**SEA FIGHT**

**v. 1.0.**

**done by Oksana Bartash**

**(group B-22 red, Vinnytsia IT-Academy)**

**Task obtained: 16.01.2023**

**Pre-defence: 17.02.2023, Teacher1\_\_\_\_\_\_\_\_\_**

**Defence: 11.02.2023, Teacher2\_\_\_\_\_\_\_\_\_\_\_**

**CONTENTS**

**1. About game "SEA FIGHT"**

**2. User guide**

**3. Programmer instruction**

**4. About the author**

**1. About game "SEA FIGHT"**

***"Sea Fight"*** is a game for one participant, in which, listening to his intuition, he enters the coordinates on the map where the computer has randomly placed ships.

If there is a ship in that place (coordinates are occupied), then the ship "Killed" or its part is "Injured". The player's goal is to sink all the ships.

***Rules for placement of ships (fleet)***

The playing field is a 10 × 10 square on which a fleet of ships is placed.

The horizontal is indicated by numbers from left to right. In the same way, the vertical is indicated by numbers in the direction from top to bottom.

There are nine ships:

1 ship — a row of 4 cells (or "four-deck")

2 ships — a row of 3 cells (or "three-deck")

3 ships — a row of 2 cells (or "double deck")

3 ships — 1 cell (or "single deck")

Ships must be placed in accordance with the rules: they can't touch each other's corners and sides. Ships can’t be square shaped ("four-deck") or the letter "L" ("three-" and "four-deck").

***Search and sinking of ships***

Before starting hostilities, the player must read the rules of the game.

Information about the existing fleet is also displayed on the screen - that is, how many and which ships are placed on the map.

The player takes a shot by entering the coordinates of the cell (from 0 to 9) from the computer keyboard, in which, in his opinion, the ship that placed the computer is located, for example: "2" Enter "3".

If the shot hits a cell that is not occupied by any of the ships, the computer informs the player with the inscription "You missed" and "—" ("dash") appears on the field.

If the shot hits the cell where the multi-deck ship is located (larger than 1 cell), the player receives the response "Injured!" and appears - "X" ("cross") on the field.

If the shot hits the cell where the single-deck ship is located, or the last "undamaged" cell of the multi-deck ship, the player receives the answer "Killed!", and a "cross" - "X" also appears on the field.

If there won't be violations and when the last ship would sink, the player would receive a victory message!

**2. User guide**

1. Find and run the file Sea Fight.exe in the folder Sea Fight.

2. Read the information about the game and its rules.

3. Familiarize yourself with the "Sea Fleet" - the number and size of ships.

4. The playing field is below, it is masked ships which aren’t visible to the player.

Taking into account the numbering of the field (horizontal and vertical numbers), start playing.

5. Enter the coordinates of the shot "x" and "y" in the range from 0 to 9.

6. Depending on the shot, an inscription appears on the screen:

- "You missed" is marked with “—” ("dash") on the field;

- "Injured!" a "X" ("cross") is marked on the field;

- "Killed!" a "X" ("cross") is marked on the field.

7. If you hit a ship, analyze the situation and decide on the next move, taking into account the rules for the location of ships (see above).

8. Repeat the steps until all ships are drowned.

9. At the end of the game, the program will automatically notify you of the winning.

10. To exit the program, click "Exit" - X in the left-right corner of the window.

**3. Programmer instructions**

The project was written in C++.

The project is a single block that contains all the necessary algorithms and functions.

Find and open the Sea Fight.exe file in the Sea Fight folder. You can use C++ compilers such as Dev C++ to edit, debug, and compile your program code.

***Basic global variables, arrays and functions:***

N — is the size of our matrix (map) 10\*10 - it is square;

size\_ship — the number of decks on the ship;

num\_ships — the number of ships;

x, y — coordinates of the 1st deck of the ship;

temp\_x and temp\_y are temporary constants for recording the initial coordinates in them, because later when checking "x" and "y" will change;

dir (direction) — an indicator of the direction where the ship will be placed (the direction is generated on all 4 sides);

count\_ship — counter of the number of ships;

count\_tact — counter to avoid looping when placing ships on the map (if after 1000 attempts the ships don't appear, the cycle will be interrupted);

setting\_is\_possible — checking the possibility of placing ships;

ships\_id — ship numbering;

counter — counter of sunken ships;

map[N][N] — is an array "map" of ship locations - "Fleet Map";

mask[N][N] — is an array to display the shots and also it masks the previous array "map" to doesn’t see the targets;

ships[ships\_id] — the array numbers all ships regardless of size and all their decks with one number;

ships[map[x][y]] — is an array of ships with decks for displaying wounds (reduced when hit) or checking the number of "lives=decks";

mask[x][y] — is an array for placing marks in the cells when we hit;

void set\_rand\_ships() — a function for placing ships - this function generates coordinates for placing a ship, determines the direction of its placement, and also checks whether it is within the map and its appearance;

void map\_show() — "Draw" function - this function displays a map on the screen, numbers it, draws ships and then masks them from the player, depending on the shot displays a mark on the fleet map;

int main() — shows information about the program, displays instructions for users, fixes the initial position of the playing field, the player shoots at the ships, and the result of each shot is displayed.

**4. About the author**

This project was carried out by Oksana Bartash, a student of the Vinnytsia IT Academy (group B22 red, January 2023).