МИНЕСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное учреждение высшего образования

«Воронежский государственный технический университет»

**Факультет** информационных технологий и компьютерной безопасности

**Кафедра** информационных систем и технологий цифровизации

Курсовой проект

По дисциплине: Основы программирования и алгоритмизации

Тема: «Программирование логических задач»

Выполнил студент: Колесников В. А.

Группа: ИСТ-201

Руководитель: доцент, к. т. н. Минакова О.В.

Допущен к защите « » \_\_\_\_\_\_\_\_\_\_\_\_ 2020 г.

Работа защищена « » \_\_\_\_\_\_\_\_\_\_\_\_ 2020 г.

С оценкой \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_

(подпись)

Воронеж 2020

Содержание

[Введение 3](#_Toc58263382)

[1. Постановка задачи 4](#_Toc58263383)

[2. Конструирование программы 6](#_Toc58263384)

[3. Тестирование программы 15](#_Toc58263385)

[Заключение 18](#_Toc58263386)

[Список литературы 19](#_Toc58263387)

[Приложения 20](#_Toc58263388)

# **Введение**

Реализация программы для решения логических задач помогает закрепить знания, полученные в процессе дисциплины «Основы программирования и алгоритмизации», приобрести практические навыки самостоятельной разработки программ, структурного программирования и на практике продемонстрировать знания языка программирования Си.

Целью данной работы является алгоритмизация и написание кода на языке си для решения логической задачи:

**«Крестики-нолики»** Клеточное поле размером m x n есть результат игры в крестики-нолики на «бесконечном» (неограниченном) поле. Проверить, является ли конфигурация пред выигрышной для одного из игроков, то есть нельзя ли за один ход достичь победы. Считается, что, например, «крестики» выиграли, если на поле найдется по горизонтали, вертикали или диагонали цепочка, состоящая из пяти крестиков.

Основные задачи, которые необходимо решить в рамках проектирования:

* Реализовать модуль программы для проверки пред выигрышной конфигурации
* Создать интерфейс программы обеспечивающий выбор вывода результатов в консоль или файл(реализованный в функции main)

## **1. Постановка задачи**

Для решения поставленной задачи необходимо реализовать следующие алгоритмы:

1. Ввод размеров игрового поля
2. Ввод данных в двумерный символьный массив с отрисовкой игрового поля
3. Выяснение чей ход в данный момент
4. Проверка комбинаций

* По диагонали
* По горизонтали
* По вертикали

1. Вывод результатов проверки

Интерфейс всей программы будет реализовываться в виде меню, в форме диалога с пользователем (рисунок 1).

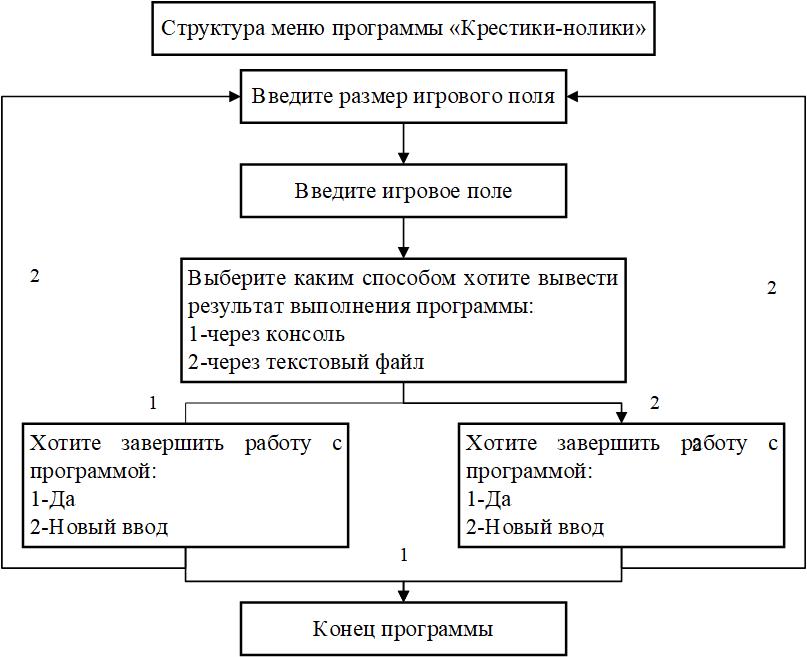


Рисунок 1 – Структурная схема диалога с пользователем

Для проверки работоспособности программы будут использоваться значения, представленные в таблице 1.

Таблица 1 – Контрольные примеры

|  |  |  |
| --- | --- | --- |
| *№* | *Входные данные* | *Результат* |
| *1* | 3X3  X O X  O \_ X  O \_ \_ | Следующий ход для X выигрышный |
| *2* | 3X3  X O X  O X O  O X \_ | Будет ничья |
| *3* | 3X3  X O O  X X \_  \_ \_ \_ | Через ход игрок X может выиграть |
| *4* | 3X3  \_ O \_  \_ X \_  \_ \_ \_ | Мало данных |
| *5* | 3X3  X X O  O X O  X O X | Все клетки уже заполнены |

## **2. Конструирование программы**

Прежде чем начинать анализ решения задачи и непосредственно приступать к ее выполнению, необходимо для большего понимания и удобства разбить задачу на подзадачи и для каждой расписать функции (рисунок 2).

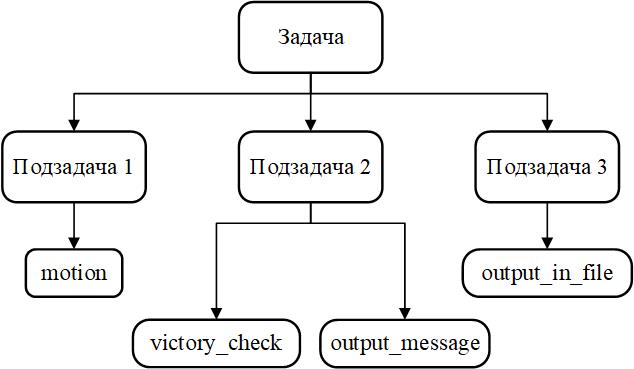


Рисунок 2 - Декомпозиция решаемой задачи

Подзадача 1. Необходимо реализовать функцию позволяющую определять чей ход происходит в данный момент.

Для этого:

* Предлагаем пользователю ввести размер игрового поля и само игровое поле.
* После создаём цикл, высчитывающий общее количество клеток на поле, а также количество пустых клеток на поле (если количество чётное ходят O, если нечётное, при нечётном количестве клеток на всём поле, при чётном наоборот).
* Добавляем вывод сообщения при полном заполненности поля.

Подзадача 2. Необходимо реализовать функции позволяющие определять выигрышную комбинацию, до которой один ход.

Для этого:

* Забираем значение символа игрока, который ходит из предыдущей функции.
* Далее реализуем три цикла поиска выигрышных комбинаций по вертикале, горизонтали и диагонали.
* После создаём функцию выбирающую какое сообщение выводить на экран или в консоль.

Подзадача 3. Необходимо реализовать вывод результата в выбранной пользователем форме.

Для этого:

* В главной функции создаём интерфейс для выбора формы вывода результатов проверки.
* Если пользователь выбрал вывод в консоль, то выводим сообщение, полученное из функции, описанной выше на экран.
* Для вывода в отдельный файл создаём функцию, в которую передаём сообщение, где открывается текстовый файл «game.txt», в который записывается сообщение с результатами.

Иерархия структуры программы показана на рисунке 3.

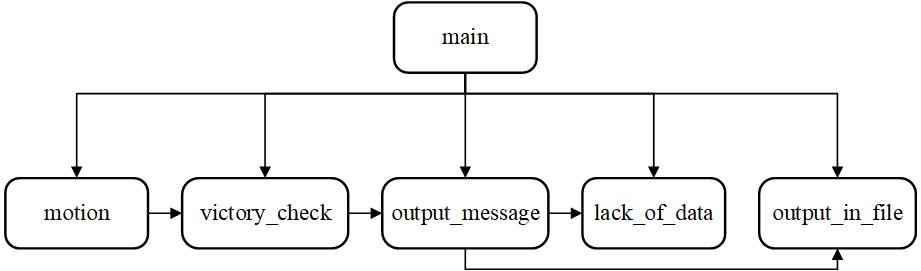


Рисунок 3 – Иерархия структуры программы

Блок – схема функции main () представлена на рисунке 4.

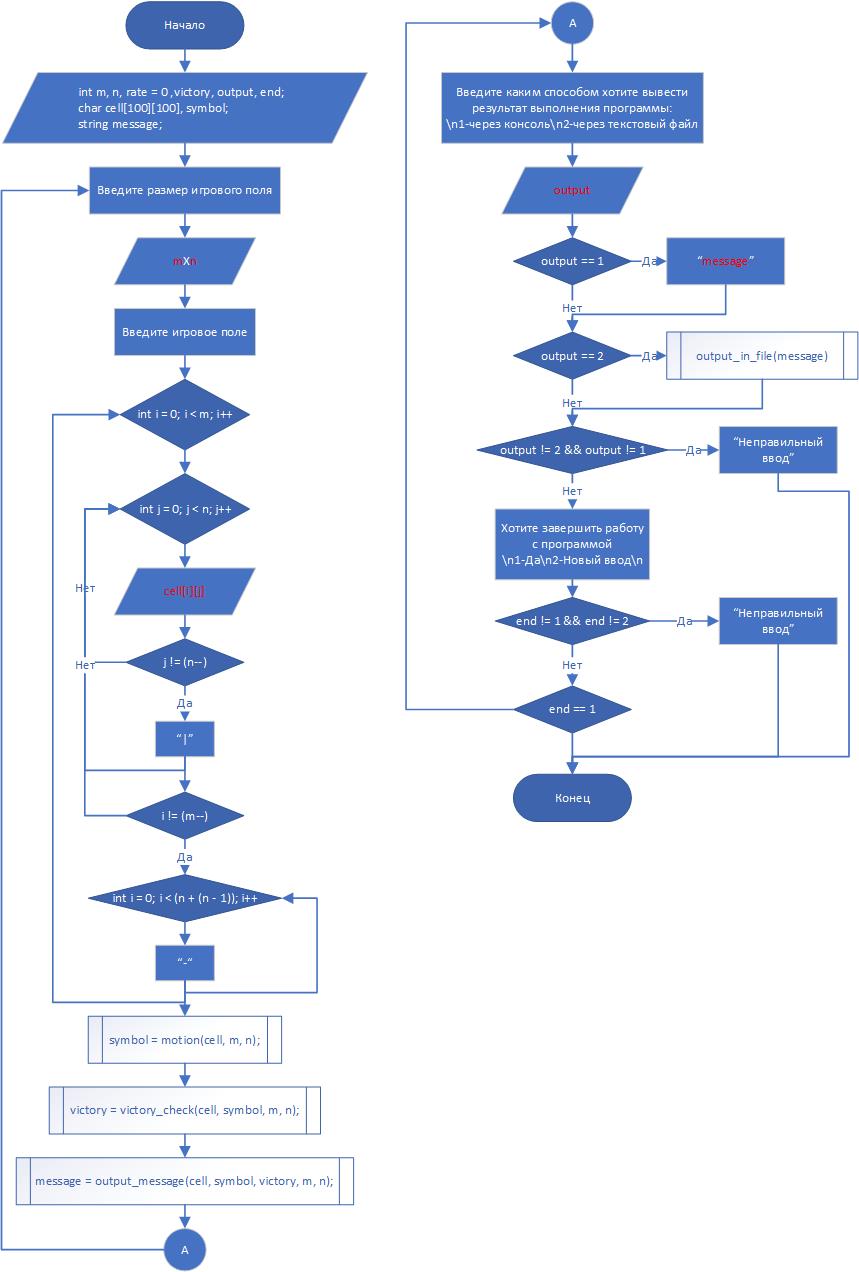


Рисунок 4 – Блок – схема главной функции

Описание собственных функций представлено в таблице 2.

Таблица 2 – Описание собственных функций

|  |  |  |
| --- | --- | --- |
| Описание собственных функция | | |
| Наименование | Назначение | Примечания |
| motion | Вычисление чей ход в данный момент | Возвращает символ игрока, делающего ход |
| victory\_check | Проверка комбинаций на выигрыш | Возвращает результат нахождения выигрышных комбинаций  (1-усть выигрышные комбинации, 0-нет) |
| output\_message | Вычисление какое сообщение выводить на экран | Возвращает сообщение, которое надо вывести на экран |
| lack\_of\_data | Проверка на заполненность поля | Выводит объявление о не заполненности поля, если оно является таковым |
| output\_in\_file | Вывод результата в файл | Печатает сообщение в текстовый файл «game.txt» |

Функция char motion(char cell[100][100], int m, int n) принимает значения игрового поля реализованного в двумерном массиве символов, а также размеры игрового поля введённого пользователем, блок-схема алгоритма представлена на рисунке 5.

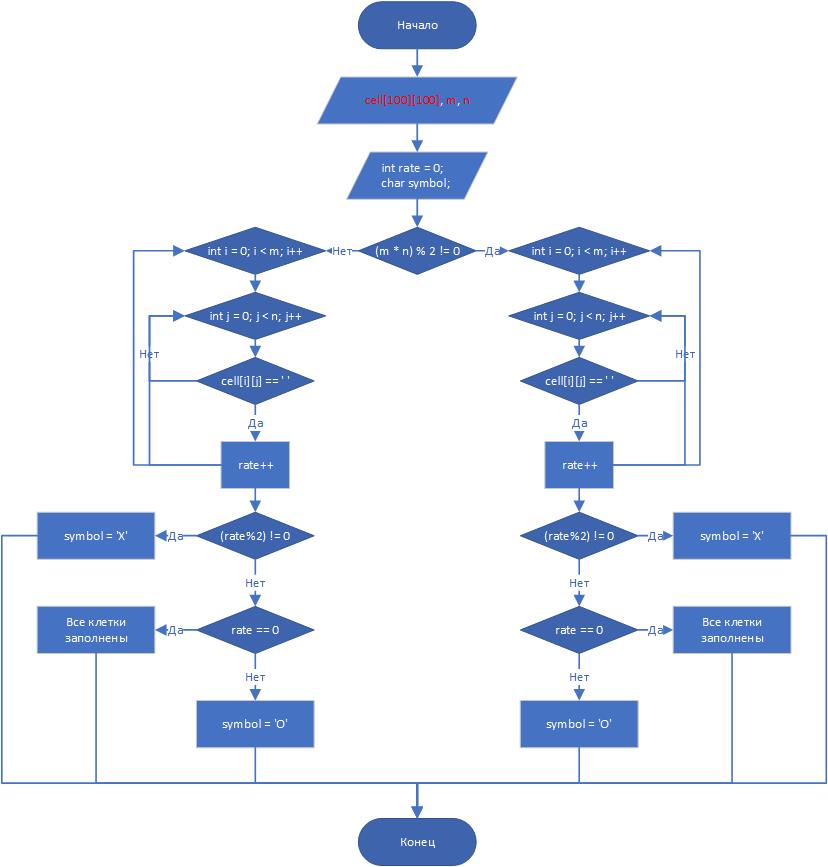


Рисунок 5 – Блок-схема вычисления какой игрок совершает ход

Функция int victory\_check (char cell[100][100], char symbol, int m, int n) принимает значения игрового поля, его размеры, а также символ игрока совершающего ход, блок-схема алгоритма представлена на рисунке 6.

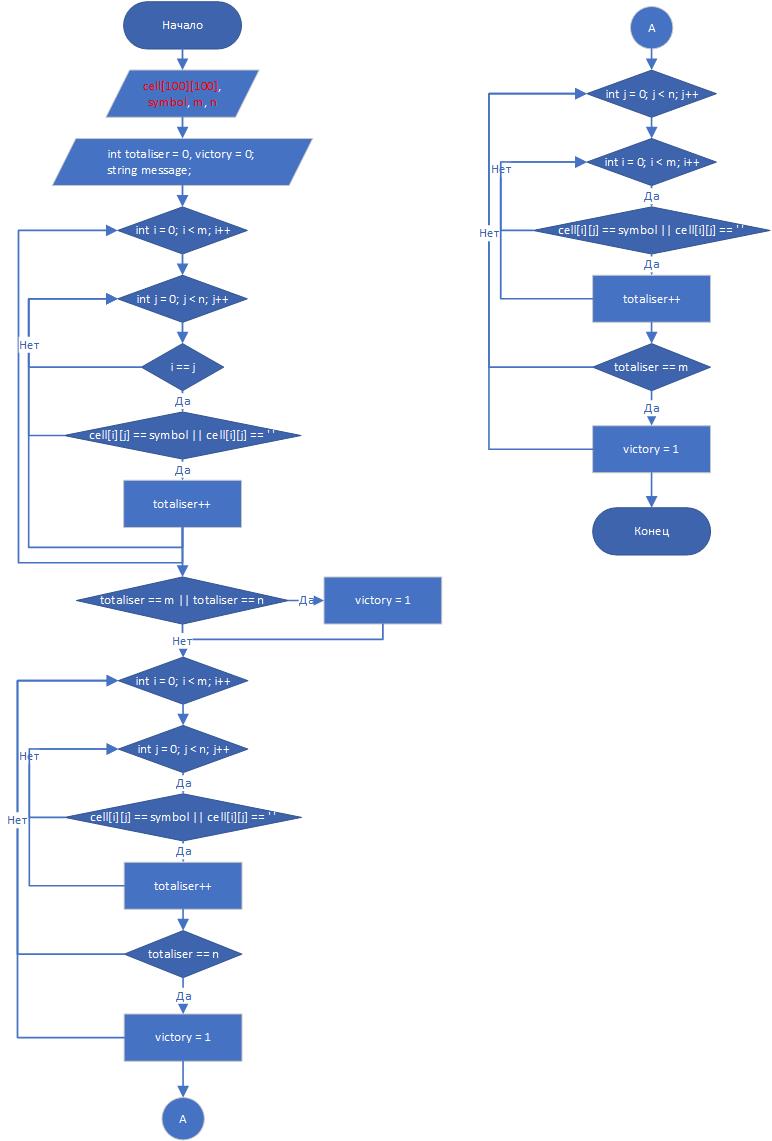


Рисунок 6 – блок-схема алгоритма для нахождения победных комбинаций

Функция string output\_message(char cell[100][100], char symbol, int victory, int m, int n) принимает размеры игрового поля, а также само игровое поле, также символ игрока делающего ход, и результаты проверки на победные комбинации, блок-схема алгоритма представлена на рисунке 7

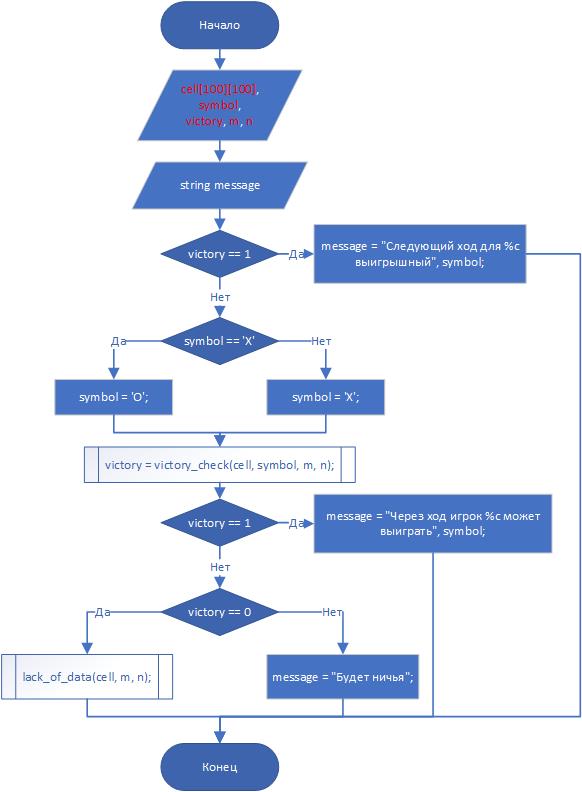


Рисунок 7 – блок-схема алгоритма по выбору выводимого сообщения

Для проверки поля на заполненность создаём функцию void lack\_of\_data(char cell[100][100], int m, int n) принимающая те же значения что и функция motion, блок-схема функции представлена на рисунке 8.

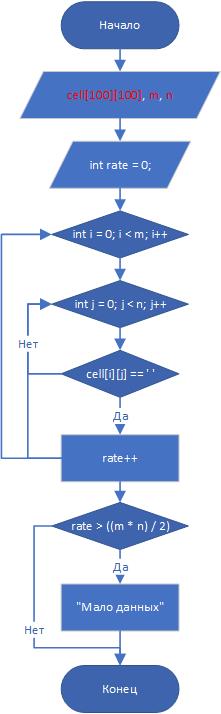


Рисунок 8 – блок-схема алгоритма проверяющего заполненость поля

Для вывода сообщений в отдельный файл реализуем функцию void output\_in\_file(string message) принимающая сообщение, которое в последствии записывается в файл, блок-схема алгоритма представлена на рисунке 9.

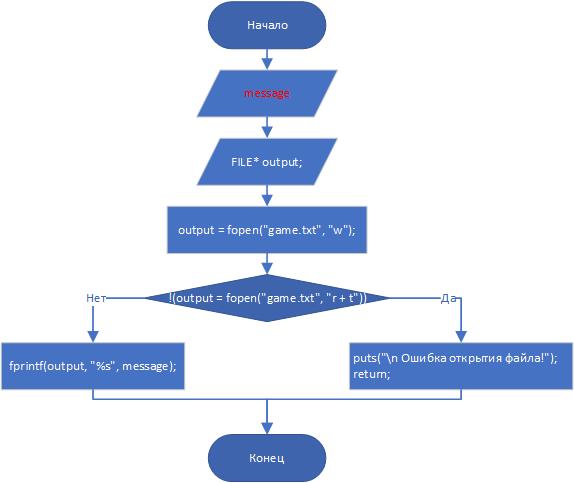


Рисунок 9 – блок-схема алгоритма записи сообщения в файл

## **3. Тестирование программы**

Для проверки работоспособности программы вносим консоль значения описанные в таблице 1. Результаты проверки представлены на рисунках 10 – 14.

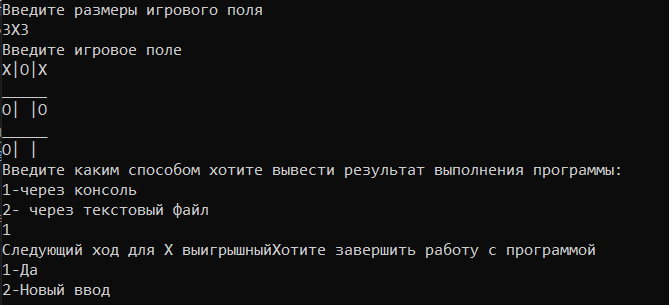


Рисунок 10 – Первый контрольный пример работы программы

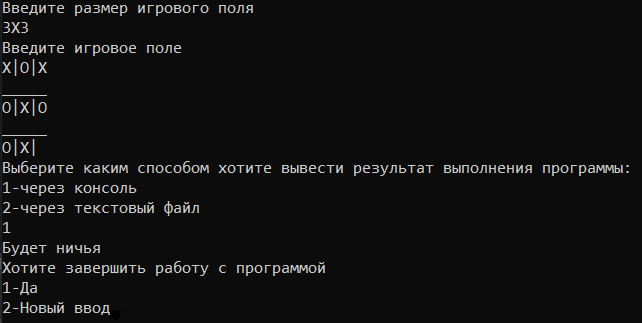


Рисунок 11 – Второй контрольный пример работы программы

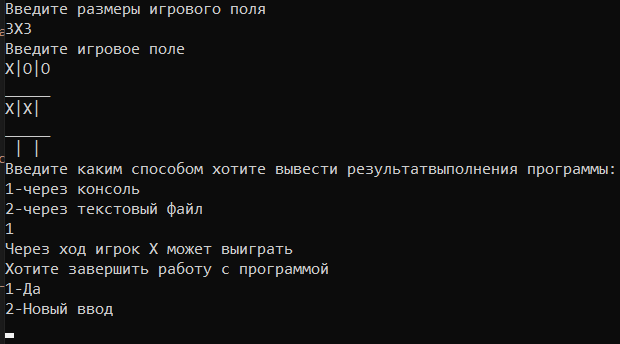


Рисунок 12 – Третий контрольный пример работы программы

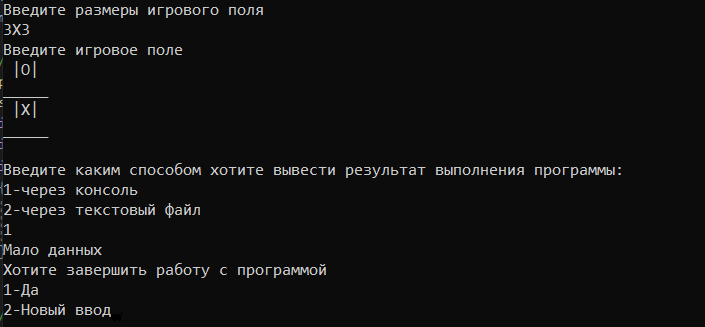


Рисунок 13 – Четвёртый контрольный пример работы программы

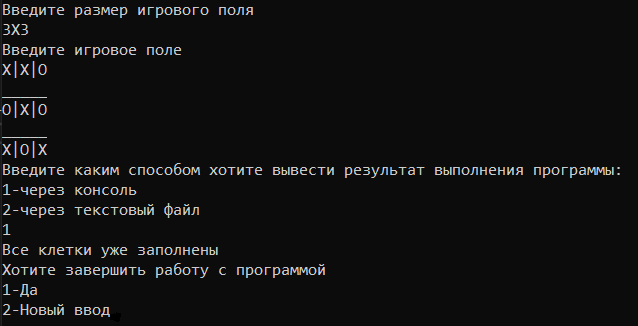


Рисунок 14 – Пятый контрольный пример работы программы

# **Заключение**

В ходе курсового проектирования на языке С с использованием среды разработки Visual Studio 2019 была реализована программа для решения логической задачи «Крестики-нолики».

Разработанная программа должна произвести нахождение выигрышной комбинации на введённом пользователем не ограниченном поле. Реализовать отрисовка игрового поля во время ввода пользователем, а также вывод результатов как в консоль, так и в отдельный текстовый файл. Повторить действия выполнения программы по желанию пользователя.

Все требования к программе выполнены успешно. Реализован интерфейс программы для решения логической задачи «Крестики-нолики». Учтён ошибочный ввод с клавиатуры имеется возможность начать программу заново или выйти из неё

# **Список литературы**

1. Курипта О.В. Основы программирования и алгоритмизации: практикум / О.В.Курипта, О.В. Минакова, Д.К. Проскурин; Воронежский ГАСУ. – Воронеж, 2015. – 132 с.
2. Неземский В.И. Процедуры и функции [Электронный ресурс]: методические указания/ Неземский В.И., Орешкина О.А.— Электрон. текстовые данные. — М.: Московский государственный технический университет имени Н.Э. Баумана, 2009.— 28 c.
3. Алексеев Е. Р. Программирование на языке С++ в среде Qt Creator:/ Е. Р. Алексеев, Г. Г. Злобин, Д. А. Костюк, О. В. Чеснокова А.С. Чмыхало – М. : ALT Linux, 2015. – 448 с.
4. Конова Е. А. Алгоритмы и программы. Язык С++: учебное пособие 2-е изд./ Поллак Г. А — СПб.: Издательство «Лань», 2017. —384 с.
5. Шишкин А.Д. Программирование на языке Си [Электронный ресурс]: учебное пособие/ Шишкин А.Д.— Электрон. текстовые данные.— СПб.: Российский государственный гидрометеорологический университет, 2003.— 104 c.
6. Кузнецов М.В. С++ мастер-класс в задачах и примерах/ М.В. Кузнецов, И.В. Симдянов. – СПб.: БХВ-Петербург, 2007.—480с.

# **Приложения**

//Поключение библиотек

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <iostream>

#include <locale.h>

#include <stdio.h>

#include <malloc.h>

#include <stdlib.h>

#include "windows.h"

using namespace std;

//Выяснение чей в данный момент ход

char motion(char cell[100][100], int m, int n)

{

int rate = 0;

char symbol;

if ((m \* n) % 2 != 0)

{

for (int i = 0; i < m; i++)

{

for (int j = 0; j < n; j++)

{

if (cell[i][j] == ' ') rate++;

};

};

if ((rate%2) != 0) symbol = 'X';

else

{

if (rate == 0) printf("Все клетки уже заполнены");

else symbol = 'O';

};

}

else

{

for (int i = 0; i < m; i++)

{

for (int j = 0; j < n; j++)

{

if (cell[i][j] == ' ') rate++;

};

};

if ((rate%2) != 0) symbol = 'O';

else

{

if (rate == 0) printf("Все клетки уже заполнены");

else symbol = 'X';

};

};

return (symbol);

}

//Проверка на нехватку данных

void lack\_of\_data(char cell[100][100], int m, int n)

{

int rate = 0;

for (int i = 0; i < m; i++)

{

for (int j = 0; j < n; j++)

{

if (cell[i][j] == ' ') rate++;

};

};

if (rate > ((m \* n) / 2)) printf("Мало данных");

}

//Проверка комбинаций

int victory\_check (char cell[100][100], char symbol, int m, int n)

{

int totaliser = 0, victory = 0;

string message;

//Проверка по диагонали

for (int i = 0; i < m; i++)

{

for (int j = 0; j < n; j++)

{

if (i == j)

{

if (cell[i][j] == symbol || cell[i][j] == ' ') totaliser++;

};

};

};

if (totaliser == m || totaliser == n) victory = 1;

//Проверка по горизонтали

for (int i = 0; i < m; i++)

{

totaliser = 0;

for (int j = 0; j < n; j++)

{

if (cell[i][j] == symbol || cell[i][j] == ' ') totaliser++;

};

if (totaliser == n) victory = 1;

};

//Проверка по вертикале

for (int j = 0; j < n; j++)

{

totaliser = 0;

for (int i = 0; i < m; i++)

{

if (cell[i][j] == symbol || cell[i][j] == ' ') totaliser++;

};

if (totaliser == m)victory = 1;

};

return (victory);

}

//Вывод результатов проверки

string output\_message(char cell[100][100], char symbol, int victory, int m, int n)

{

string message;

if (victory == 1) message = "Следующий ход для %c выигрышный", symbol;

else

{

if (symbol == 'X') symbol = 'O';

else symbol = 'X';

victory = victory\_check(cell, symbol, m, n);

if (victory == 1) message = "Через ход игрок %c может выиграть", symbol;

if (victory == 0) lack\_of\_data(cell, m, n);

else message = "Будет ничья";

};

return (message);

}

//Ввод результата в файл

void output\_in\_file(string message)

{

FILE\* output;

output = fopen("game.txt", "w");

if (!(output = fopen("game.txt", "r + t")))

{

puts("\n Ошибка открытия файла!");

return;

}

else fprintf(output, "%s", message);

}

//Основная функция

int main()

{

setlocale(LC\_ALL, "RUS");

system("clear");

//Объявление переменных

int m, n, rate = 0 ,victory, output, end;//n-столдцы, m-строки

char cell[100][100], symbol;

string message;

do {

system("clear");//Очистка экрана

//Ввод значений

printf("Введите размер игрового поля\n");

scanf\_s("%iX%i", &m, &n);

//Заполнение массива значениями и отрисовка интерфейса

printf("Введите игровое поле\n");

for (int i = 0; i < m; i++)

{

for (int j = 0; j < n; j++)

{

scanf\_s("%c", &cell[i][j]);

if (j != (n--)) printf("|");//Отрисовка боковых линий

};

//Отрисовка нижней линии

if (i != (m--))

{

for (int i = 0; i < (n + (n - 1)); i++) printf("\_");

};

};

//Выполнение функций

symbol = motion(cell, m, n);

victory = victory\_check(cell, symbol, m, n);

message = output\_message(cell, symbol, victory, m, n);

//Выбор способа вывода результата

printf("Введите каким способом хотите вывести результат выполнения программы:\n1-через консоль\n2-через текстовый файл\n");

scanf\_s("%d", &output);

if (output == 1) printf("%s", message);

if (output == 2) output\_in\_file(message);

if (output != 2 && output != 1)

{

printf("Неправильный ввод, завершение работы с программой");

system("pause");

};

//Меню выхода из программы

printf("Хотите завершить работу с программой\n1-Да\n2-Новый ввод\n");

scanf\_s("%d", &end);

if (end != 1 && end != 2)

{

printf("Неправильный ввод, завершение работы с программой");

system("pause");

};

} while (end == 1);

system("pause");

}