aLLIotaLP

Чемпионат по программированию: Бэкенд-разработка - Квалификация

3 ноя 2019, 17:51:44
старт: 2 ноя 2019, 20:02:59
финиш: 3 ноя 2019, 01:02:59
длительность: 05:00:00
начало: 14 окт 2019, 12:00:00
конец: 20 окт 2019, 23:59:59

D. Заказы по цепочке

Ограничение времени	1 секунда
Ограничение памяти	64Mb
Ввод	стандартный ввод или input.txt
Вывод	стандартный вывод или output.txt

Водитель выбирает заказы на день. Он может выбирать из $N \leq 1000$ заказов. Для каждого заказа известны время начала A_i и время окончания B_i, причём $0 \leq A_i < B_i \leq 20000$. Водитель также знает стоимость каждого заказа W_i, причём $1 \leq W_i \leq 10^5$. Нужно найти последовательность заказов с наибольшей суммарной стоимостью. В каждый момент времени водитель может выполнять только один заказ.

Формат ввода

В первой строчки задано целое положительное число N — количество заказов. Далее следует N строк. В i-й строке содержится 3 целых числа, разделённых пробелами: A_i, B_i и W_i.

Формат вывода

Требуется вывести одно число — наибольший заработок водителя.

Пример 1

Ввод	Вывод
2 0 1 2 2 3 8	10

Пример 2

Ввод	Вывод
3 1 3 15 3 7 25 2 6 30	40

Язык GNU c++17 7.3

Набрать здесь | Отправить файл

```cpp
#include <algorithm>
#include <vector>

#include <iostream>

struct PeriodPrice {
    uint16_t Start, End;
    uint32_t Price;
};

std::vector<PeriodPrice> ReadInput();
uint64_t Solve(std::vector<PeriodPrice>& periodPrices);

int main() {
    auto periodPrices = ReadInput();
    std::cout << Solve(periodPrices) << std::endl;
    return 0;
}

std::vector<PeriodPrice> ReadInput() {
    uint16_t n;
    std::cin >> n;

    std::vector<PeriodPrice> result;
    result.reserve(n);

    for (uint16_t i = 0; i < n; i++) {
        PeriodPrice periodPrice;
        std::cin >> periodPrice.Start >> periodPrice.End >> periodPrice.Price;
        result.push_back(periodPrice);
    }

    return result;
}

uint64_t Solve(std::vector<PeriodPrice>& periodPrices) {
    sort(periodPrices.begin(), periodPrices.end(), [](auto x, auto y) { return x.End < y.End; });
```

Отправить

Предыдущая