

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ
“БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ”
ИНТЕЛЛЕКТУАЛЬНЫЕ ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

ОТЧЁТ

по лабораторной работе №6

“Разработка консольного приложения в Windows.”

Выполнил:
студент группы ПО-9
Сольшко Д.А.

Проверил:
Козик И.Д.

Брест 2023

Лабораторная работа №6

Вариант №19

Цель работы: отработать навыки по созданию консольных приложений в Windows, используя C++.

Задание: 4. Создать консольный морской бой. Программа должна выводить в консоль игровые поля (корабли на поле оппонента должны быть скрыты) и иметь поддержку игры с компьютером (заполняет поле и делает ходы случайным образом, например).

Текст программы:

```
#include <iostream>
using namespace std;

const int SIZE_OF_BOARD = 10;
const int COUNT_OF_ONE_POINT_SHIPS = 4;
const int COUNT_OF_TWO_POINT_SHIPS = 3;
const int COUNT_OF_THREE_POINT_SHIPS = 2;
const int COUNT_OF_FOUR_POINT_SHIPS = 1;

class Board
{
public:
    char board[SIZE_OF_BOARD][SIZE_OF_BOARD];

    Board()
    {
        for (int i = 0; i < SIZE_OF_BOARD; i++)
        {
            for (int j = 0; j < SIZE_OF_BOARD; j++)
            {
                this->board[i][j] = ' ';
            }
        }
    }

    bool CheckAdjacent(int xpos, int ypos)
    {
        for (int i = xpos - 1; i <= xpos + 1; ++i)
        {
            for (int j = ypos - 1; j <= ypos + 1; ++j)
            {
                if (i >= 0 && i < SIZE_OF_BOARD && j >= 0 && j < SIZE_OF_BOARD && this->board[i][j] ==
'S')
                {
                    cout << "Вблизи уже есть корабль, попробуйте снова" << endl;
                    return false;
                }
            }
        }
        return true;
    }

    bool InitializeOnePointShip(int xpos, int ypos)
    {
        if (xpos >= SIZE_OF_BOARD || ypos >= SIZE_OF_BOARD || ypos < 0 || xpos < 0)
        {
            cout << "Ваш корабль за границей, попробуйте снова" << endl;
        }
    }
};
```

```

        return false;
    }

    if (!CheckAdjacent(xpos, ypos))
    {
        cout << "Ваш корабль Находится на позиции другого корабля, попробуйте снова" << endl;
        return false;
    }

    this->board[xpos][ypos] = 'S';
    return true;
}

bool InitializeTwoPointShip(int xpos, int ypos, char direction)
{
    if (direction == 'H')
    {
        if (xpos >= SIZE_OF_BOARD || ypos + 1 >= SIZE_OF_BOARD || ypos < 0 || xpos < 0)
        {
            cout << "Ваш корабль за границей, попробуйте снова" << endl;
            return false;
        }

        if (!CheckAdjacent(xpos, ypos) || !CheckAdjacent(xpos + 1, ypos))
        {
            cout << "Ваш корабль Находится на позиции другого корабля, попробуйте снова" <<
endl;

            return false;
        }

        this->board[xpos][ypos] = 'S';
        this->board[xpos][ypos + 1] = 'S';
    }
    else if (direction == 'V')
    {
        if (xpos + 1 >= SIZE_OF_BOARD || ypos >= SIZE_OF_BOARD || ypos < 0 || xpos < 0)
        {
            cout << "Ваш корабль за границей, попробуйте снова" << endl;
            return false;
        }

        if (!CheckAdjacent(xpos, ypos) || !CheckAdjacent(xpos + 1, ypos))
        {
            cout << "Ваш корабль Находится на позиции другого корабля, попробуйте снова" <<
endl;

            return false;
        }

        this->board[xpos][ypos] = 'S';
        this->board[xpos + 1][ypos] = 'S';
    }
    else
    {
        cout << "Неправильно обозначено направление" << endl;
        return false;
    }
    return true;
}

bool InitializeThreePointShip(int xpos, int ypos, char direction)
{

```

```

        if (direction == 'H')
        {
            if (xpos >= SIZE_OF_BOARD || ypos + 2 >= SIZE_OF_BOARD || ypos < 0 || xpos < 0)
            {
                cout << "Ваш корабль за границей, попробуйте снова" << endl;
                return false;
            }

            if (!CheckAdjacent(xpos, ypos) || !CheckAdjacent(xpos + 1, ypos) || !CheckAdjacent(xpos + 2,
ypos))
            {
                cout << "Ваш корабль Находится на позиции другого корабля, попробуйте снова" <<
endl;

                return false;
            }

            this->board[xpos][ypos] = 'S';
            this->board[xpos][ypos + 1] = 'S';
            this->board[xpos][ypos + 2] = 'S';
        }
        else if (direction == 'V')
        {
            if (xpos + 2 >= SIZE_OF_BOARD || ypos >= SIZE_OF_BOARD || ypos < 0 || xpos < 0)
            {
                cout << "Ваш корабль за границей, попробуйте снова" << endl;
                return false;
            }

            if (!CheckAdjacent(xpos, ypos) || !CheckAdjacent(xpos + 1, ypos) || !CheckAdjacent(xpos + 2,
ypos))
            {
                cout << "Ваш корабль Находится на позиции другого корабля, попробуйте снова" <<
endl;

                return false;
            }

            this->board[xpos][ypos] = 'S';
            this->board[xpos + 1][ypos] = 'S';
            this->board[xpos + 2][ypos] = 'S';
        }
        else
        {
            cout << "Неправильно обозначено направление" << endl;
            return false;
        }
        return true;
    }

    bool InitializeFourPointShip(int xpos, int ypos, char direction)
    {
        if (direction == 'H')
        {
            if (xpos >= SIZE_OF_BOARD || ypos + 3 >= SIZE_OF_BOARD || ypos < 0 || xpos < 0)
            {
                cout << "Ваш корабль за границей, попробуйте снова" << endl;
                return false;
            }

            if (!CheckAdjacent(xpos, ypos) || !CheckAdjacent(xpos + 1, ypos) || !CheckAdjacent(xpos + 2,
ypos) || !CheckAdjacent(xpos + 3, ypos))
            {

```

```

        cout << "Ваш корабль Находится на позиции другого корабля, попробуйте снова" <<
endl;

        return false;
    }

    this->board[xpos][ypos] = 'S';
    this->board[xpos][ypos + 1] = 'S';
    this->board[xpos][ypos + 2] = 'S';
    this->board[xpos][ypos + 3] = 'S';
}
else if (direction == 'V')
{
    if (xpos + 2 >= SIZE_OF_BOARD || ypos >= SIZE_OF_BOARD || ypos < 0 || xpos < 0)
    {
        cout << "Ваш корабль за границей, попробуйте снова" << endl;
        return false;
    }

    if (!CheckAdjacent(xpos, ypos) || !CheckAdjacent(xpos + 1, ypos) || !CheckAdjacent(xpos + 2,
ypos) || !CheckAdjacent(xpos + 3, ypos))
    {
        cout << "Ваш корабль Находится на позиции другого корабля, попробуйте снова" <<
endl;

        return false;
    }

    this->board[xpos][ypos] = 'S';
    this->board[xpos + 1][ypos] = 'S';
    this->board[xpos + 2][ypos] = 'S';
    this->board[xpos + 3][ypos] = 'S';
}
else
{
    cout << "Неправильно обозначено направление" << endl;
    return false;
}
return true;
}

bool Attack_On_Ship(int xpos, int ypos)
{
    if (xpos >= SIZE_OF_BOARD || ypos >= SIZE_OF_BOARD || ypos < 0 || xpos < 0)
    {
        cout << "Ваш корабль за границей, попробуйте снова" << endl;
        return true;
    }

    if (this->board[xpos][ypos] == 'S')
    {
        this->board[xpos][ypos] = 'X';
        cout << "Поподание! Ваш ход продолжается." << endl;
        return true;
    }
    else if (this->board[xpos][ypos] == ' ')
    {
        this->board[xpos][ypos] = '#';
        cout << "Промех" << endl;
        return false;
    }
    else if (this->board[xpos][ypos] == '#' || this->board[xpos][ypos] == 'X')
    {

```

```

        this->board[xpos][ypos] = '#';
        cout << "Вы уже стреляли в данные координаты попробуйте снова" << endl;
        return true;
    }
    else
    {
        cout << "Ошибка в коде" << endl;
        return -1;
    }
}

void ShowBoard()
{
    cout << " 0123456789" << endl;
    for (int i = 0; i < SIZE_OF_BOARD; i++)
    {
        cout << i;
        for (int j = 0; j < SIZE_OF_BOARD; j++)
        {
            cout << this->board[i][j];
        }
        cout << endl;
    }
}

void ShowEnemyBoard()
{
    cout << " 0123456789" << endl;
    for (int i = 0; i < SIZE_OF_BOARD; i++)
    {
        cout << i;
        for (int j = 0; j < SIZE_OF_BOARD; j++)
        {
            if (this->board[i][j] == 'S')
            {
                cout << " ";
            }
            else
            {
                cout << this->board[i][j];
            }
        }
        cout << endl;
    }
}

bool GamelsOver()
{
    for (int i = 0; i < SIZE_OF_BOARD; i++)
    {
        for (int j = 0; j < SIZE_OF_BOARD; j++)
        {
            if (this->board[i][j] == 'S')
            {
                return false;
            }
        }
    }
    return true;
}

};

```

```

int main()
{
    setlocale(LC_ALL, "rus");
    Board playerBoard;
    Board computerBoard;
    bool try_again_player = true, try_again_computer = true;
    int xpos = 0, ypos = 0;
    char direction = ' ';
    for (int i = 0; i < COUNT_OF_ONE_POINT_SHIPS; i++)
    {
        playerBoard.ShowBoard();
        do
        {
            cout << "Введите координаты корабля-единички: ";
            cin >> xpos >> ypos;
            try_again_player = playerBoard.InitializeOnePointShip(xpos, ypos);
        } while (!try_again_player);

        do
        {
            xpos = rand() % (SIZE_OF_BOARD);
            ypos = rand() % (SIZE_OF_BOARD);
            try_again_computer = computerBoard.InitializeOnePointShip(xpos, ypos);
        } while (!try_again_computer);
    }

    for (int i = 0; i < COUNT_OF_TWO_POINT_SHIPS; i++)
    {
        playerBoard.ShowBoard();
        do
        {
            cout << "Введите координаты корабля-двочки: ";
            cin >> xpos >> ypos;
            cout << "Введите направление корабля-двочки (H - Horizontal; V - Vertical): ";
            cin >> direction;
            try_again_player = playerBoard.InitializeTwoPointShip(xpos, ypos, direction);
        } while (!try_again_player);

        do
        {
            xpos = rand() % (SIZE_OF_BOARD);
            ypos = rand() % (SIZE_OF_BOARD);
            direction = (rand() % 2 == 1) ? 'H' : 'V';
            try_again_computer = computerBoard.InitializeTwoPointShip(xpos, ypos, direction);
        } while (!try_again_computer);
    }

    for (int i = 0; i < COUNT_OF_THREE_POINT_SHIPS; i++)
    {
        playerBoard.ShowBoard();
        do
        {
            cout << "Введите координаты корабля-троечки: ";
            cin >> xpos >> ypos;
            cout << "Введите направление корабля-троечки (H - Horizontal; V - Vertical): ";
            cin >> direction;
            try_again_player = playerBoard.InitializeThreePointShip(xpos, ypos, direction);
        } while (!try_again_player);

        do
        {

```

```

        xpos = rand() % (SIZE_OF_BOARD);
        ypos = rand() % (SIZE_OF_BOARD);
        direction = (rand() % 2 == 1) ? 'H' : 'V';
        try_again_computer = computerBoard.InitializeThreePointShip(xpos, ypos, direction);
    } while (!try_again_computer);
}

for (int i = 0; i < COUNT_OF_FOUR_POINT_SHIPS; i++)
{
    playerBoard.ShowBoard();
    do
    {
        cout << "Введите координаты корабля-четверочки: ";
        cin >> xpos >> ypos;
        cout << "Введите направление корабля-четверочки (H - Horizontal; V - Vertical): ";
        cin >> direction;
        try_again_player = playerBoard.InitializeFourPointShip(xpos, ypos, direction);
    } while (!try_again_player);

    do
    {
        xpos = rand() % (SIZE_OF_BOARD);
        ypos = rand() % (SIZE_OF_BOARD);
        direction = (rand() % 2 == 1) ? 'H' : 'V';
        try_again_computer = computerBoard.InitializeFourPointShip(xpos, ypos, direction);
    } while (!try_again_computer);
}

while (!computerBoard.GameIsOver() && !playerBoard.GameIsOver())
{
    playerBoard.ShowBoard();
    do
    {
        computerBoard.ShowEnemyBoard();
        cout << "Введите координаты выстрела: ";
        cin >> xpos >> ypos;
        try_again_player = computerBoard.Attack_On_Ship(xpos, ypos);
    } while (try_again_player && !computerBoard.GameIsOver());

    cout << "Ход компьютера: ";
    do
    {
        xpos = rand() % (SIZE_OF_BOARD);
        ypos = rand() % (SIZE_OF_BOARD);
        cout << "Компьютер стреляет по координатам " << xpos << " " << ypos;
        try_again_computer = playerBoard.Attack_On_Ship(xpos, ypos);
    } while (try_again_computer && !playerBoard.GameIsOver());
}

if (computerBoard.GameIsOver())
{
    cout << "Вы победили" << endl;
}
else
{
    cout << "Вы проиграли" << endl;
}

return 0;
}

```


Результаты программы:

Заполнение корабля-единички, на месте корабля появляется S (Ship), в это же время рандомно заполняется корабль компьютера:

```
Введите координаты корабля-единички: 0 0
0123456789
0S
1
2
3
4
5
6
7
8
9
```

При попытке поставить за границы нас оповестит программа

```
Введите координаты корабля-единички: -1 2
Ваш корабль за границей, попробуйте снова
Введите координаты корабля-единички: 0 11
Ваш корабль за границей, попробуйте снова
```

При попытке поставить в область корабля (областью считаем клетку где он находится, и все его соседние клетки)

```
Введите координаты корабля-единички: 0 1
Вблизи уже есть корабль, попробуйте снова
Ваш корабль Находится на позиции другого корабля, попробуйте снова
```

Установка кораблей вместимостью 2, 3 и 4 отличаются, тем что еще задаётся направление корабля Н (Horizontal) и V (Vertical)

```
Введите координаты корабля-двоечки: 2 5
Введите направление корабля-двоечки (Н - Horizontal; V - Vertical): V
```

После заполнения каждого корабля выводится поле с кораблями.

После заполнения всех кораблей (4 корабля-единички, 3 корабля-двоечки, 2 корабля-троечки, 1 корабль-четверочка)

```
0123456789
0S S S
1
2 S S S
3 S S S
4
5
6 SSS S
7 S
8 S
9SSSS
0123456789
0
1
2
3
4
5
6
7
8
9
Введите координаты выстрела: _
```

После выстрела следует выстрел компьютера, и так по очереди

```
Компьютер стреляет по координатам 4 8Промех
0123456789
0S S S
1
2 S S S
3 S S S
4 #
5
6 SSS S
7 S
8 S
9SSSS
0123456789
0 #
1
2
3
4
5
6
7
8
9
Введите координаты выстрела:
```

При попадании даётся еще одна попытка на выстрел

```

Введите координаты выстрела: 7 0
Поподание! Ваш ход продолжается.
0123456789
0 #
1
2
3
4
5
6
7X
8
9
Введите координаты выстрела:

```

В случае если корабли одного из оппонентов уничтожены, игра прерывается и объявляется победитель (в данном случае ход 9 4, поразил последний корабль компьютера и я выиграл)

```

0123456789
0
1 XX # #X#
2   XX #
3  X
4X#X
5##  XXX#
6X#####
7X#
8X#   XXX#
9X#
Введите координаты выстрела: 9 4
Поподание! Ваш ход продолжается.
Ход компьютера: Компьютер стреляет по координатам 4 2Промех
Вы победили

```

Вывод: я отработал навыки по созданию консольных приложений в Windows, используя C++.