**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №1**

**по дисциплине «Построение и анализ алгоритмов»**

Тема: «Поиск с возвратом»

|  |  |  |
| --- | --- | --- |
| Студент гр. 6381 |  | Токун М.С. |
| Преподаватель |  | Филатов А. Ю. |

Санкт-Петербург

2018

**Цель работы.**

Изучение алгоритма поиска на основе задачи минимального квадрирования квадрата размером - N x N.

**Описание алгоритма.**

На вход подается число - размер стороны квадрата. Проверяется его делимость на 2, 5 и 3.

В случае если число делится на 2, 3 или 5, то используется оптимизация, которая может быть осуществлена, поскольку наименьшее число квадратов, на которое можно разбить большой квадрат, зависит от наименьшего делителя числа, и если число не является простым, то его наименьшим делителем является 2, 3 или 5.

Для оптимизации алгоритма в этом случае функция бэктрекинга строит разбиение для простого числа, являющегося наименьшем делителем исходного числа, а затем, перед выводом получившегося разбиения на экран, все координаты и длинны умножается на коэффициент:

coef = исходное число / наименьший делитель

Далее, для оптимизации работы программы, происходит заполнение квадратом со стороной (n+1)/2 и двумя квадратами со стороной (n-1)/2 в левый верхний, правый верхний и левый нижний углы соответственно, или заполнения четырьмя квадратами со стороной size/2, если размер квадрата четный. Затем осуществляется вход в рекурсивную функцию для оставшейся незаполненной части.

В рекурсивной функции начинается заполнение пустого места квадратами, начиная с квадратов длиной 1х1 и двигаясь слева направо сверху вниз. Наименьшее число полученных квадратов хранится в отдельной переменной, и как только во время перебора текущее число квадратов становится больше или равно наименьшему, либо заканчиваются пустые ячейки и число квадратов оказывается больше полученного на тот момент наименьшего, решение забывается и делается шаг назад в поисках более оптимального решения. Если же по окончании пустых ячеек квадратов получилось меньше, чем наименьшее их число, полученное на тот момент, программа перезаписывает лучшее решение. В итоге программа получит разделение на наименьшее количество квадратов.

**Описание классов и функций.**

**class Table –** класс, реализует заполняемый квадрат. Класс содержит следующее приватное поле table – массив содержащий квадрат, и публичные поля: private: table\_size – размер квадрата, min\_table\_num – минимальное количество квадратов для разбиения, min\_table – массив для сохранения минимально заполненного квадрата.

**bool findZeroTable(int& x, int& y)** – проверка наличия пустых клеток в поле, начиная с точки (x;y).

**void makeFullTable(int size, Table\* &table) –** функция с вызовами функций setTable и fillTable для заполнения квадрата.

**void printResult(int coef, Table\* &table)** – функция вывода конечного результата.

**void setZeroTable(int x, int y, int size)** – функция обнуления квадрата размером size, начиная с точки (x;y).

**void saveTable(int table\_num)** – функция сохранения текущего заполненного квадрата с количеством входящих квадратов square\_num.

**void fillTable(int x, int y)** – рекурсивная функция заполнения поля, начиная с точки (x;y).

**bool setTable(int x, int y, int size, int number)** – установка квадрата с номером number и размером size начиная с точки (x;y).

**Тестирование.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Номер теста | 1 | 2 | 3 | 4 | 5 | 6 |
| Результат |  |  |  |  |  |  |

**Вывод.**

В процессе выполнения лабораторной работы был изучен и реализован алгоритм поиска с возвратом для построения минимального квадрирования квадрата размером N x N.