МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ПЕРМСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

ЛАБОРАТОРНАЯ РАБОТА   
на тему «Восемь ферзей»

Выполнила: студентка группы РИС-23-2Б Жилина Анастасия Александровна

Проверила: доцент кафедры ИТАС О.А.Полякова

2024**Разработка алгоритма нахождения максимального элемента в массиве.**

1. **Постановка задачи** – Дана шахматная доска 8 на 8. Расставить на доске 8 ферзей так, чтобы в каждой строке стоял 1 ферзь, пр этом расстановка должна быть такой, чтобы ни один ферзь не бил никакого другого.

Примечание: ферзь ходить и бьет по вертикали, горизонтали и диагонали.

1. **Анализ задачи:**

1) Выполнить расстановку через рекурсивную функцию;

2) Вначале ферзь ставится на 1 строку свободной ячейки. В матрице расстановки это координата [0, 0];

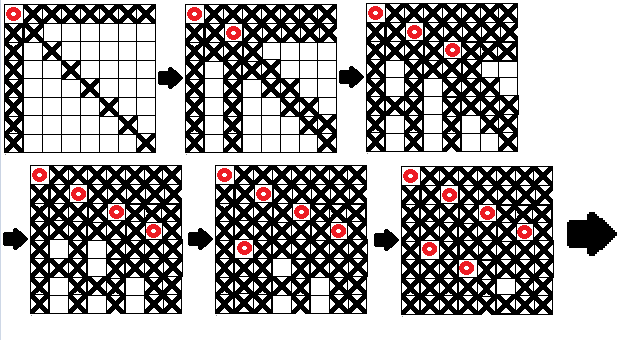
3) Отмечаем все клетки, которые бьет ферзь;

4) Переходим на следующую строку и повторяем шаги 2-3;

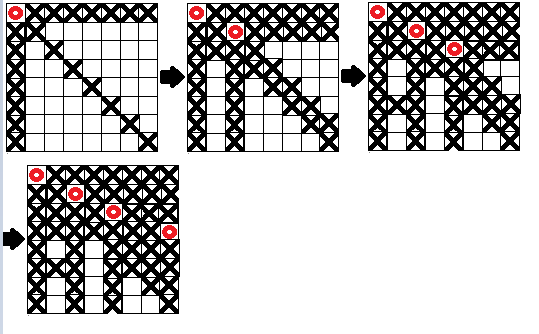
5) В случае, если в очередной строке нет свободной клетки, то рекурсивно возвращаемся на столько клеток обратно, чтоб расстановка ферзей удовлетворяла условию задачи.

Визуализация:

На первых 6ти шагах видно, что далее некуда будет поставить 8го ферзя. Поэтому необходимо вернуться на несколько шагов назад.



Вернемся на 4тый шаг и поставим ферзя во вторую свободную ячейку.



Таким образом, пока полностью 8 ферзей не будут расставлены, выполняются данные манипуляции.

1. **Листинг программы**

#include <iostream>

using namespace std;

const int N = 8;//размер стороны доски

int A[N][N];//шахм.доска

//"Кресты" - метка о запрете на постановку ферзя в данной ячейке

void draw\_X(int n, int m)

{

for (int j = 0; j < N; j++)//расставляет кресты по горизонтали

{

if (A[n][j] != -1) ++A[n][j];

}

for (int i = 0; i < N; i++)//расставляет кресты по вертикали

{

if (A[i][m] != -1) ++A[i][m];

}

//далее 2 цикла - расстановка крестов на диагонали элемента-ферзя с левого верхнего угла до правого нижнего

int j = m - 1;//расставляет кресты по диагонали от элемента влево вверх

for (int i = n - 1; i >= 0 && j >= 0; i--)

{

++A[i][j];

j--;

}

j = m + 1;//расставляет кресты по диагонали от элемента вправо вниз

for (int i = n + 1; i < N && j < N; i++)

{

++A[i][j];

j++;

}

//далее 2 цикла - расстановка крестов на диагонали элемента-ферзя с левого нижнего угла до правого верхнего

j = m + 1;//расставляет кресты по диагонали от элемента вправо вверх

for (int i = n - 1; i >= 0 && j < N; i--)

{

++A[i][j];

j++;

}

j = m - 1;//расставляет кресты по диагонали от элемента вправо вниз

for (int i = n + 1; i < N && j >= 0; i++)

{

++A[i][j];

j--;

}

}

void delete\_X(int &n, int &m)

{

for (int j = 0; j < N; j++)//убирает кресты с горизонтали

{

if (A[n][j] != -1) --A[n][j];

}

for (int i = 0; i < N; i++)//убирает кресты с вертикали

{

if (A[i][m] != -1) --A[i][m];

}

//далее 2 цикла - удаление крестов с диагонали элемента-ферзя с левого верхнего угла до правого нижнего

int j = m - 1;//убирает кресты с диагонали от элемента влево вверх

for (int i = n - 1; i >= 0 && j >= 0; i--)

{

--A[i][j];

j--;

}

j = m + 1;//убирает кресты с диагонали от элемента вправо вниз

for (int i = n + 1; i < N && j < N; i++)

{

--A[i][j];

j++;

}

//далее 2 цикла - удаление крестов с диагонали элемента-ферзя с левого нижнего угла до правого верхнего

j = m + 1;//убирает кресты с диагонали от элемента вправо вверх

for (int i = n - 1; i >= 0 && j < N; i--)

{

--A[i][j];

j++;

}

j = m - 1;//убирает кресты с диагонали от элемента вправо вниз

for (int i = n + 1; i < N && j >= 0; i++)

{

--A[i][j];

j--;

}

++A[n][m];//ставим 0 (нач.положение ячейки) вместо ферзя

j = m + 1;

while (A[n][j] != 0 && j < N)//ищем, есть ли еще свободная ячейка в строке

{

++j;

}

if (j == N)//если нет, то удаляем и кресты от ферзя выше

{

n = n - 1;

for (int i = 0; i < N; i++)//ищем столбец, в котором расположен ферзь выше

{

if (A[n][i] == -1)

{

m = i; break;

}

}

delete\_X(n, m);

}

else

{

m = j;

}

}

void ferz(int n, int m)

{

if (A[n][m] == 0)//проверка на свободную ячейку

{

A[n][m] = -1;//постановка ферзя

draw\_X(n, m);//расстановка крестов после постановки ферзя

int j = 0;

if (n < N - 1)

{

while (A[n + 1][j] != 0 && j < N)//ищем индекс столбца след.свободной ячейки

{

++j;

}

if (j == N)//если не нашлось такого индекса (то есть вся строчка состоит из Х)

{

delete\_X(n, m);//удаляем кресты, которые проставил этот ферзь + запоминаем новые м и н для расстановки ферзя

ferz(n, m);

}

else

{

m = j;

ferz(n + 1, m);

}

}

}

}

int main()

{

setlocale(LC\_ALL, "Rus");

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

{

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

{

A[i][j] = 0;

}

}

ferz(0,0);

for (int i = 0; i < N; i++)//печать вывода

{

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

{

if (A[i][j] == -1) cout << "Q";

else cout << ".";

}

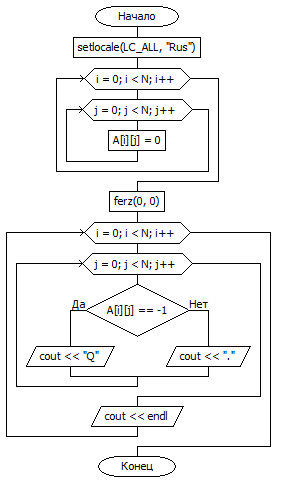
cout << endl;

}

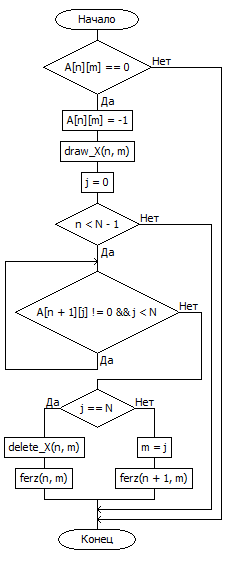
}

1. **Блок-схема**

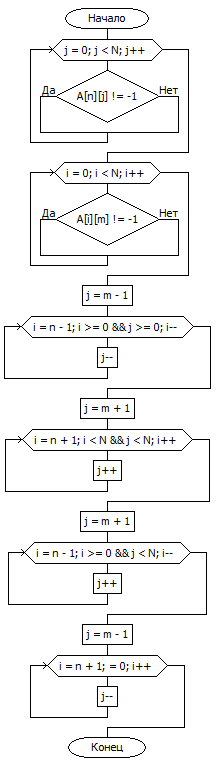
Главная фукция:



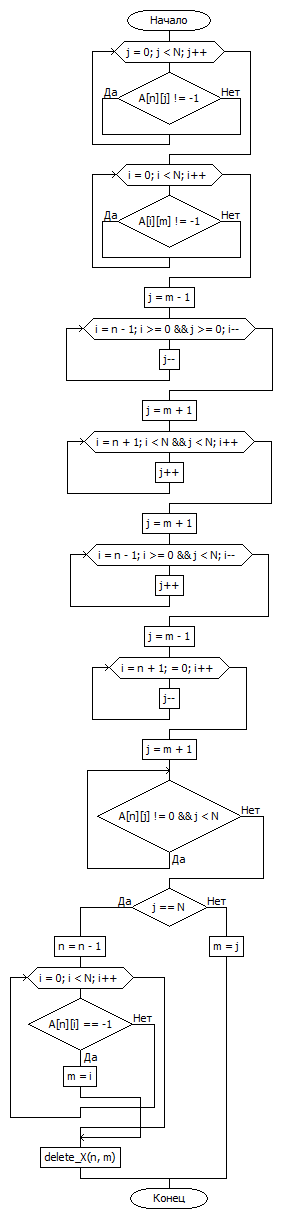
Функция «ферзь»:



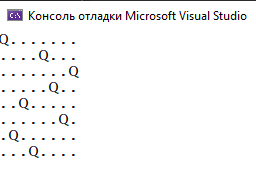
Функция по расстановке клеток, которые бьет ферзь:



Функция, которая удаляет бьющиеся клетки при удалении ферзя



1. **Результаты работы**



Разместим готовые исполнимые файлы в созданный репозиторий на GitHub

