ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Отчёт по лабораторной работе № 7

«Жадные алгоритмы

Выполнила работу

Андреева Наталия

Академическая группа №J3110

Принято

Вершинин Владислав Константинович

Санкт-Петербург

2024

**Структура отчёта:**

1. Введение

Цель работы: Разработать эффективное решение для задачи “Earliest Possible Day of Full Bloom” с использованием жадного алгоритма.

Задачи:

1. Проанализировать условие задачи и ограничения.
2. Реализовать алгоритм с использованием жадного подхода на языке C++.
3. Оптимизировать решение с учетом ограничений по памяти и времени выполнения.
4. Провести анализ временной и пространственной сложности данного алгоритма.
5. Реализация
6. Создание пар. Создаем вектор пар из значений plantTime и growTime.

int n = plantTime.size();

vector<pair<int, int>> seeds(n);

for (int i = 0; i < n; ++i) {

seeds[i] = {plantTime[i], growTime[i]};

}

Создаем вектор seeds, который содержит пары значений. Каждая пара состоит из времени посадки и времени роста для каждого семени.

1. Сортировка

sort(seeds.begin(), seeds.end(), [](const pair<int, int>& a, const pair<int, int>& b) {

return a.second > b.second;

});

Сортируем вектор по времени роста в порядке убывания.

1. Инициализация переменных

int t = 0; // Общее время посадки

int ans = 0; // Наименьший день цветения

1. Итерация по семенам

for (const auto& seed : seeds) {

t += seed.first; // Увеличиваем общее время посадки

ans = max(ans, t + seed.second); // Обновляем наименьший день цветения

}

Проходим по каждому семени и вычисляем день его цветения.

1. Экспериментальная часть

Оценка сложности:

- Временная сложность: O(nlogn) для сортировки и O(n) для итерации по вектору.

- Пространственная сложность: Используется дополнительная память для хранения пар семян O(n).

1. Заключение

В ходе лабораторной работы был реализован жадный алгоритм для определения наименьшего дня цветения всех семян цветов. Цель работы была достигнута путем тестирования на различных наборах данных и анализа производительности алгоритма.

Применение жадного алгоритма в решении данной задачи оказалось оправданным по нескольким причинам. Во-первых, жадный алгоритм основывается на выборе локально оптимальных решений на каждом шаге, что в данной задаче позволяет эффективно минимизировать общее время до цветения всех семян. Сортировка семян по времени их роста в порядке убывания позволяет начинать период роста семян с более длительным временем как можно раньше, что снижает общее время ожидания. Во-вторых, эффективность алгоритма доказывает и его временная сложность. Это делает алгоритм подходящим для обработки больших наборов данных.

В качестве дальнейших исследований можно рассмотреть возможность оптимизации алгоритма с точки зрения уменьшения затрат памяти или изучить параллельные версии алгоритма для обработки больших наборов данных.

1. Приложения

ПРИЛОЖЕНИЕ А

Листинг кода файла earliest\_possible\_day\_of\_full\_bloom.cpp

#include <vector>

#include <algorithm>

using namespace std;

class Solution {

public:

int earliestFullBloom(vector<int>& plantTime, vector<int>& growTime) {

int n = plantTime.size();

vector<pair<int, int>> seeds(n);

// Создаем пары (plantTime[i], growTime[i])

for (int i = 0; i < n; ++i) {

seeds[i] = {plantTime[i], growTime[i]};

}

// Сортируем по времени роста в порядке убывания

sort(seeds.begin(), seeds.end(), [](const pair<int, int>& a, const pair<int, int>& b) {

return a.second > b.second;

});

int t = 0; // Общее время посадки

int ans = 0; // Наименьший день цветения

for (const auto& seed : seeds) {

t += seed.first; // Увеличиваем общее время посадки

ans = max(ans, t + seed.second); // Обновляем наименьший день цветения

}

return ans;

}

};

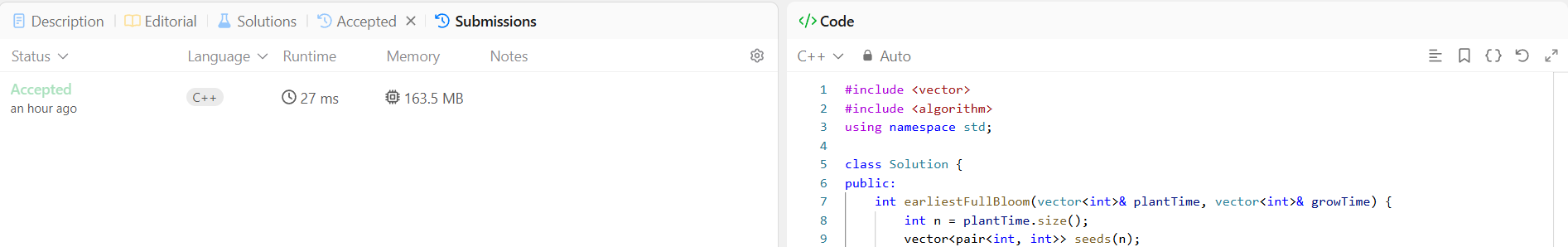
ПРИЛОЖЕНИЕ B

Рис 1. Задача прошла все тесты

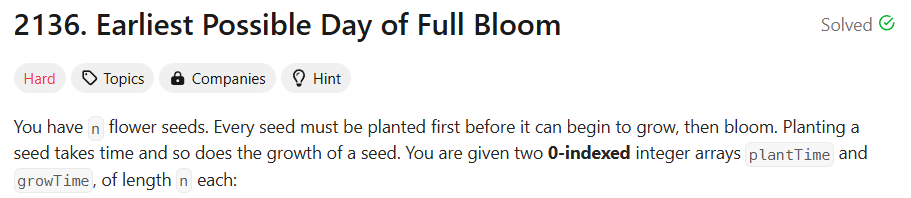
ПРИЛОЖЕНИЕ С

Рис 2. Сложность задачи