ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Отчёт по лабораторной работе № 6

«leetcode: [**123. Best Time to Buy and Sell Stock III**](https://leetcode.com/problems/best-time-to-buy-and-sell-stock-iii/)»

Выполнил работу

Никитин Данил Алексеевич

Академическая группа №J3110

Принято

Должность, звание Фамилия Имя преподавателя

Санкт-Петербург

2024

**Структура отчёта:**

1. Введение
2. Понять какие подзадачи у этой задачи.
3. Понять какая структура может хранить решения подзадач.
4. Решить задачу методом динамического программирования.
5. Теоретическая подготовка

Метод динамического программирования.

Простейшие синтаксические конструкции.

1. Реализация

1)Поняли ,что основой является решение именно задачи о нахождение прибыли для одной транзакции.

2) Основную задачу можно разделить на подзадачи – и записывать из результат для использования в решении следующей подзадачи.

Такими подзадачами будет нахождение лучшей транзакции для первых i элементов , затем мы будем сравнивать максимальное значения из них с новым возможным максимальным значением.

3)Заметим ,что чтоб найти вторую транзакцию нам нужно ,проходится справа налево по массиво с последнего до n-i элементы, тогда для любого i первая и вторая транзакции будут согласоованны (одина пара строго правее другой )

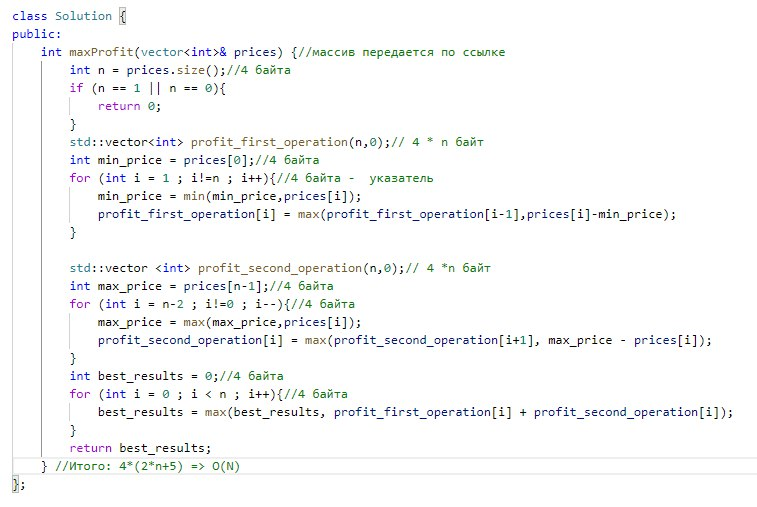
Замечание: Если нам подходит случай , когда совершается только одна транзакциия , то в таблице лучших операций для второй или наоборот первой транзакции будут нули.

1. Экспериментальная часть

Подсчет сложности:

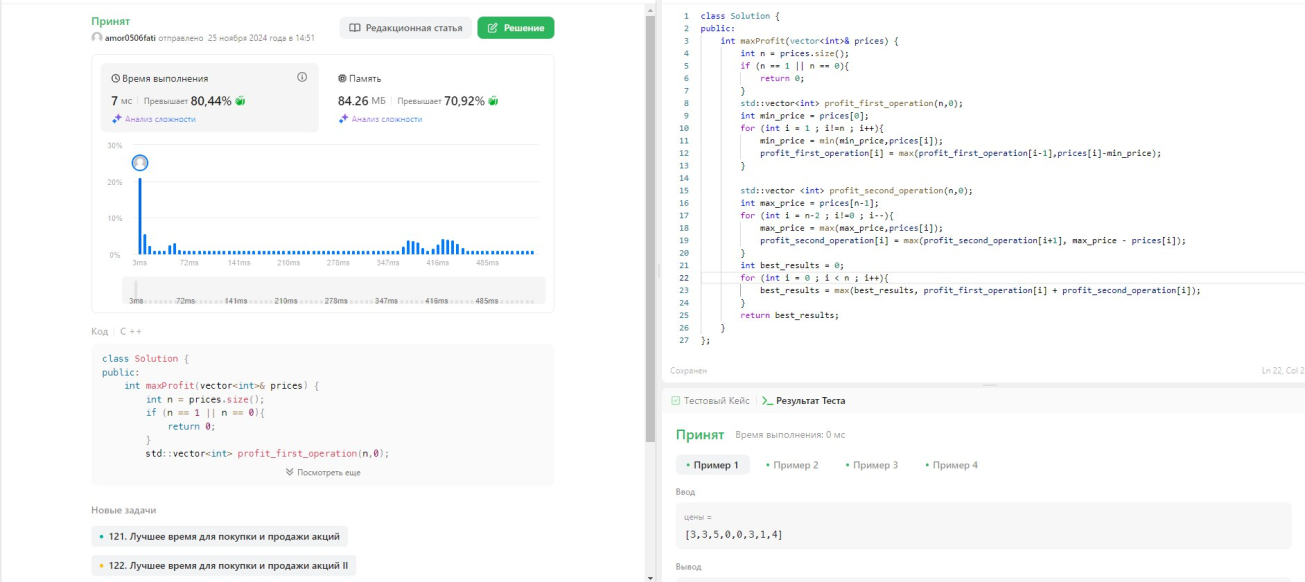


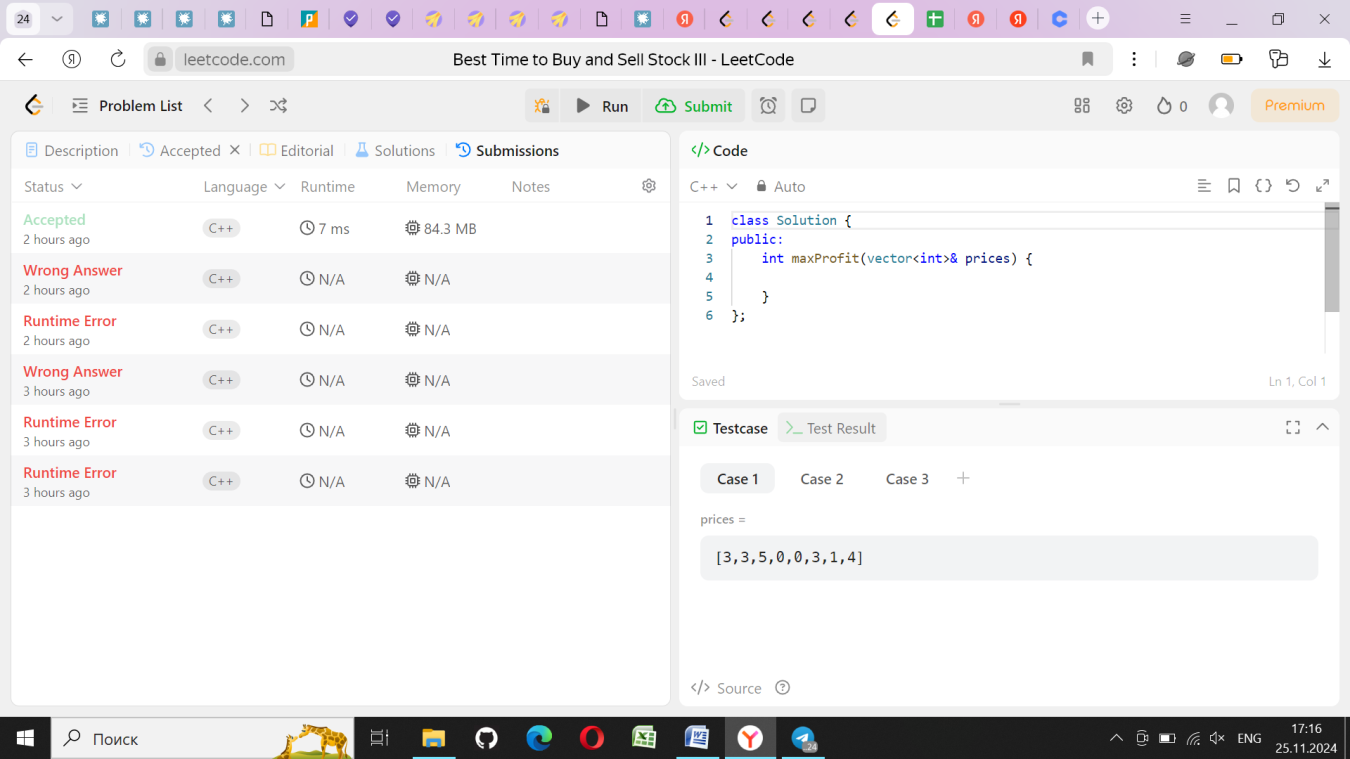
Подсчет памяти:



1. Заключение

Решение прошло все тесты памяти и времени выполнения.





Оптимизировать решение можно объеденив два первых цикла в один.

В этой задаче необходимо динамическое программирование , потому ,что если решать задачу без запоминания промежуточных решений вспомогательных задач , то мы будем слишком часто пересчитывать различные разности пар , если выбрать самый наивный способ то сложность может выйти от О(n\*(n-1)) (Тут мы бы потратили столько же памяти) до O((n^2\*(n-1)^2)/4), мы же смогли реализовать алгоритм за О(N).

1. Приложения

Задачу можно оптимизировать объеденив первый и второй циклы. Ассимптотика та же , но реальных операций примерно на N меньше. Код может быть тоже реализован по требованию.