Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Пермский национальный исследовательский

политехнический университет»

Кафедра “Информационные технологии и автоматизированные системы”

**О Т Ч Ё Т**

**по Лабораторной работе №2 вариант 6**

Дисциплина: основы алгоритмизации и программирования

Тема: Вычисление функций с использованием их разложения в степенной ряд.

Выполнил работу

Студент группы ИВТ-24-2б

Роговик.Н.А

Проверила

Доцент кафедры ИТАС

Полякова.О.А

Пермь, 2024

**Постановка задачи**

1. Необходимо реализовать алгоритм для поиска всех возможных решений задачи.
2. Программа должна выводить все найденные решения в виде расположения ферзей на доске.
3. Решение должно включать эффективные методы для проверки конфликтов между ферзями.

**Решение**

**Объявление констант и переменных**:

const int N = 8;

int board[N][N];

int solutionsCount = 0;

Объявляем размер доски и переменные для хранения состояний доски и количества решений.

**Функция проверки безопасности позиции**:

bool isSafe(int row, int col) {

// Проверяем столбец на левую сторону

for (int i = 0; i < row; ++i) {

if (board[i][col] == 1) return false;

}

// Проверяем левую верхнюю диагональ

for (int i = row, j = col; i >= 0 && j >= 0; --i, --j) {

if (board[i][j] == 1) return false;

}

// Проверяем правую верхнюю диагональ

for (int i = row, j = col; i >= 0 && j < N; --i, ++j) {

if (board[i][j] == 1) return false;

}

return true;

}

Эта функция проверяет, безопасно ли размещение ферзя в указанной позиции, проверяя столбец, левую и правую верхние диагонали.

**Функция для вывода решения**:

void printSolution() {

cout << "Решение " << ++solutionsCount << ":" << endl;

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

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

if (board[i][j] == 1) {

cout << "Q ";

} else {

cout << ". ";

}

}

cout << endl;

}

cout << endl;

}

Функция выводит текущее решение, увеличивая счётчик решений и выводя доску на экран.

**Рекурсивная функция для решения задачи**:

void solveNQueensUtil(int row) {

if (row >= N) {

printSolution();

return;

}

for (int col = 0; col < N; ++col) {

if (isSafe(row, col)) {

board[row][col] = 1;

solveNQueensUtil(row + 1);

board[row][col] = 0;

}

}

}

Эта рекурсивная функция пытается разместить ферзя в каждой строке, проверяя безопасность позиции и вызывая саму себя для следующей строки.

**Функция для инициализации доски и запуска решения**:

void solveNQueens() {

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

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

board[i][j] = 0;

}

}

solveNQueensUtil(0);

}

Функция инициализирует доску нулями и запускает рекурсивную функцию с первой строки.

**Главная функция**:

int main() {

solveNQueens();

return 0;

}

Основная функция, запускающая процесс решения задачи о 8 ферзях.

Изображение выглядит как текст, снимок экрана, число, документ

Автоматически созданное описание

Изображение выглядит как текст, диаграмма, Параллельный, План

Автоматически созданное описание