**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 |  | Кухарев М.А. |
| Преподаватель |  | Филатов А.Ю. |

Санкт-Петербург

2018

**Цель работы:**

Изучение алгоритма поиска с возвратом (backtracking).

**Условия задания:**

У Вовы много квадратных обрезков доски. Их стороны (размер) изменяются от 1 до *N*−1, и у него есть неограниченное число обрезков любого размера. Но ему очень хочется получить большую столешницу - квадрат размера *N*. Он может получить ее, собрав из уже имеющихся обрезков (квадратов).



Рисунок 1. Пример разбиения квадрата со стороной 7.

Внутри столешницы не должно быть пустот, обрезки не должны выходить за пределы столешницы и не должны перекрываться. Кроме того, Вова хочет использовать минимально возможное число обрезков.

**Входные данные:**

Размер столешницы - одно целое число ***N***(2≤***N***≤20).

**Выходные данные:**

Одно число *K*, задающее минимальное количество обрезков (квадратов), из которых можно построить столешницу (квадрат) заданного размера *N*. Далее должны идти *K* строк, каждая из которых должна содержать три целых числа *x*,*y* и *w*, задающие координаты левого верхнего угла (1≤*x*, *y*≤*N*) и длину стороны соответствующего обрезка (квадрата).

**Пример входных данных**7 **Соответствующие выходные данные**9  
1 1 2  
1 3 2  
3 1 1  
4 1 1  
3 2 2  
5 1 3  
4 4 4  
1 5 3  
3 4 1

**Описание функций**

|  |  |
| --- | --- |
| void printAnswer( std::vector<answer> out) | Выводит ответ, состоящий из точек p и их количества |
| size\_t fill\_rect( std::vector<std::vector<int> >\* input\_arr, size\_t limit, point p, size\_t sq\_num) | Рисование квадрата: рисует квадрат в поле arr, размеров limit, начиная от точки p, “цвета” sq\_num |
| Bool def\_free(std::vector<std::vector<int> > input\_arr, point\* p) | Поиск точки: находит первую пустую точку и возвращает существует ли она |
| bool fit\_tight( std::vector<std::vector<int> > input\_arr, point p, size\_t rec\_size ) | Проверяет, есть ли пустоты более чем за одной стороной квадрата |
| void trace( std::vector<std::vector<int> > input\_arr , size\_t boundary, size\_t sq\_num, std::vector<answer>\* ans, std::vector<answer> comp\_arr = std::vector<answer>()) | Основная функция trace рекурсивна, суть её в следующем: Находится пустая точка, строится от неё максимально большой квадрат, если квадрат заходит ровно, то рисование продолжается в цикле, каждая точка, от которой происходит прорисовка, заносится в буферный ответ.  На каждой итерации цикла снова происходит проверка на то, насколько ровно зашёл квадрат, если не ровно, то от этой точки вызывается рекурсивная функция с параметром максимального размера, уменьшенным на 1. |

**Тестирование:**

|  |  |
| --- | --- |
| **Ввод** | **Вывод** |
| **5** |  |
| **9** |  |
| **13** |  |

**Вывод:**

В процессе выполнения лабораторной работы был изучен, разработан и реализован алгоритм поиска с возвратом для поиска ответа на задачу о минимальном заполнении квадрата размером NxN квадратами.