the map.

4   Structures

We have used the following structs in this code.

4.1   enemyStruct

This struct stores data of the enemies such as condition and size.

4.2   spaceShipStruct

This struct stores data of the spaceship such as condition and size.

4.3   bullets

This struct stores data of the bullets in the map such as condition.

4.4   graundStruct

This struct stores all data of the game ground.

5   Functions

Our code functions are as follows

5.1   main

This function is actually our main function, and first it calls the menu function, then it checks the condition of the heal of the ship, whether the heal of the spaceship is more than 0 or not. Another task of this function is to print the checkerboard of the game, which calls the grandDraw function Then it does the work of moving the enemies and the spaceship and saving the game information.

5.2   grandDraw

This function has the task of printing the game page, if we select newGame in the menu, it first takes a number from the user and prints the number of rows and columns equal to that number. At the top of the game page, we have the name of the game, Final Fight, and the heal ship, in this function, we call the horizontalLineDraw function to print the lines.


5.3   horizontalLineDraw

This function has the task of printing the horizontal lines of the game map by using two for loops

5.4   menu

Our menu function, what it displays, consists of four sections, which we first use a do while loop to display once, and then according to the definitions at the top of the code using the ascii code, the up and down buttons and left and we have defined correctly, we can choose the options, by selecting any of the options, we will enter the switch and case section of this function, each of which applies commands.
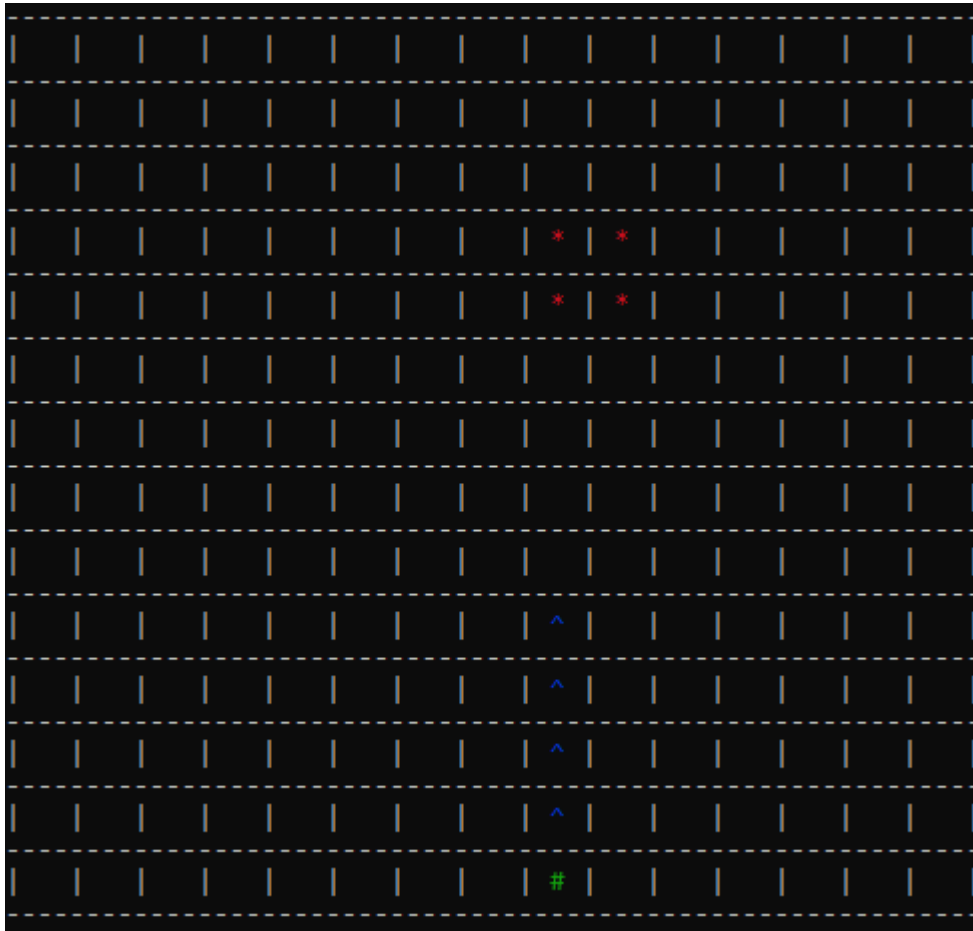
5.5   newGame

If the user selects the second option i.e. newgame in the game menu, this
function will be called and at first it will ask the user for the number of
rows and columns of the game screen and after waiting for the user to enter
the target point after entering these two numbers of the game screen will be
printed.

5.6   gameSetting

If the user selects the second option in the game menu, i.e. game setting, this
function will be called. When it is selected, three options will be printed on
the screen, which include changing the character of the ship and the enemy
and returning to the menu. If we choose any of the two options to change
the character, the system will ask us for the new character and then change

it.

5.7   move

Using a do while loop, this function first receives commands from the user once, then by using switch case and the right, left, down and up keys, whose codes have been defined as define at the top of the program, the commands are sent to the ship. If we select the right or left key, according to the program code, the current position of the ship will be empty and its line will be moved by one unit, and if we select the down key, the shoot function will be called.

5.8   shoot

In the shoot function, we create a new variable called newbullet using the bullet structure, and position the ship's beam according to the two variables x and y that we introduced in the structure. Finally, we use the push-back function, which is in the vector class.

5.9   save

We have used the fstream library to use this function. In this function, we first create a file, of course, the information in this file is stored in binary form because we have used the structure, and to transfer information from the structure to the file, this command must be used (ios::binary). Then we have used the flush command, which puts the information in the file part by part

5.10   load

The structure of this function is the same as the save function. This function reads the saved information of the game, of course, this is a binary operation with the same file "saveFile.bin". When we select continue game option in the menu, since the information was saved in the file in a piece by piece, from where the game left off. He starts to continue If the user has exited the game, the file is also deleted, so there is no information to display. This discussion, which was said to continue the game after reading the file, continues the game with those for loops and the new command.

Part III
links and references

1   Links

project's GitHub link

https://github.com/Matin0789/FinalFight.git

Ask question on StackOverflow

https://stackoverflow.com/questions/78060183/declare-array-in-structure-with-specified-size-in-c

2   References

https://www.geeksforgeeks.org/c-plus-plus/
https://www.programiz.com/cpp-programming