Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

**«Пермский национальный исследовательский**

**политехнический университет»**

Электротехнический факультет

Кафедра «Информационные технологии и автоматизированные системы»

направление подготовки: 09.03.01 - «Информатика и вычислительная техника»

**Отчет по Лабораторной работе 8 ферзей**

**по дисциплине**

**«Основы алгоритмизации и программирования»**

Выполнил студент гр. ИВТ-23-1б

Аблалимов Алексей сергеевич

Проверил:

Ст. Преподаватель кафедры ИТАС

Яруллин Денис Владимирович

(оценка) (подпись)

г. Пермь - 2024

**Постановка задачи:**

Расставить 8 ферзей на шахматной доске так, чтобы они не били друг друга

**Анализ задачи:**

int board[8][8];

void setQueen(int i, int j)

{

    for (int x = 0; x < 8; x++)

    {

        ++board[x][j];

        ++board[i][x];

        int foo;

        foo = j - i + x;

        if (foo >= 0 && foo < 8)

        {

            ++board[x][foo];

        }

        foo = j + i - x;

        if (foo >= 0 && foo < 8)

        {

            ++board[x][foo];

        }

    }

    board[i][j] = -1;

}

Объявляю глобальную переменную board – двумерный массив 8x8. Функция setQueen, как понятно из названия устанавливает Королеву на доску по координатам i и j.

Сначала доску(board) - глобальная переменная, объявляю как двумерный массив (8х8). После данная функция (Func1) – устанавливает ферзя в клетку с координатами (i, j). C помощью внешних и вложенных циклов смещаем ферзя по клеткам. По координатам где стоит ферзь разместим (-1). Если в массиве встретится (-1), значит там стоит ферзь.

void resetQueen(int i, int j)

{

    for (int x = 0; x < 8; x++)

    {

        --board[x][j];

        --board[i][x];

        int foo;

        foo = j - i + x;

        if (foo >= 0 && foo < 8)

        {

            --board[x][foo];

        }

        foo = j + i - x;

        if (foo >= 0 && foo < 8)

        {

            --board[x][foo];

        }

    }

    board[i][j] = 0;

}

Данная функция служит для удаления ферзя, в случае если будет пересечение, или свободных клеток нет.

bool tryQu(int i)

{

    bool result = false;

    for (int j = 0; j < 8; j++)

    {

        if (board[i][j] == 0)

        {

            setQu(i, j);

            if (i == 7)

            {

                result = true;

            }

            else

            {

                if (!(result = tryQu(i + 1)))

                {

                    resetQu(i, j);

                }

            }

        }

        if (result)

        {

            break;

        }

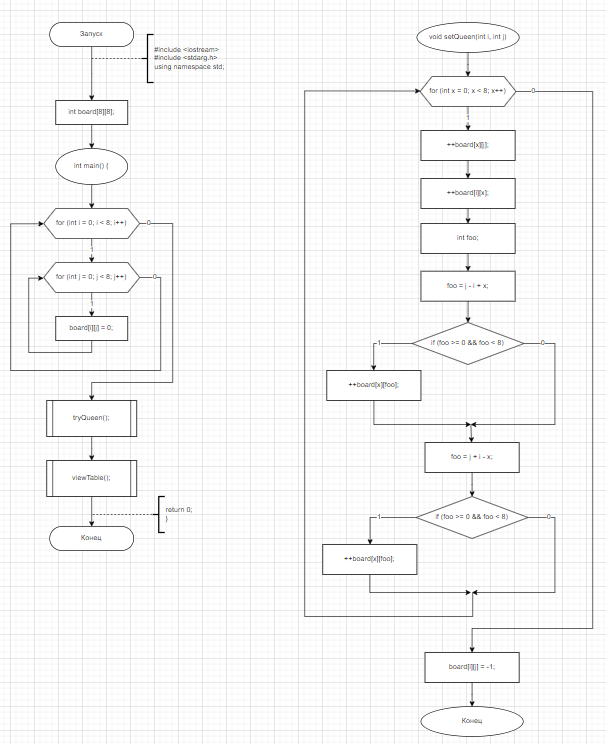
    }

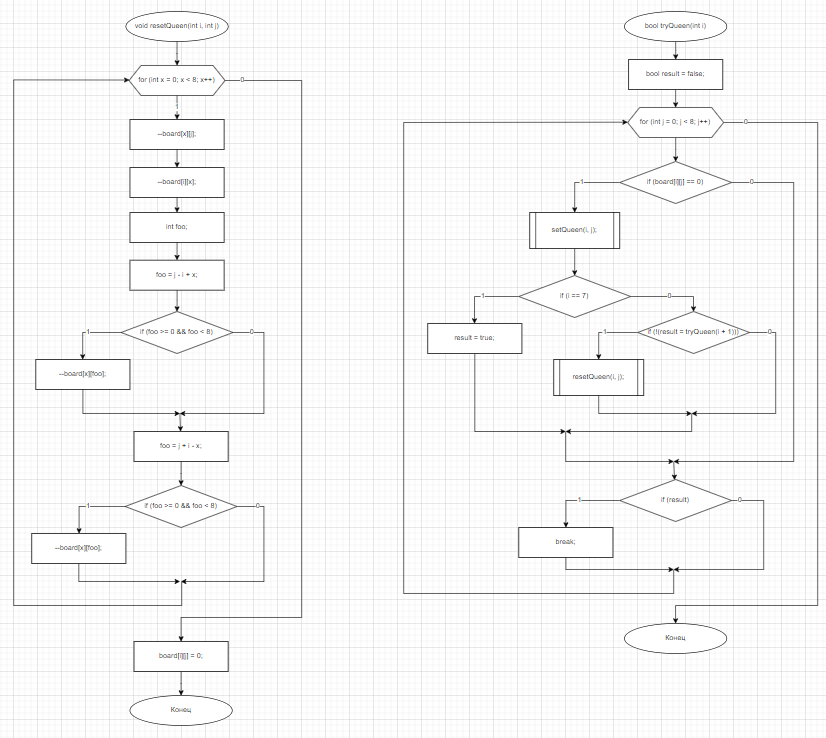
    return result;

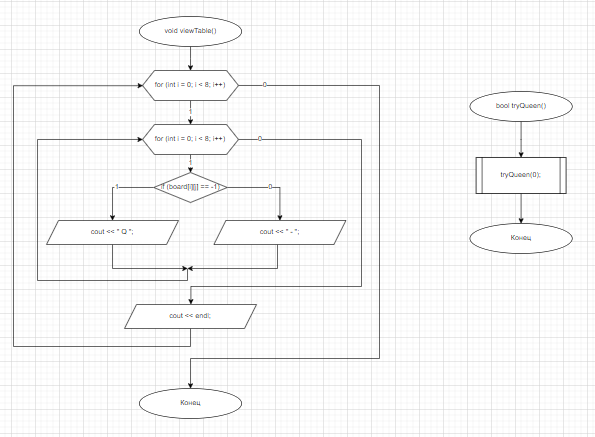
}

Данная функция пытается поставить Королеву и, если это возможно, возвращает истину, в противном случае ложь.

**Блок-схема**

****

****

****

**Код программы**

#include <iostream>

using namespace std;

int board[8][8];

void setQueen(int i, int j)

{

    for (int x = 0; x < 8; x++)

    {

        ++board[x][j];

        ++board[i][x];

        int foo;

        foo = j - i + x;

        if (foo >= 0 && foo < 8)

        {

            ++board[x][foo];

        }

        foo = j + i - x;

        if (foo >= 0 && foo < 8)

        {

            ++board[x][foo];

        }

    }

    board[i][j] = -1;

}

void resetQueen(int i, int j)

{

    for (int x = 0; x < 8; x++)

    {

        --board[x][j];

        --board[i][x];

        int foo;

        foo = j - i + x;

        if (foo >= 0 && foo < 8)

        {

            --board[x][foo];

        }

        foo = j + i - x;

        if (foo >= 0 && foo < 8)

        {

            --board[x][foo];

        }

    }

    board[i][j] = 0;

}

bool tryQueen(int i)

{

    bool result = false;

    for (int j = 0; j < 8; j++)

    {

        if (board[i][j] == 0)

        {

            setQueen(i, j);

            if (i == 7)

            {

                result = true;

            }

            else

            {

                if (!(result = tryQueen(i + 1)))

                {

                    resetQueen(i, j);

                }

            }

        }

        if (result)

        {

            break;

        }

    }

    return result;

}

bool tryQueen()

{

    return tryQueen(0);

}

void viewTable(int i, int j)

{

    for (int i = 0; i < 8; i++)

    {

        for (int j = 0; j < 8; j++)

        {

            if (board[i][j] == -1)

            {

                cout << " Q ";

            }

            else

            {

                cout << " - ";

            }

        }

        cout << endl;

    }

}

int main()

{

    for (int i = 0; i < 8; i++)

    {

        for (int j = 0; j < 8; j++)

        {

            board[i][j] = 0;

        }

    }

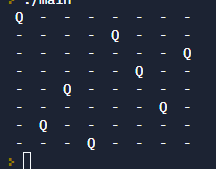
    tryQueen();

    viewTable();

    return 0;

}

**Вывод программы**

****