Лабораторная работа №8 по курсу дискретного анализа: жадные алгоритмы

Выполнил студент группы М8О-307Б-21 МАИ *Кажекин Денис*.

Условие

Вариант 1 – Размен монет

Метод решения

Метод решение очень, просто необходимо было реализовать жадный алгоритм, который заключался просто в вычислении максимального количества вхождения текущего номинала в сумму на текущий момент.

Описание программы

В архитектуре программы всего один файл greedy.cpp

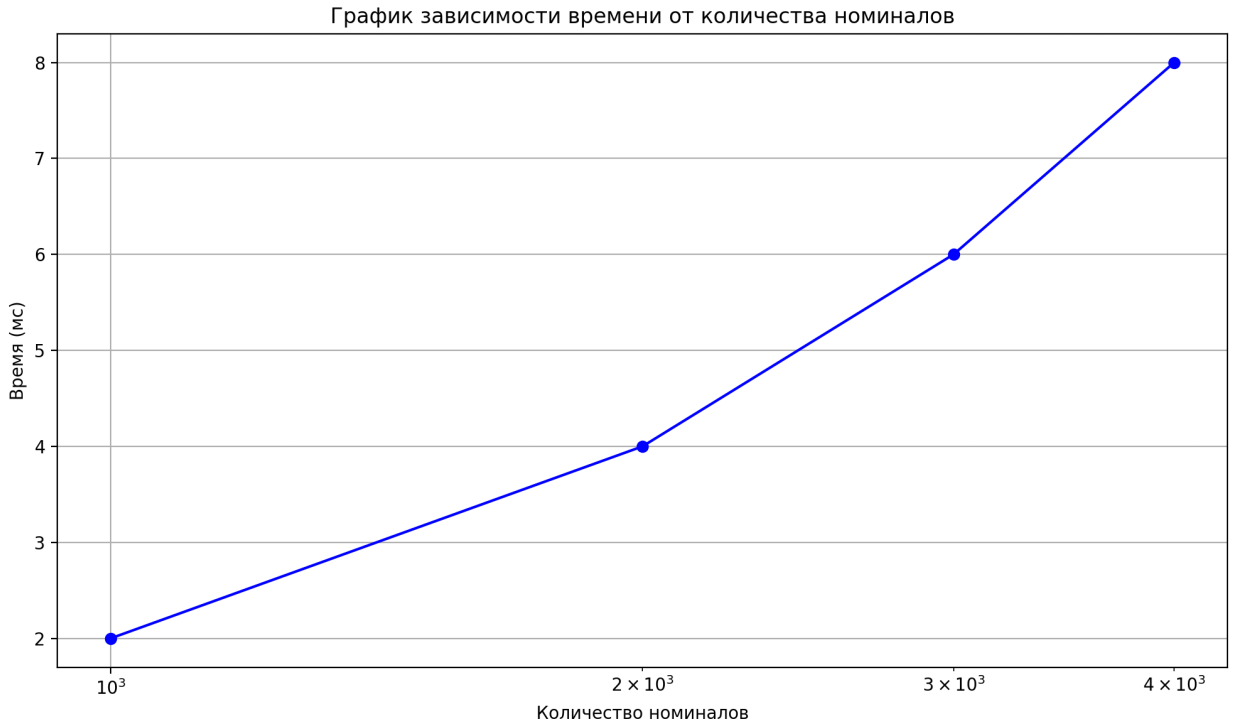
Дневник отладки

Единственный недочет заключался в том, что я не учел переполнение типа int. После исправления на long long программа сработала корректно

Недочёты

Недочеты не выявлены

Тест производительности



Алгоритм имеет сложность O(n*log n) и это подтверждается вычислениями, потому что для каждой основание монеты p возводится в каждую из степеней N. В силу логарифмической сложности возведения в степень, получаем искомую сложность.

Выводы

Жадный алгоритм не применим в общем случае и это доказывается простым контрпримером, когда взятие старшего номинала наибольшее число раз ограничивает нас возможности получить разложение.

Пусть сумма равна 27 монет. И у нас есть номиналы 3, 6, 10. Чтобы использовать наименьшее число номиналов, необходимо жадным алгоритмом брать два раза по десять и получить оставшуюся сумму, равную семи. Затем взять один раз по шесть и получить единицу. Но вот незадача мы не можем разложить единицу по номиналу три. Решение жадным алгоритмом с произвольными номиналами не работает.

Предлагается следующий алгоритм:

Составим матрицу N * S, где N – номиналы, дополненные нулевым номиналом, а S – суммы от 0 до S включительно. Изначально матрицу необходимо заполнить нулями и выполнить следующий обход: Итерируемся по матрице и заполняем в ячейку (i, j) результат деления суммы j на номинал i, если это деление с нулевым остатком. Иначе оставляем ячейку нулевой. После такого обхода нужно совершить точно такой же, но обратный обход матрицы. Если встречается значение отличное от нуля, то необходимо перейти к ячейке (i-1, j - (j - i * (i, j))). Если с помощью такого спуска мы доходим до ячейки (0, 0), то это ответ. Если при спуске мы спустились в ячейку отличную от (0, 0) которая имеет значение в себе равное нулю, то сразу продолжаем обход.

Сложность такого алгоритма должна быть в среднем O(n^2*m)