Московский Авиационный Институт

(Национальный Исследовательский Университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

**Лабораторная работа №7 по курсу**

**«Дискретный анализ»**

**Жадные алгоритмы**

Студент: Куценко Максим Дмитриевич

Группа: М8О–312Б-22

Вариант: 6

Преподаватель: Н.Д. Глушин

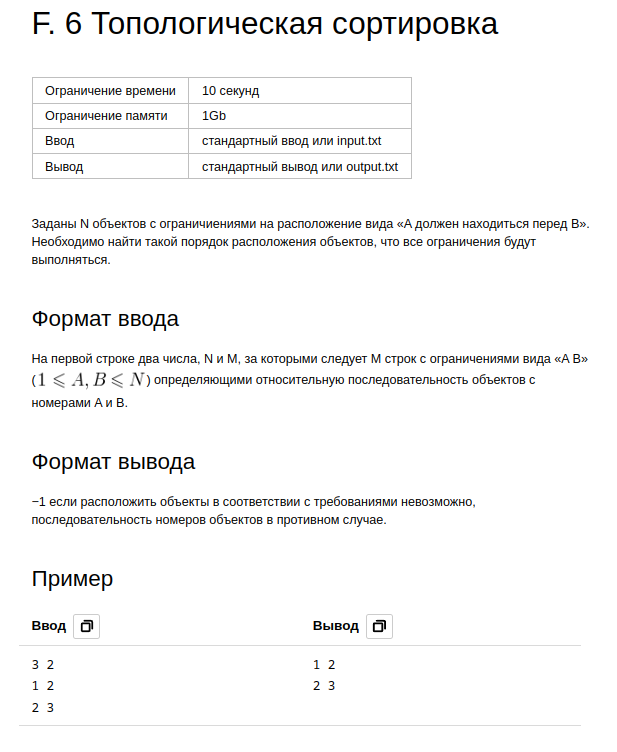
Оценка: \_\_\_\_\_\_\_\_\_\_

Дата: \_\_\_\_\_\_\_\_\_\_

Подпись: \_\_\_\_\_\_\_\_\_\_

Москва, 2024.

**Условие**

****

**Метод решения**

Данная задача решается при помощи поиска в глубину с “окрашиванием” вершин в целях проверки на цикличность.

Каждая вершина изначально белого цвета. Когда приходим в неё в процессе DFS, красим её в серый. Когда уходим из вершины после завершения обхода всех её потомков, красим вершину в чёрный и откладываем её к отсортированным.

Если мы приходим в серую вершину, то значит в графе есть цикл и топологическая сортировка невозможна.

**Описание программы**

В классе графа храним вектор списков смежности edges, вектор цветов вершин states, булевую переменную цикличности has\_cycles и вектор результата result.

Топологическая сортировка состоит из двух отдельных функций — top\_sort() и dfs\_visit(u). В top\_sort мы подготавливаем цвета, булевую переменную и вектор, затем проводим dfs\_visit(u) для каждой вершины, если она белая. В dfs\_visit(u) красим вершину в серый, берём список смежности для вершины и идём по нему, выбирая белые вершины для посещения. Если видим серую вершину, то принимаем has\_cycles = true.

Когда мы прошли все вершины в списке, меняем цвет вершины на чёрный и добавляем её в конец result. Отражаем result (хотя можно использовать и стек) и выводим результат.

**Дневник отладки**

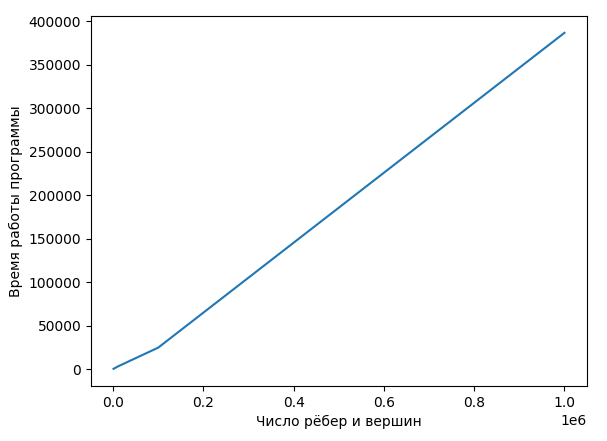
1) Слишком поздно красил вершину в чёрный, из-за этого программа видела циклы даже там, где две вершины шли в одну.

2) Получал ошибку из-за индексации.

3) Из-за неочевидной ошибки в логике решил переделать алгоритм со стека на рекурсию.

**Тест производительности**

График времени создания дерева

Сложность данного алгоритма — O(V + E), как у поиска в глубину.

**Выводы**

Вспомнил работу с графами и алгоритм поиска в глубину, использовал его на практике для решения данной задачи. Топологическая сортировка весьма полезна в такой задаче как job scheduling.