# C PROGRAM REPORT: SUBMARINE WAR

**ABSTRACT**

With the development of economy and society and the development of social rhythm, people are facing more and more pressure. There is an urgent need for some ways to release themselves. The popularity of computer mobile phones in our country has been quite high, so more and more people use these electronic products to release themselves. Some of the small games are meeting people's needs and are welcomed by more and more people.

Submarine warfare games first appeared in the 1980s and were once popular. Our work not only inherits its many advantages, but also extends its functions, enriches the game experience, more adapts to the requirements of the times, and embodies the design idea of keeping pace with the times.

**1 INTRODUCTION AND PROBLEM STATEMENT**

Submarine war game, a total of eight customs. Different levels, game maps, enemy submarines appear in different types. There are up to four enemy submarines on the map. After killing, new submarines will appear until four are supplemented. When killing a certAIn number of submarines, new enemy submarines will not increase. After killing all the submarines, we won the pass. Our submarines have also been resurrected several times, and when they are used up, they fAIl.The user can adjust the speed of game according to there need.

To achieve our goal, we use the Windows 1064 Bit Operating System and VS2019 Compilation.

There are three problems that we have to face.

The first is every moving object in the game moves in a specific range to prevent data from crossing the boundary. In this game, many control variables are in bounded areas, some correspond to the boundary of moving objects, and some correspond to the limited number of operations and storage. Once these data cross the boundaries, it is likely to lead to system errors, the game can not work properly, greatly affecting the player's game experience. Therefore, it is very necessary to prevent data from crossing boundaries.

The second is that four types of AI-Submarines can appear randomly and move freely and fire bullets at random. The reason why we set up such a problem is to improve the balance of the game, so that players can enjoy a better game experience.

The last is that the game interface can display the current game speed, score, the number of remAIning enemies and the remAIning life in real time. It is necessary to show the game state date in real time during the progress of playing the game, which means we might have to design many data windows to exchange information with players.

**2 GROUP DIVISION**

**3 ANALYSIS**

First of all, when facing the problem, we put forward two key points and imaginary analytical methods.

Point 1: Every moving object in the game moves in a specific range to prevent data crossing the boundary.

Analysis 1: As for moving objects, we use the form of two-dimensional array to express the position of each object. The change of position is realized by the regular change of array coordinates. Because of the limitation of array, the object can be prevented from moving beyond the boundaries. Team instant variables, we use the idea of counters and conditional judgment, using a multi-threaded approach to deal with them, so as to prevent data from crossing the border.

Point 2: How to divide the game into several execution blocks to reduce the complexity of the algorithm.

Analysis 2: According to the needs of game functions, the program can be divided into three categories. A class of game auxiliary functions. A bullet system, a submarine system. Because the functions of these three plates are relatively independent, the cross-linked data can be transmitted through nodes, which is simple and convenient. At the same time, this classification is also suitable for the division of labor.

Next, after analysis, we put forward the following four requirements for the game.

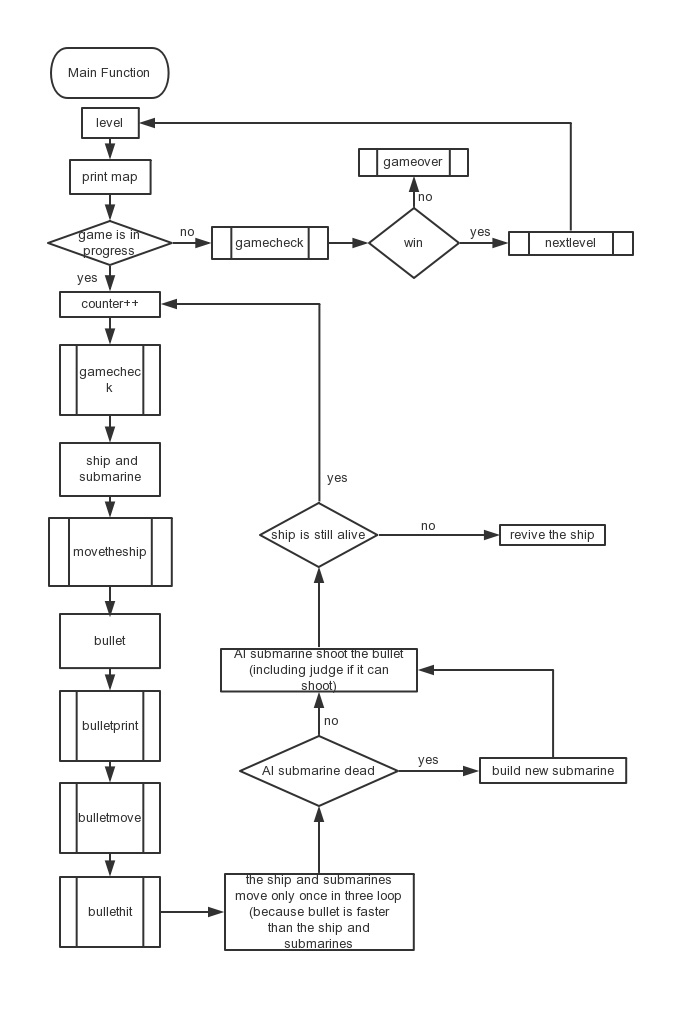
Requirement 1: Keyboard message(coordinates , X key and direction keys and so on). The user's operation is executed by keyboard input. We must ensure that the user's input has an accurate and good execution effect. In addition, cooling time should be set to prevent button sticking.

Requirements 2: the functions of moving AI-ship. We design four types of AI-ship. These AI submarines can be randomly generated in many locations, and can also randomly adjust the direction of motion and launch bullets to attack warships. At the same time, it is necessary to control the maximum number of submarines appearing on the screen at a certain time. When the number is less than that, the submarine can regenerate.

Requirement 3: the functions of moving ship, which is controlled by player. Warships can move freely on the sea and fire bullets according to the player's instructions (keyboard information). It is necessary to set a maximum life value. When the life value is smaller than that, the warship can be regenerated if destroyed. On the contrary, it can not be regenerated, and the player loses the game. We also designed several warships with different performance and appearance for players to choose from.

Requirements 4: the functions of bullets. The initial process of bullet launching depends on reading the coordinates of submarines and warships. The trajectories of warship bullets and submarine bullets are planned respectively. In addition, when a bullet hits a warship or submarine, it is necessary to perform clearance of multiple objects and corresponding music.

**4 DESIGN**

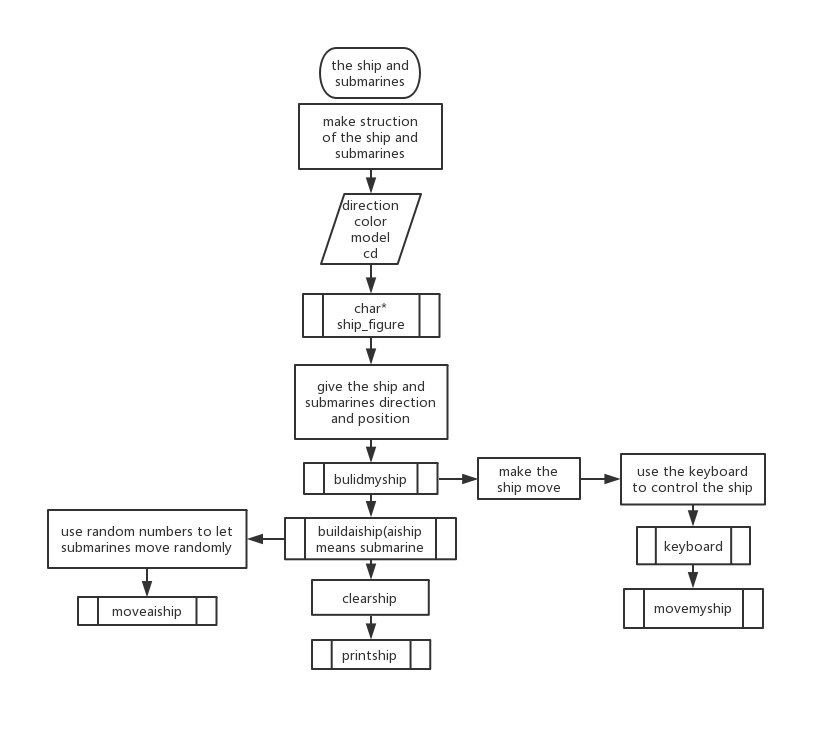
****

Picture 4.1 The design of the whole program

As you can see in the picture, the whole process is a half dead cycle. At first, the system automatically opens the map, and then opens the dead cycle, unless there is external interference. In the half dead cycle, we define some important global variable and important structures, such as Ship Structure and Bullet Structure. Then, operate gamecheck function that can Judging the state of the game and making corresponding adjustments. If it is normal, the system starts to generate ship/AI-ship and its bullets according the rules. Next, it is to standard the motion of these objects. As for bullets that have been launched, using the bullethit function to judge whether the bullets has approach to the goals. If it does, operate the clearship or clearbullet function and revive function according to the game mode.

In the level system, there are eight levels, each of which is different in difficulty. Users can enter the next level after passing the level, and the game ends when the last level or the life value is used up. We also consider that players might want to pause the game sometimes, so we design the stop function to satisfy the needs of them.

1. **The generation and movement of ships and submarines**

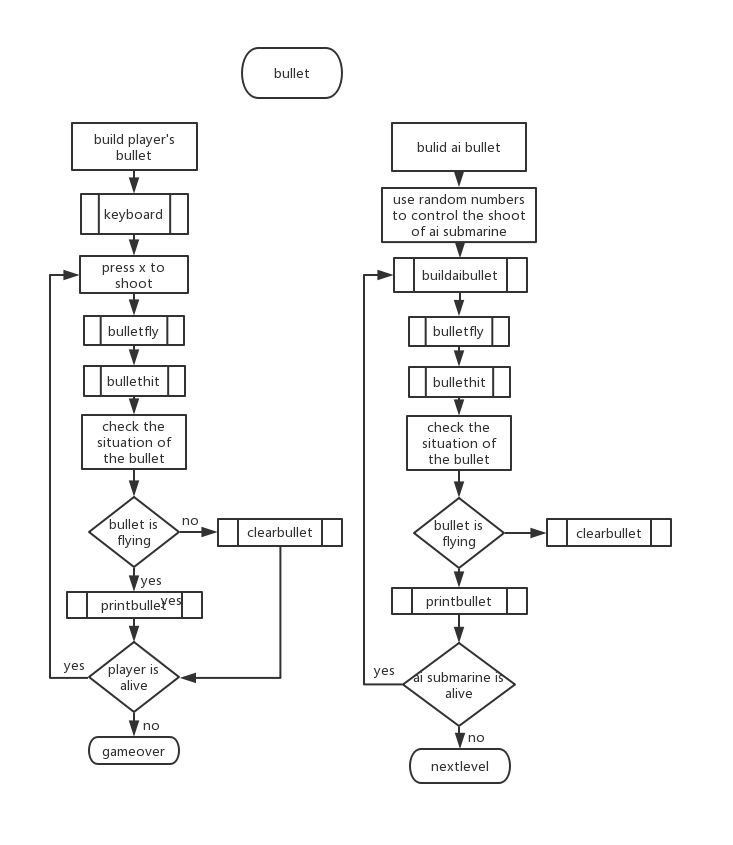


Picture 4.2 The design of the ship/submarine

To achieve the function of ship and submarine, the first thing is create the structure of them, then define some characteristic variables, such as direction, color, model and CD (cooling time). Next, we used char variable to design the model of many kinds ships and submarines. When they start generating, the system will them corresponding initial direction and position. At the same time, the players can control the motion of warship to move left or right by direction keys and fire bullets to shoot these submarines.

In addition, we achieve the function that these submarines can randomly fire bullets and change directions to strengthen them by using the random function and Periodic function. So, we create AI submarines successfully. When these objects are When these objects are hit by bullets, the corresponding clearance program is started and the current game status is displayed in time to ensure timely updating of the interface and better interaction with users.

1. **The launching and striking of Bullets**



Picture 4.3 the design of bullets and AI-bullets

As the picture shows, we divide bullets into two categories: mine and enemy bullets. Each bullet is for a thread, but in the thread, it shares several functions, such as bulletfly, bullethit, printbullet and clearbullet. The biggest difference is that they are generated in different ways. The former is controlled by players, and the latter is generated randomly by systems.

It is noticeable that there are reading, writing and storage of the position of the warship or submarine in the launching function of the bullet, and we adopt the CD method (cooling time) to ensure that the interface is updated in time.

1. **Adopting the coordinate classification method**

In order to reduce the complexity of computer operation, we adopt two kinds of coordinates, i.e. inner coordinates and outer coordinates.

The inner part is composed of an int map array of 41\*41. Each element represents the case in which the row and column subscripts of the array elements are the map coordinates y and X. Different map obstacles have different values in the array. Tanks occupy 3\*3 units on the map and 3\*3 elements of the corresponding coordinates in the map array correspond to special values. 。 The tank information can be read from the map array values.

The table part prints the corresponding characters on the command line interface through gotoxy and printf functions according to the map array of the inner part to form the game interface.

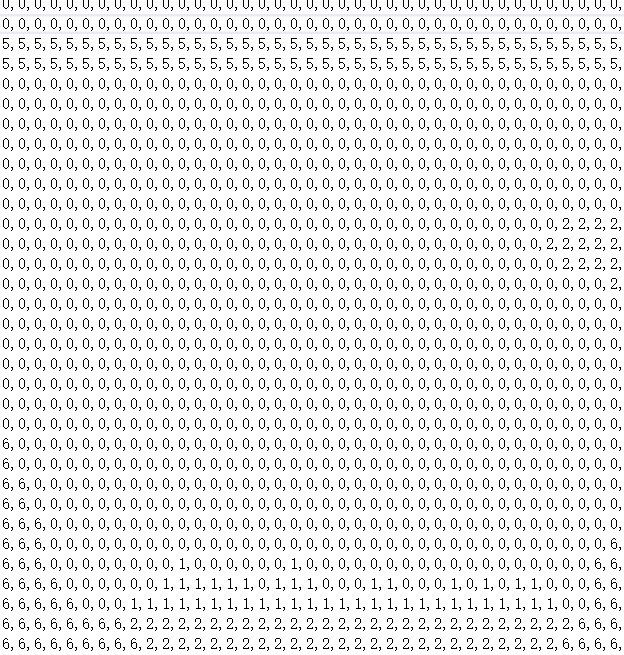
Each function operation in the program is judged by the inner part (map array), then the inner part (map array) is operated, and then the outer part is changed by the inner part. The visual interface is presented to the player by gotoxy and printf functions.

That is to say, the inner and outer parts of the mAIn body function of the game are operated together and not separated.

1. **IMPLEMENTION**

In this part, we will show some important parts of our code and explain them.

1. **Inner and outer table: realizing the motion of objects and the visualization of of Game Operating Interface**

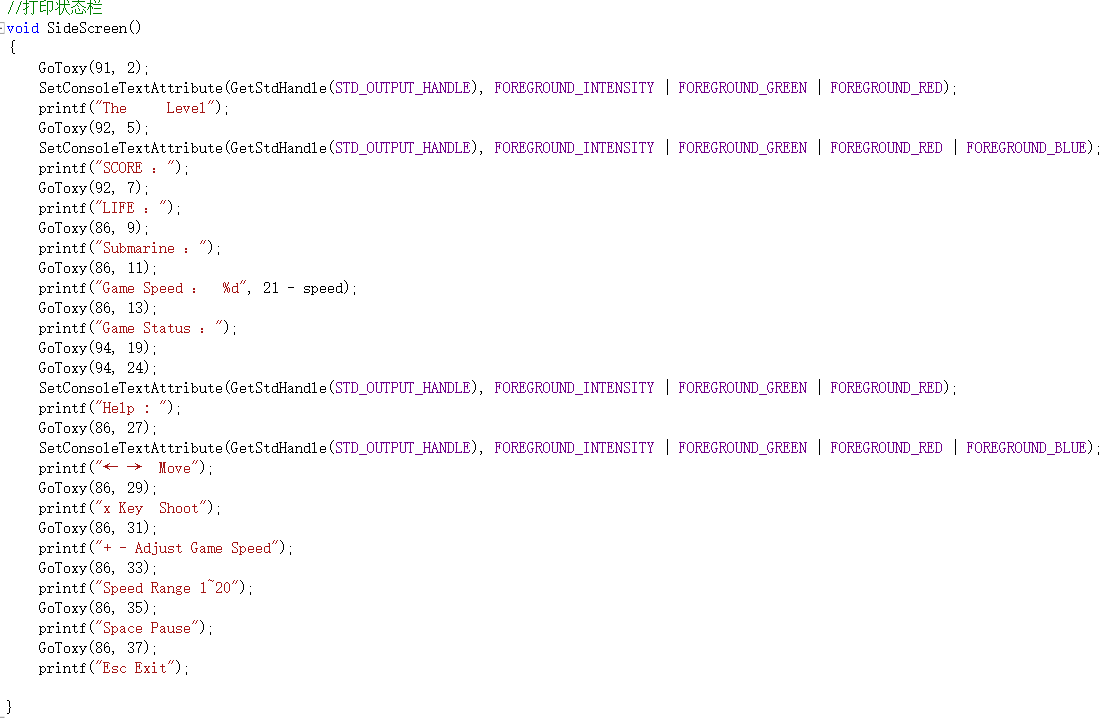


We use the two-dimensional array to make our maps. One number connects to a square which means one little space in our game. In the first map that I show you in the word, you can see that the number 0 means the place that the ship and submarines can move through. The number 5 means the water face, which makes an interval of the ship and the submarines. The number 1, 2 and 6 mean some special square like sand, oasis and iron. With this way to make our map, we can simply define the actions of the ship and submarines. The two-dimensional array connects to the coordinate. We can give the ship and submarines the x, y coordinates. For example, the water face has the y coordinate of 10. As a result, we can move the ship and submarines by changing their coordinates. Moving to right equals their x coordinate plus a number. The same as bullet, the bullet can also move with this way.

It makes it easy for us to make our project.

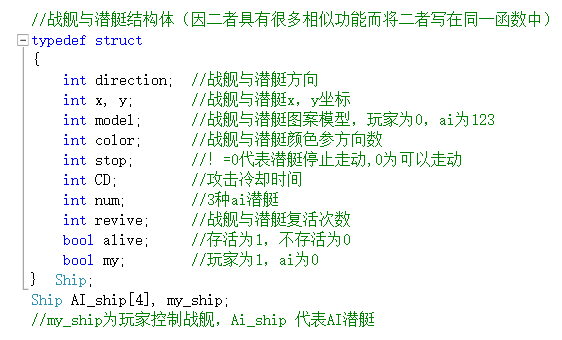
The array way of how to make the map is my favorite. It helps a lot in other works. It is interesting to design our map by give different places different numbers. We make many different maps by this way and make the game more beautiful.

1. **Real-time display of current game status on screen**



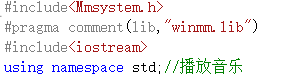
Besides the map, the game also needs a chart to inform the game situation. We use “printf” to print some different kinds of situations on the screen. We divide the place in two parts, one up and one down. In the up part, we list the game situation to make sure that players know their score, their life and their remaining enemies. This part prints the data of the information I mention. In the down part, we introduce the basic control of the game. We do not use another screen because our ability and time are limited.

1. **Creating a large number of structures to achieve functional partitioning**



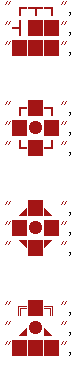
To make our ship and submarines, we use the structure. With this way, we can define many data and situations of the ship and submarines together. Using one structure of both the ship and submarines can also help us save our time and give us more time to finish our project because they have many things that have the same function and they have the same way to move or shoot. The differences are that the submarines use the AI to control their movement and shoot.

1. **Setting background music using Playsound function.**





1. **Making game objects with Two-Dimensional array**



We design the ship and submarines by using some two-dimensional pictures. To make the game easy and pithy, we did not use other program or use some 3d model. To make the ship and submarines fit the map, we did not use the real pictures of the ship and submarines.

The first picture is our design of the ship. We did not make itsymmetrical because it will make our players have no sense of the front and the back of the ship. Besides, the dissymmetrical model also has it beauty. In the bottom, we just print 3 squares, because the model is not big. And using square seems that the ship is stable. In the middle, we use these pictures to make it like a war ship. On the top, we draw a helmet which looks like a protection. When we design submarines, we use symmetrical design. We did not make them like real submarines because with these kinds of designs submarines will have a unique model which will surprise our players. Specially, in the third design, we make the submarine more like a big gun, which means it is more dangerous than the first and the second.

1. **Counter + Cyclic Judgment: Implementing dead loop body of program main function**

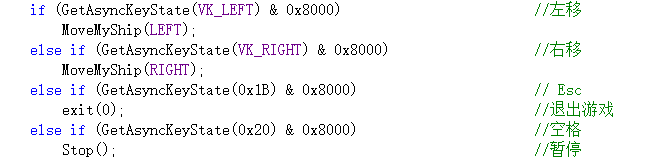


It is about the execution mode of the dead loop body of the main function of the program.Originally, the cooling time of submarine bullet launching and the bullet movement were all completed by sub-threads. After learning how to add counters in the dead cycle of the main thread to realize the multi-threading function, multi-threading was abandoned in a way similar to the multi-threading implementation of single-core cpu, that is, the single-threaded program ran back and forth between different operations to achieve the effect of multi-threading. This can also be more convenient to achieve the game built-in function of adjusting the speed of the game.

Because the bullet has been the smallest unit of time, the program is executed in every cycle, so there is no need to set a counter. It takes two cycles to execute one submarine movement, which reflects the speed difference and eliminates multi-threading.

1. **Using getasynckeystate instead of getch.**



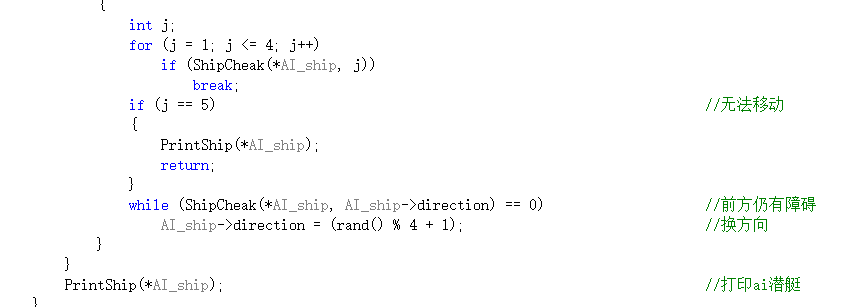




We first used getch to use the keyboard, but then we found a better way to use getasynckeystate.

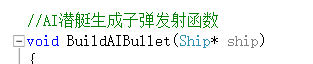
1. **Rand function: realize the intelligent submarine and increase the fun of the game.**

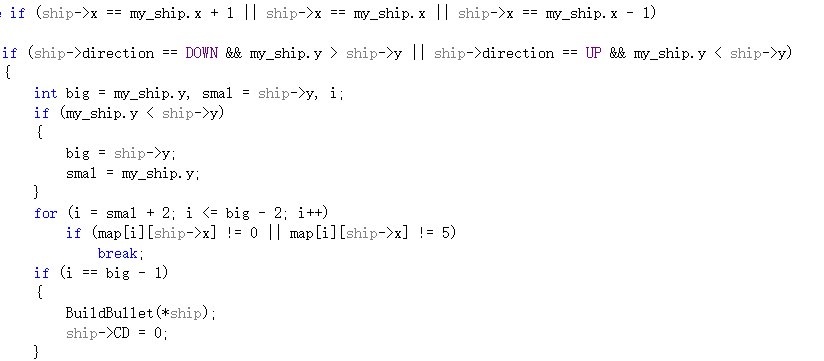


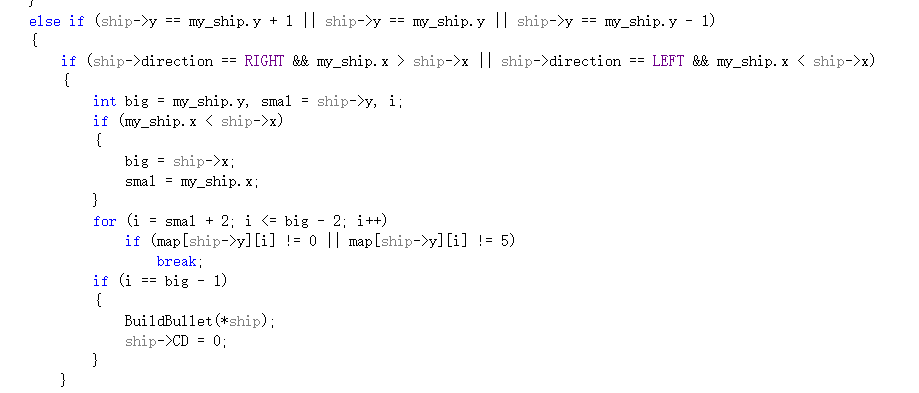


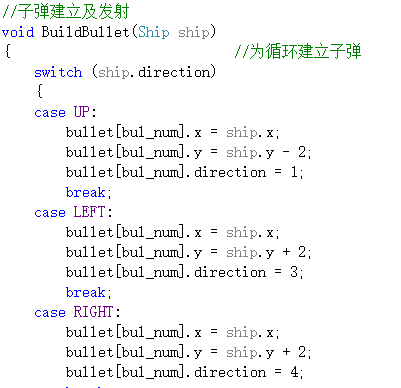
We use random numbers to control the move of AI submarines. We first build a random number and judge its value. After accepting its value, we give submarines their movement according to the value of the random numbers. If it can divide one number without remAIning number, it can do its action. It will also check their position to let it move to a right place.

1. **Structural Information Call**









The bullet launching program needs to initialize the bullet, in which the position information needs to temporarily read and store the position information of the submarine or warship.

In this part, we make our bullet. For AI, we also use random numbers to control its shoot. For player, we press x to shoot. The movement of bullet can be divided into 4 parts. The first part is giving it a coordinate of both x and y and print it iin the map of two-dimensional array. The second part is clearing the bullet by printing the black square in the place of the bullet. The third part is giving it a new coordinate. If it moves down, the y coordinate will plus one. If it moves up, the y coordinate will minus one. When it hit something, it will be destroyed by clearing it. Then judge what it hit. If it hit the ship or submarines, their life will minus one. If they hit the frame or other square, nothing happens but the bullet disappears.

**6.TESTING**

We test and debug the program in blocks according to the function that we mentioned before. In the block test section, we test the following three parts, player part, AI part, gamecheck part.After realizing the expected functions of each small block successfully, we conduct a system test, and comprehensively check the stability and coordination of the system operation.

In the player part, we check the keyboard that the player will use when he is playing the game.

In the AI part, we test whether the AI submarines and bullets can work successfully.

In the game check part, wetest the win and lose of the game which includes the game finish part and if it will successfully to play next level if player wins one level.

1. **Wetest the action of player controlling part.**

|  |  |  |
| --- | --- | --- |
| Input | Expected result | Real result |
| Enter | Game start | Game start |
| Left | Ship.x=x-1 | Ship.x=x-1 |
| Right | Ship.x=x+1 | Ship.x=x+1 |
| X | Bullet is built and it moves down. | Bullet is built and it moves down. |
| Space | Game stop | Game stop |
| Esc | Exit the game | Exit the game |

Table 6.1 The test of Player part

As you can see, each keyboard button corresponds to a game function. In the test, we tested the function of keys for each game several times, as we expected. It's worth noting that even in fast-paced mode, our design works very well to prevent button sticking.

1. **Wetest allAI functions.**

|  |  |  |
| --- | --- | --- |
| Input | Expected result | Real result |
| Submarine1 | Submarine move and shoot | Submarine move and shoot |
| Submarine2 | Submarinemove and shoot | Submarine move and shoot |
| Submarine3 | Submarine move and shoot | Submarine move and shoot |
| Submarine is hit | It is destroyed | It is destroyed |
| Submarine move to the frame | It change its direction | It change its direction |

Table6.2 The test of AI functions

1. **S : All of the above are to test the randomly generated parts of the system.**

As the table shows, all the default AI functions are functioning properly. However, this is the result of the operation of a single structure, and the final effect can only be eliminated by system debugging.

1. **test the game over part.**

|  |  |  |
| --- | --- | --- |
| Input | Expected result | Real result |
| Ship is destroyed | Gameover | Gameover |
| Submarines are all destroyed | Nextlevel | Nextlevel |
| Pass all level | Print you win | Print you win |

Table6.3 The test of the gamecheck

This chart shows the test of gamecheck. Because this part is a little complex, we only test the important part of it. For example, in this chart, we can see that the real result is the same as the expected result. This function controls the whole game process, and the system chooses the direction of threads according to many judging factors.

By now, we have tested the player part, AI part, game check part and the level functions. Just as the results showed, these parts work as well as we expected.

**7.CONCLUSION**

Compared with the original version, the game is run by PC written by C/C++, and the game screen is realized by character interface. As for funtion, the game refers to the game Subhunter on FC/Family Computer in the 1980s, including maps, game modes and so on (the game was burned directly on the circuit board at that time). So the balance of the game has a good reference, no need to spend a lot of time testing balance. Considering that PC performance is different, game speed adjustment is built in. Players can adjust to the appropriate speed according to PC performance.

We have already finished almost all our expectation. Although our game still has some places to improve, now it is playable enough. After testing our game, we make sure that our game can work successfully. Our team worked together for three month, and we all devote ourselves to work. We are glad to finish the project and the c class this term.