**Министерство науки и высшего образования Российской Федерации**

федеральное государственное автономное образовательное учреждение высшего образования

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»**

**Отчет**

по задачам №1494, №1604, №1726

по дисциплине «**Алгоритмам и структурам данных**»

Автор: Братушка Н. И.

Факультет: СУиР

Группа: R3238

Поток: АиСД 2.2

Преподаватель: Тропченко А. А.

Изображение выглядит как черный, темнота

Автоматически созданное описание

Санкт-Петербург 2024

Содержание

[Задача 1494. Монобильярд 3](#_Toc160200162)

[Текст задачи 3](#_Toc160200163)

[Исходные данные 3](#_Toc160200164)

[Результат 3](#_Toc160200165)

[Примеры 3](#_Toc160200166)

[Замечания 3](#_Toc160200167)

[Программа для решения задачи 4](#_Toc160200168)

[Результат на сайте Timus 4](#_Toc160200169)

[Задача 1604. В стране Дураков 5](#_Toc160200170)

[Текст задачи 5](#_Toc160200171)

[Исходные данные 5](#_Toc160200172)

[Результат 5](#_Toc160200173)

[Пример 5](#_Toc160200174)

[Программа для решения задачи 6](#_Toc160200175)

[Результат на сайте Timus 6](#_Toc160200176)

[Задача 1726. Стенка на стенку 6](#_Toc160200177)

[Текст задачи 6](#_Toc160200178)

[Исходные данные 7](#_Toc160200179)

[Результат 7](#_Toc160200180)

[Пример 7](#_Toc160200181)

[Программа для решения задачи 7](#_Toc160200182)

[Результат на сайте Timus 7](#_Toc160200183)

# **Задача 1494. Монобильярд**

## **Текст задачи**

Стол для монобильярда, установленный в игровом доме уездного города N, оказался очень прибыльным вложением. До того, как в городе появился небезызвестный господин Чичиков. Раз за разом он выигрывал, и хозяин, подсчитывая убытки, понимал, что дело тут нечисто. Однако уличить подлеца в жульничестве не удавалось до прибытия в город N ревизора из Петербурга.

Правила игры в монобильярд очень просты: нужно последовательно закатить в единственную лузу шары с номерами 1, 2, …, *N* (именно в этом порядке). Пока господин Чичиков играл, ревизор несколько раз подходил к столу и забирал из лузы последний закатившийся туда шар. В конце концов, оказалось, что Чичиков закатил в лузу все шары, а ревизор все шары достал и обследовал. Аферист утверждал, что закатил шары в правильном порядке. Хозяин понял, что это его шанс: ревизор должен помнить, в каком порядке он доставал шары. Однако так ли легко будет доказать жульничество?

### Исходные данные

В первой строке записано целое число *N* — количество бильярдных шаров (1 ≤ *N* ≤ 100000). В следующих *N* строках даны номера этих шаров в том порядке, в котором ревизор забирал их из лузы.

### Результат

Выведите слово «Cheater», если Чичиков не мог закатить все *N* шаров в правильном порядке. Иначе выведите «Not a proof».

### Примеры

|  |  |
| --- | --- |
| **исходные данные** | **результат** |
| 2  2  1 | Not a proof |
| 3  3  1  2 | Cheater |

### Замечания

В первом примере Чичиков мог закатить шары в правильном порядке, если ревизор достал их оба по очереди уже после того, как Чичиков закатил второй шар. Во втором примере Чичиков мог закатить шары в любом порядке, кроме правильного 1-2-3.

## **Программа для решения задачи**

Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Автоматически созданное описание

Мы создаем стек, в котором изначально лежит контейнер (вложенный массив) [0, 0], который нужен для того, чтобы не возникло ошибки с индексами при первой итерации цикла. В цикле мы считываем значение бильярдного шара и проверяем ряд условий:

* Если шар равен второму значению в любом контейнере, то мы уменьшаем это значение (из контейнера) на 1. Если окажется, что второе значение окажется меньше первого, то мы удаляем этот контейнер.
* Если шар больше второго значения последнего контейнера, то мы проверяем, что это значение больше переменной h, чем 1. Если ответ положителен, то мы создаем новый контейнер [h+1, значение-1]. Иначе мы пропускаем этот шаг и далее присваиваем переменной h считанное значение.
* Если первые 2 условия не выполнены, то мы выводим надпись ‘Cheater’ и завершаем программу.

Если программа выполняет все итерации, то в конце мы выводим надпись ‘Not a proof’.

Мы используем вложенные массивы для обозначения начала и конца интервала (последовательность шаров), который должен находиться в лузе. Если мы считываем число, которое равняется правой границе любого интервала, то мы уменьшаем значение границы на 1. Если правая граница при этом станет меньше левой, то мы должны будем удалить этот интервал, так как все шары из этого интервала закатились в лузу последовательно. Если же число будет меньше правой границы, то это означает, что игрок играет нечестно.

h – это переменная, содержащая значение для определения начального числа при создании нового интервала. Изначально её значение равняется 0, далее ей присваивается значение считанного шара. Если мы создаем новый контейнер, то очевидно, что в лузе не будет шара, которой был извлечён, но для выигрыша в ней должны находиться шары от (h+1) до (следующее извлеченное значение-1). Однако если считанное число больше h на 1, то мы получим контейнер [(h+1), h], который должен быть удален, поэтому нам не имеет никакого смысла добавлять его в стек.

## **Результат на сайте Timus**

****

# **Задача 1604. В стране Дураков**

## **Текст задачи**

Главный бульдог-полицейский Страны Дураков решил ввести ограничение скоростного режима на автомобильной трассе, ведущей от Поля Чудес к пруду Черепахи Тортиллы. Для этого он заказал у Папы Карло *n* знаков ограничения скорости. Папа Карло слабо разбирался в дорожном движении и поэтому изготовил знаки с разными ограничениями на скорость: 49 км/ч, 34 км/ч, 42 км/ч, и т.д. Всего получилось *k* различных ограничений: *n*1 знаков с одним ограничением, *n*2 знаков со вторым ограничением, и т.д. (*n*1 + … + *nk* = *n*)

Бульдог-полицейский ничуть не расстроился, получив такие знаки, напротив, он решил извлечь из этого экономическую выгоду. Дело в том, что по Правилам дорожного движения Страны Дураков ограничение на скорость действует вплоть до следующего знака. Если на знаке написано число 60, это означает, что участок от данного знака до следующего нужно проехать ровно со скоростью 60 километров в час — не больше и не меньше. Бульдог распорядился расставить знаки так, чтобы обогатившимся на Поле Чудес автолюбителям во время своего движения по трассе приходилось как можно больше раз менять скорость. Для этого нужно расставить имеющиеся знаки в правильном порядке. Если Вы поможете бульдогу это сделать, то он готов будет поделиться с Вами частью своих доходов.

### Исходные данные

В первой строке дано число *k* — количество различных типов знаков с ограничением скорости (1 ≤ *k* ≤ 10000). Во второй строке через пробел перечислены целые положительные числа *n*1, …, *nk*. Сумма всех *ni* не превосходит 10000.

### Результат

Выведите *n* целых чисел в пределах от 1 до *k* — порядок, в котором нужно расставить по трассе имеющиеся знаки. Вне зависимости от того, какой знак стоит первым, считается, что, проезжая его, водитель меняет скорость, так как до этого ограничения не действовали. Если задача имеет несколько решений, выведите любое.

### Пример

|  |  |
| --- | --- |
| **исходные данные** | **результат** |
| 2  2 2 | 1 2 1 2 |

**Автор задачи:**Александр Ипатов (идея — Александр Торопов)  
**Источник задачи:**Девятое открытое личное первенство УрГУ (1 марта 2008)

## **Программа для решения задачи**

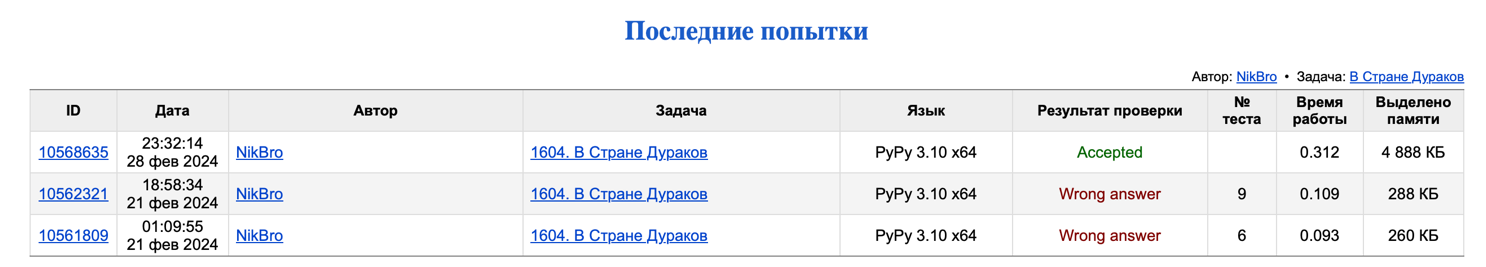
**Изображение выглядит как текст, снимок экрана, программное обеспечение, компьютер

Автоматически созданное описание**

Сначала мы считываем данные. Для расстановки знаков в нужном для нас порядке воспользуемся следующим алгоритмом:

* Находим тип знаков с наибольшим числом экземпляром. Считаем число промежутков (число экземпляров) и заполняем наш массив знаками этого типа.
* Далее берем второй по количеству экземпляров тип знаков и расставляем их в промежутки между изначальными знаками. Если у нас закончились знаки, то переходим к следующему типу. Когда мы заполнили все промежутки между изначальными знаками, мы начинаем сначала и заполняем пространство между расставленными в ходе предыдущей итерации знаками и изначальными.

## **Результат на сайте Timus**

****

# **Задача 1726. Кто ходит в гости…**

## **Текст задачи**

Программный комитет школьных соревнований по программированию, проходящих в УрГУ — многочисленная, весёлая и дружная команда. Дружная настолько, что общения в университете им явно не хватает, поэтому они часто ходят друг к другу в гости. Все ребята в программном комитете очень спортивные и ходят только пешком.

Однажды хранитель традиций олимпиадного движения УрГУ подумал, что на пешие прогулки от дома к дому члены программного комитета тратят слишком много времени, которое могли бы вместо этого потратить на придумывание и подготовку задач. Чтобы доказать это, он решил посчитать, какое расстояние в среднем преодолевают члены комитета, когда ходят друг к другу в гости. Хранитель традиций достал карту Екатеринбурга, нашёл на ней дома всех членов программного комитета и выписал их координаты. Но координат оказалось так много, что хранитель не смог справиться с этой задачей самостоятельно и попросил вас помочь ему.

Город Екатеринбург представляет собой прямоугольник со сторонами, ориентированными по сторонам света. Все улицы города идут строго с запада на восток или с севера на юг, проходя через весь город от края до края. Дома всех членов программного комитета расположены строго на пересечении каких-то двух перпендикулярных улиц. Известно, что все члены комитета ходят только по улицам, поскольку идти по тротуару гораздо приятнее, чем по дворовым тропинкам. И, конечно, при переходе от дома к дому они всегда выбирают кратчайший путь. Программный комитет очень дружный, и все его члены ходят в гости ко всем одинаково часто.

### Исходные данные

Первая строка содержит целое число *n* — количество членов программного комитета (2 ≤ *n* ≤ 105). В *i*-й из следующих *n* строк через пробел записаны целые числа *xi*, *yi* — координаты дома *i*-го члена программного комитета (1 ≤ *xi*, *yi* ≤ 106).

### Результат

Выведите среднее расстояние, которое проходит член программного комитета от своего дома до дома своего товарища, округлённое вниз до целых.

### Пример

|  |  |
| --- | --- |
| **исходные данные** | **результат** |
| 3  10 10  20 20  10 20 | 13 |

## **Программа для решения задачи**

**Изображение выглядит как текст, снимок экрана, программное обеспечение, компьютер

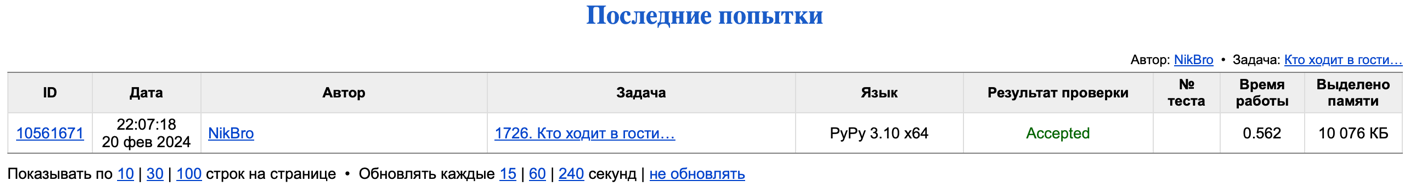
Автоматически созданное описание**

Для начала мы считываем число товарищей и затем координаты каждого дома в два массива – для абсцисс и ординат – и сортируем их. Далее необходимо найти дистанцию, которую проходят все товарищи вместе взятые. Для этого можно найти длину всех маршрутов (от дома А до дома Б, от А до В, от Б до В и т. д.), а затем умножить её на 2, так как по одному маршруту товарищи ходят друг к другу. Для вычисления дистанции мы из конечных координат вычисляем начальные, поэтому для удобства будем считать, что конечной точкой будут дома с большими координатами. Так как товарищи ходят по прямоугольному городу, то мы можем отдельно вычислить дистанцию, которую они пройдут вдоль осей OX и OY. Для вычисления дистанции мы будем вычислять расстояния от точки с наименьшими координатами (X и Y) до других. Приведем пример на основе примера, приведенного в условии:

Изображение выглядит как текст, рукописный текст, Шрифт, линия

Автоматически созданное описание

## **Результат на сайте Timus**

****