Лабораторная работа № 7 по курсу дискретного анализа: жадные алгоритмы

Выполнил студент группы М08-312Б МАИ *Лобанов Олег*.

Условие

Дана последовательность длины N из целых чисел 1, 2, 3. Необходимо найти минимальное количество обменов элементов последовательности, в результате которых последовательность стала бы отсортированной.

Входные данные: число N на первой строке и N чисел на второй строке.

Выходные данные: минимальное количство обменов.

Вариант 5. Оптимальная сортировка чисел.

Метод решения

Сначала записываю все цифры, не превышающие 3, в вектор *counts*, для подсчета количества единиц, двоек и троек. Далее произвожу инкрементацию соответсующего счетчика в векторе *counts*, проходя по всем элементам массива *numbers*.

Осуществляю перестановку индексов в векторе *counts* так, что, начиная с последней, в каждую последующую ячейку записываю суммы предыдущих, а в нулевую ячейку - 0.

Цикл проходит по каждому элементу массива *numbers* и проверяет, нужно ли поменять местами текущий эелемент с другим, чтобы соблюсти порядок "единицы-двойки-тройки". Если текущая позиция меньше *counts*[1] (то есть находится в зоне единиц, но значение равно 2 или 3), то выполняется поиск подходящей позиции для обмена. Аналогично обрабатываются случаи, когда текущая позиция находится между *counts*[1] и *counts*[2] (то есть в зоне двоек) и нужно проверить наличие лишних троек.

Когда подходящий элемент найден, производится обмен значениями через временную переменную *tmp*, и увеличивает счетчик *count*.

В конце получаем итоговое количество обменов, которое необходимо для приведения массива к нужному порядку.

Описание программы

SwapNumbers - основная функция, в которой происходит подсчет обменов, *count* - счетчик обменов, *counts* - вектор для подсчета количества единиц, двоек и троек.

В *main* происходит считывание входных данных, вызов функции *SwapNumbers*, вывод конечного результата.

Дневник отладки

Перед отправкой обнаружил неточность в конечном ответе. Выяснил, что я ошибся в знаке, из-за чего программы выдавала на одну перестановку меньше. Исправив этот недочет, получилось "окнуть"с первой попытки.

Выводы

После выполнения лабораторной работы я познакомился с жадными алгоритмами. Выяснил, что скорость этого алгоритма $O(n^2)$. По сравнению с линейными алгоритмами сортировок это сильно больше. Зато если swap элементов занимает очень много времени, то эта сортировка будет работать быстрее, чем сортировки за линию. В большинстве случаев ее сложность будет ближе к линеарифметической, за счет того, что средняя сложность поиска ближе к логорифму, чем к $O(n)$. Плюс, памяти для алгоритма требуется линейное количество, так как вектор для подсчета занимает константную память, а вектор с числами для сортировки я не копирую.