**МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**Ордена Трудового Красного Знамени**

**федеральное государственное бюджетное**

**образовательное учреждение высшего образования**

**«Московский технический университет связи и информатики»**

Кафедра «Математическая кибернетика и информационные технологии»

Курсовая работа по дисциплине

«Структуры и алгоритмы обработки данных»

Вариант №8

Выполнила: студентка группы БФИ2001

Калмыкова Д.С.

Проверил: Симонов С.Е.

Москва, 2022

**Содержание**

[Цель курсовой работы: 2](#_Toc103375245)

[Задание №1 3](#_Toc103375246)

[Задание №2 5](#_Toc103375247)

[Задание №3 7](#_Toc103375248)

[Задание №4 9](#_Toc103375249)

[Задание №5 11](#_Toc103375250)

[Задание №6 14](#_Toc103375251)

[Задание №7 17](#_Toc103375252)

[Задание №8 20](#_Toc103375253)

[Задание №9 21](#_Toc103375254)

[Задание №10 23](#_Toc103375255)

[Вывод 25](#_Toc103375256)

[Приложение А 26](#_Toc103375257)

[Листинг А.1 26](#_Toc103375258)

[Листинг А.2 27](#_Toc103375259)

[Листинг А.3 27](#_Toc103375260)

[Листинг А.4 28](#_Toc103375261)

[Листинг А.5 30](#_Toc103375262)

[Листинг А.6 31](#_Toc103375263)

[Листинг А.7 32](#_Toc103375264)

[Листинг А.8 33](#_Toc103375265)

[Листинг А.9 34](#_Toc103375266)

[Листинг А.10 35](#_Toc103375267)

[Листинг А.11 36](#_Toc103375268)

**Цель курсовой работы:** разработать алгоритмы для решения 10 задач в соответствии с заданным вариантом.

# **Задание №1**

Юный тренер покемонов Сергей Б. обнаружил большой дом, состоящий из n квартир, выстроенных в ряд слева направо, причем в каждую квартиру можно зайти с улицы, из каждой квартиры можно выйти на улицу, а также из каждой квартиры можно зайти в соседнюю слева или справа квартиру. Из квартиры номер 1 можно попасть только в квартиру номер 2, а из квартиры номер n можно попасть только в квартиру номер n - 1.

Так сложилось, что в каждой из этих квартир находится ровно один покемон какого-то типа. Возможно, что в каких-то квартирах находятся покемоны одинаковых типов. Сергей Б. попросил жителей дома пустить его к ним в квартиры, чтобы поймать покемонов. Посовещавшись, жители дома решили, что разрешат Сергею Б. зайти с улицы ровно в одну квартиру, посетить несколько квартир, а затем выйти из какой-то квартиры на улицу, причем они не разрешили Сергею Б. посещать никакую квартиру более одного раза.

Сергей Б. очень обрадовался, но как человек интеллигентный, решил посетить как можно меньше квартир, но при этом, естественно, собрать покемонов всех типов, которые есть в доме (иначе зачем тогда это всё). Перед вами стоит задача помочь ему и определить это минимальное число квартир.

**Входные данные**

В первой строке следует целое число n (1 ≤ n ≤ 100 000) — количество квартир в доме.

Во второй строке следует строка s длины n, состоящая из строчных и прописных букв английского алфавита, причем i-я буква равна типу покемона, который находится в квартире номер i.

**Выходные данные**

Выведите минимальное число квартир, которые должен посетить Сергей Б., чтобы поймать покемонов всех типов, которые есть в доме.

**Примеры**  
входные данные  
3  
AaA  
выходные данные  
2  
входные данные  
7  
bcAAcbc  
выходные данные  
3  
входные данные  
6  
aaBCCe  
выходные данные  
5

**Примечание**

В первом тестовом примере Сергей Б. может, например, начать с квартиры 1 и закончить в квартире 2.

Во втором тестовом примере Сергей Б. может, например, начать с квартиры 4 и закончить в квартире 6.

В третьем тестовом примере Сергей Б. должен начать с квартиры 2 и закончить в квартире 6.

Код для решения задачи представлен в Листинге А.2. Результат работы программы представлен на Рисунке 1:

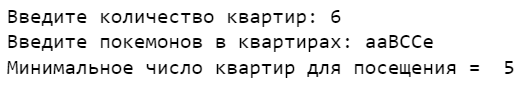


Рисунок 1 – Результат работы программы

# **Задание №2**

Массив 𝑎1,𝑎2,…,𝑎𝑛 называется хорошим, если и только если для любого подотрезка 1≤𝑙≤𝑟≤𝑛, выполняется следующее: 𝑎𝑙+𝑎𝑙+1+…+𝑎𝑟=0.5(𝑎𝑙+𝑎𝑟)⋅(𝑟−𝑙+1).

Вам дан массив целых чисел 𝑎1,𝑎2,…,𝑎𝑛. За одну операцию вы можете заменить один любой элемент массива на любое вещественное число. Определите минимальное число операций, необходимое, чтобы сделать массив хорошим.

**Входные данные**

Первая строка содержит одно целое число 𝑡 (1≤𝑡≤100): количество наборов входных данных. Далее следуют описания 𝑡 наборов входных данных, по две строки на набор.

Первая строка содержит одно целое число 𝑛 (1≤𝑛≤70): количество чисел в массиве.

Вторая строка содержит 𝑛 целых чисел 𝑎1,𝑎2,…,𝑎𝑛 (−100≤𝑎𝑖≤100): изначальный массив.

**Выходные данные**

Для каждого набора входных данных выведите одно целое число: минимальное число элементов, которое нужно заменить, чтобы массив стал хорошим.

**Примеры**  
входные данные 5  
4  
1 2 3 4  
4  
1 1 2 2  
2  
0 -1  
6  
3 -2 4 -1 -4 0  
1  
-100  
выходные данные  
0  
2  
0  
3  
0

**Примечание**  
В первом примере массив хороший изначально.

Во втором примере один из возможных хороших массивов — [1,1,1⎯⎯,1⎯⎯] (замененные элементы подчеркнуты).

В третьем примере массив хороший изначальное.

Во четвертом примере один из возможных хороших массивов — [−2.5⎯⎯⎯⎯⎯⎯⎯⎯,−2,−1.5⎯⎯⎯⎯⎯⎯⎯⎯,−1,−0.5⎯⎯⎯⎯⎯⎯⎯⎯,0].

Код для решения задачи представлен в Листинге А.3. Результат работы программы представлен на Рисунке 2:

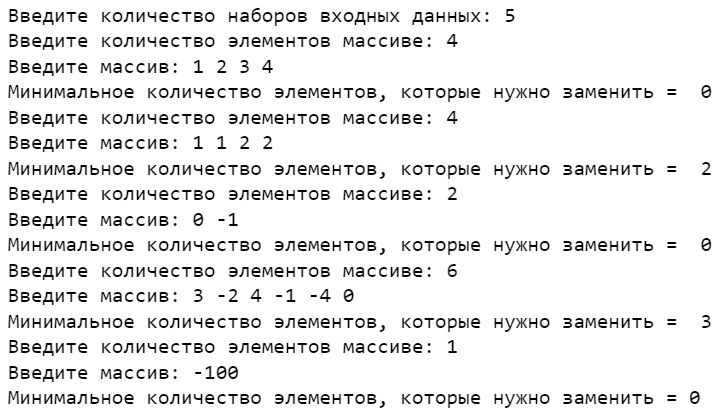


Рисунок 2 – Результат работы программы

# **Задание №3**

Чтобы удовлетворить свою любовь к собиранию носков в пары, Феникс принес свои 𝑛 носков (𝑛 — четное) в магазин по продаже носков. Каждый его носок определенного цвета 𝑐𝑖 и либо левый, либо правый.

Феникс может заплатить один доллар магазину, чтобы:

либо перекрасить какой-то носок в любой цвет 𝑐′ (1≤𝑐′≤𝑛), либо превратить левый носок в правый, либо превратить правый носок в левый. Магазин может производить любое из изменений любое количество раз. Заметим, что цвет левого носка не меняется, когда он превращается в правый, и наоборот.

Два носка образуют пару, если это левый и правый носок одинакового цвета. Какую минимальное количество денег нужно заплатить Фениксу, чтобы собрать 𝑛/2 пар? Каждый носок должен попасть ровно в одну пару.

**Входные данные**

Входные данные состоят из нескольких наборов. В первой строке задано одно целое число 𝑡 (1≤𝑡≤1000) — количество наборов входных данных.

В первой строке каждого набора заданы три целых числа 𝑛, 𝑙 и 𝑟 (2≤𝑛≤2⋅105; 𝑛 — четное; 0≤𝑙,𝑟≤𝑛; 𝑙+𝑟=𝑛) — общее количество носков и количество левых и правых носков соответственно.

В следующей строке заданы 𝑛 целых чисел 𝑐𝑖 (1≤𝑐𝑖≤𝑛) — цвета носков. Первые 𝑙 носков — левые, остальные 𝑟 носков — правые.

Гарантируется, что сумма 𝑛 по всем наборам входных данных не превосходит 2⋅105.

**Выходные данные**

Для каждого набора входных данных, выведите одно число — минимальное количество денег, которое нужно заплатить Фениксу, чтобы собрать 𝑛/2 пар носков. Каждый носок должен попасть ровно в одну пару.

**Примеры**  
входные данные  
4  
6 3 3  
1 2 3 2 2 2  
6 2 4  
1 1 2 2 2 2  
6 5 1  
6 5 4 3 2 1  
4 0 4  
4 4 4 3  
выходные данные  
2  
3  
5  
3

**Примечание**  
В первом наборе, Феникс может заплатить 2 доллара для того, чтобы:

перекрасить носок 1 в цвет 2,  
перекрасить носок 3 в цвет 2.  
В результате он получит 3 пары. Например, пары (1,4), (2,5) и (3,6).

Во втором наборе, Феникс может заплатить 3 доллара, чтобы:

превратить носок 6 из правого в левый,  
перекрасить носок 3 в цвет 1,  
перекрасить носок 4 в цвет 1.  
В результате он получит 3 пары. Например, пары (1,3), (2,4) и (5,6).

Код для решения задачи представлен в Листинге А.4. Результат работы программы представлен на Рисунке 3:

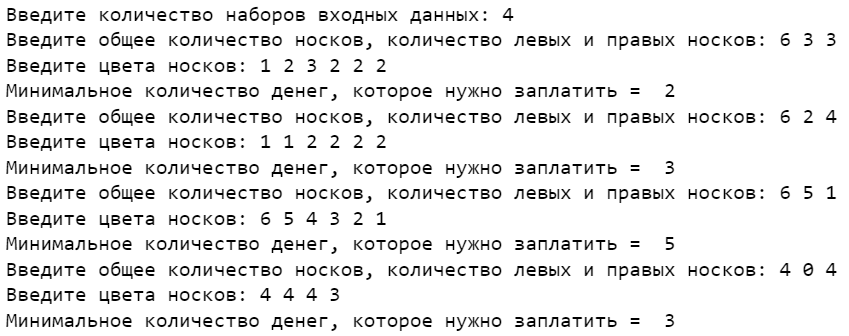


Рисунок 3 – Результат работы программы

# **Задание №4**

У Монокарпа есть две строки 𝑠 и 𝑡 одинаковой длины, состоящие из латинских букв «a» и «b».

Монокарп хочет сделать строки 𝑠 и 𝑡 одинаковыми. Для этого он может выполнять следующие операции: выбрать в строке 𝑠 позицию 𝑝𝑜𝑠1, выбрать в строке 𝑡 позицию 𝑝𝑜𝑠2 и поменять местами буквы 𝑠𝑝𝑜𝑠1 и 𝑡𝑝𝑜𝑠2.

Перед вами стоит задача определить минимальное количество операций, которые должен сделать Монокарп, чтобы сделать строки 𝑠 и 𝑡 одинаковыми, а также вывести сами операции. Если невозможно сделать строки одинаковыми, сообщите об этом.

**Входные данные**  
В первой строке следует целое число 𝑛 (1≤𝑛≤2⋅105) — длина строк 𝑠 и 𝑡.

Во второй строке следует строка 𝑠 длины 𝑛, состоящая из латинских букв «a» и «b».

В третьей строке следует строка 𝑡 длины 𝑛, состоящая из латинских букв «a» и «b».

**Выходные данные**

Если невозможно сделать строки одинаковыми, выведите −1.

В противном случае, в первую строку выведите целое число 𝑘 — минимальное количество операций, которые нужно сделать, чтобы сделать строки одинаковыми. В каждой из следующих 𝑘 строк выведите по два целых числа — позицию в строке 𝑠 и позицию в строке 𝑡, которые нужно использовать для обмена букв во время очередной операции.

**Примеры**входные данные  
4  
abab  
aabb  
выходные данные  
2  
3 3  
3 2  
входные данные  
1  
a  
b  
выходные данные  
-1  
входные данные  
8  
babbaabb  
abababaa  
выходные данные  
3  
2 6  
1 3  
7 8

**Примечание**

В первом примере достаточно двух операций обмена. Например, можно сначала поменять местами третью букву в строке 𝑠 с третьей буквой в строке 𝑡. После этого 𝑠= «abbb», 𝑡= «aaab». Затем нужно поменять третью букву в строке 𝑠 со второй буквой в строке 𝑡. После этого обе строки 𝑠 и 𝑡 будут равны «abab».

Во втором примере невозможно сделать строки одинаковыми.

Код для решения задачи представлен в Листинге А.5. Результат работы программы представлен на Рисунке 4:

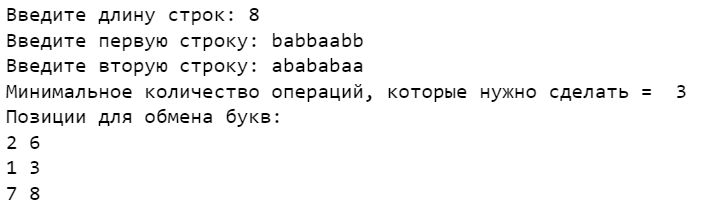


Рисунок 4 – Результат работы программы

# **Задание №5**

Рон — счастливый обладатель перестановки 𝑎 длины 𝑛.

Перестановкой длины 𝑛 является массив, состоящий из 𝑛 различных целых чисел от 1 до 𝑛 в произвольном порядке. Например, [2,3,1,5,4] — перестановка, но [1,2,2] — не перестановка (2 встречается в массиве дважды) и [1,3,4] тоже не перестановка (𝑛=3, но в массиве встречается 4).

Над перестановкой Рона ставится 𝑚 экспериментов следующего вида: (𝑟𝑖, 𝑝𝑖). Это обозначает, что элементы в диапазоне [1,𝑟𝑖] (другими словами, префикс длины 𝑟𝑖) будут отсортированы в возрастающем порядке с вероятностью 𝑝𝑖. Все эксперименты проводятся в том же порядке, в котором задаются во входных данных.

Для примера рассмотрим перестановку [4,2,1,5,3] и эксперимент (3,0.6). После такого эксперимента с вероятностью 60% перестановка примет вид [1,2,4,5,3], а с вероятностью 40% останется без изменений.

Вам требуется определить, с какой вероятностью перестановка станет полностью отсортированной по возрастанию после 𝑚 экспериментов.

**Входные данные**  
Каждый тест содержит один или несколько наборов входных данных. В первой строке записано количество наборов входных данных 𝑡 (1≤𝑡≤100).

Первая строка каждого набора входных данных содержит два целых числа 𝑛 и 𝑚 (1≤𝑛,𝑚≤105) — количество элементов в перестановке и количество экспериментов.

Вторая строка каждого набора входных данных содержит 𝑛 целых чисел 𝑎1,𝑎2,…,𝑎𝑛 (1≤𝑎𝑖≤𝑛) — описание перестановки.

Каждая из следующих 𝑚 входных данных содержат целое число 𝑟𝑖 и вещественное число 𝑝𝑖 (1≤𝑟𝑖≤𝑛,0≤𝑝𝑖≤1) — длина префикса массива и вероятность, с которой он отсортируется. Все вероятности даны с точностью не более 6 знаков после запятой.

Гарантируется, что сумма 𝑛 и сумма 𝑚 не превосходят 105 (∑𝑛,∑𝑚≤105).

**Выходные данные**  
Для каждого набора входных данных выведите единственное число — вероятность, что после всех экспериментов перестановка окажется отсортированной. Ваш ответ будет считаться правильным, если его абсолютная или относительная ошибка не превосходит 10−6.

Формально, пусть ваш ответ равен 𝑎, а ответ жюри равен 𝑏. Ваш ответ будет зачтен, если и только если |𝑎−𝑏|max(1,|𝑏|)≤10−6.

**Примеры**  
входные данные  
4  
4 3  
4 3 2 1  
1 0.3  
3 1  
4 0.6  
5 3  
4 2 1 3 5  
3 0.8  
4 0.6  
5 0.3  
6 5  
1 3 2 4 5 6  
4 0.9  
5 0.3  
2 0.4  
6 0.7  
3 0.5  
4 2  
1 2 3 4  
2 0.5  
4 0.1  
выходные данные  
0.600000  
0.720000  
0.989500  
1.000000

**Примечание**Для первого набора входных данных можно показать, что только от того, выполнится ли сортировка с помощью правила (4,0.6), зависит, будет ли итоговая перестановка отсортированной.

Код для решения задачи представлен в Листинге А.6. Результат работы программы представлен на Рисунке 5:

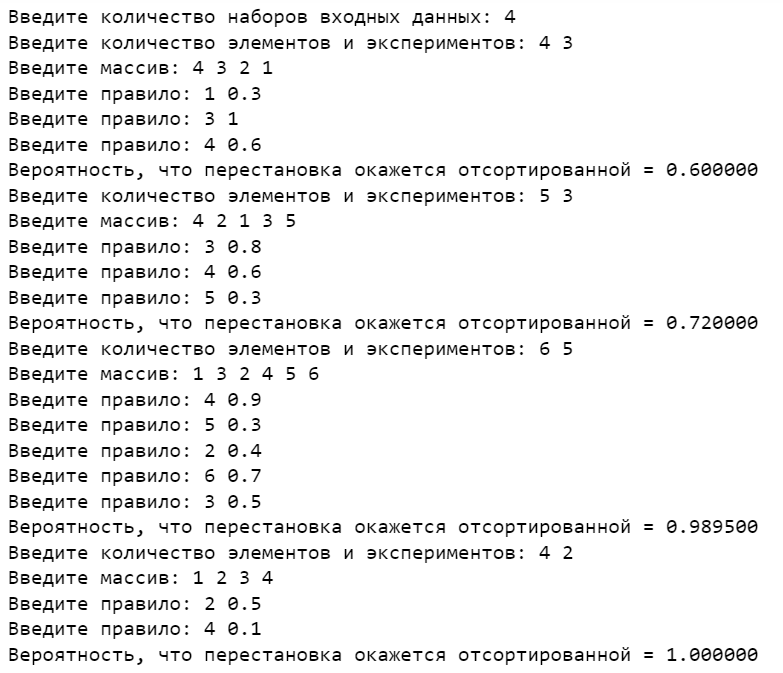


Рисунок 5 – Результат работы программы

# **Задание №6**

Вот и подошел к концу Beautiful Regional Contest (BeRC)! В соревновании приняли участие 𝑛 студентов. Уже известна таблица финальных результатов — участник на 𝑖-м месте решил 𝑝𝑖 задач. Так как в первую очередь участники упорядочиваются по количеству решенных задач, то 𝑝1≥𝑝2≥⋯≥𝑝𝑛.

Помогите жюри распределить золотые, серебряные и бронзовые медали. Пусть их количества равны 𝑔, 𝑠 и 𝑏, соответственно. Вот перечень требований из регламента, которые все должны быть удовлетворены:

для каждого из трёх достоинств медалей хотя бы одна медаль должна быть вручена (то есть 𝑔>0, 𝑠>0 и 𝑏>0); количество золотых медалей должно быть строго меньше и количества серебряных и количества бронзовых (то есть 𝑔<𝑠 и 𝑔<𝑏, но требований между 𝑠 и 𝑏 нет); каждый золотой медалист должен решить строго больше задач, чем любой награждённый серебряной медалью; каждый серебряный медалист должен решить строго больше задач, чем любой награждённый бронзовой медалью; каждый бронзовый медалист должен решить строго больше задач, чем любой не награждённый медалью участник; суммарное количество медалистов 𝑔+𝑠+𝑏 не должно превышать половину всех участников (например, если 𝑛=21, то можно наградить максимум 10 участников, а если 𝑛=26, то можно наградить максимум 13 участников). Жюри хочет наградить медалями суммарно наибольшее количество 𝑔+𝑠+𝑏 участников так, чтобы все перечисленные пункты выполнялись. Помогите жюри найти такой способ награждения медалями.

**Входные данные**  
В первой строке записано целое число 𝑡 (1≤𝑡≤10000) — количество наборов входных данных в тесте. Далее следуют 𝑡 наборов входных данных.

Первая строка набора входных данных содержит целое число 𝑛 (1≤𝑛≤4⋅105) — количество участников BeRC. Вторая строка набора содержит последовательность целых чисел 𝑝1,𝑝2,…,𝑝𝑛 (0≤𝑝𝑖≤106), где 𝑝𝑖 равно количеству задач, решенных 𝑖-м участником из таблицы финальных результатов. Значения 𝑝𝑖 отсортированы по невозрастанию, то есть 𝑝1≥𝑝2≥⋯≥𝑝𝑛.

Сумма значений 𝑛 по всем наборам входных данных в тесте не превосходит 4⋅105.

**Выходные данные**  
Выведите 𝑡 строк, 𝑗-я строка должна содержать ответ на 𝑗-й набор входных данных.

Ответ состоит из трёх неотрицательных целых чисел 𝑔,𝑠,𝑏.

Выведите 𝑔=𝑠=𝑏=0, если не существует способа наградить участников медалями, чтобы одновременно выполнялись все требования из условия. В противном случае выведите три положительных числа 𝑔,𝑠,𝑏 — возможное количество золотых, серебряных и бронзовых медалей, соответственно. Сумма 𝑔+𝑠+𝑏 должна быть максимально возможной. Если ответов несколько, то выведите любой из них.

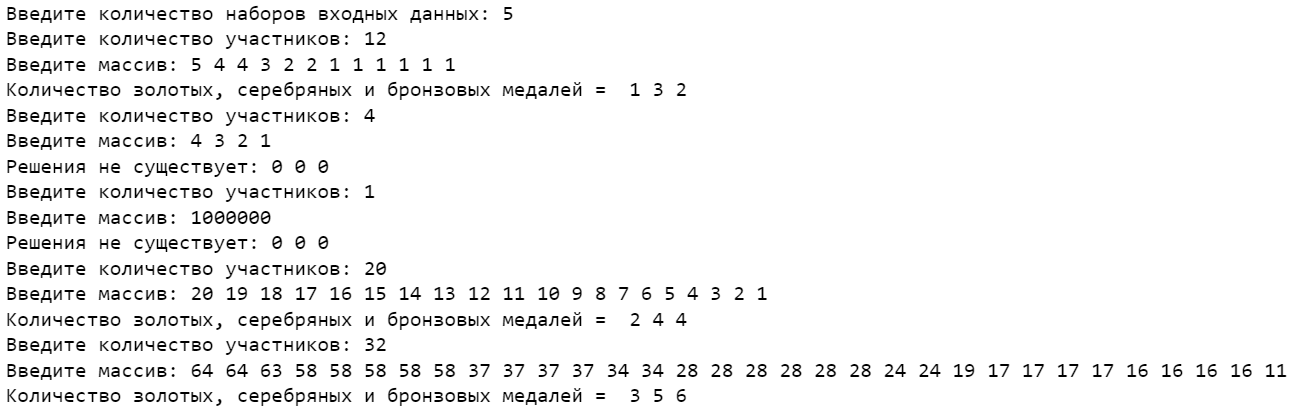
**Примеры**  
входные данные  
5  
12  
5 4 4 3 2 2 1 1 1 1 1 1  
4  
4 3 2 1  
1  
1000000  
20  
20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1  
32  
64 64 63 58 58 58 58 58 37 37 37 37 34 34 28 28 28 28 28 28 24 24 19 17 17 17 17 16 16 16 16 11  
выходные данные  
1 2 3  
0 0 0  
0 0 0  
2 5 3  
2 6 6

**Примечание**

В первом тестовом случае можно наградить 1 золотой, 2 серебряными и 3 бронзовыми медалями участников. Тогда золотую медаль получит участник, решившая 5 задач, серебрянные медали получат участники, решившие 4 задачи, бронзовые медали получат участники решившие 2 или 3 задачи. Участники, решившие 1 задачу медаль не получат. Заметим, что все условия выполнены и раздать медали таким способом можно. Больше, чем 6 медалей выдать невозможно, потому что нельзя выдать больше чем половину от количества участников медалей. Ответ 1, 3, 2 также является верным в этом тестовом случае.

Во втором и третьем тестовых случаев нельзя так раздать медали, потому что медаль каждого достоинства должна получить хотя бы один участник, но количество выданных медалей не должно превышать половину от количества участников.

Код для решения задачи представлен в Листинге А.7. Результат работы программы представлен на Рисунке 6:

Рисунок 6 – Результат работы программы

# **Задание №7**

𝑛 человек собрались, чтобы провести заседание жюри предстоящего соревнования, 𝑖-й член жюри придумал 𝑎𝑖 задач, которыми он хочет поделиться с другими.

Сначала жюри выбирает порядок, которому они будут следовать при обсуждении задач. Пусть это будет перестановка 𝑝 чисел от 1 до 𝑛 (массив размера 𝑛, в котором каждое число от 1 до 𝑛 встречается ровно один раз).

Затем обсуждение происходит следующим образом:

если у члена жюри 𝑝1 остались задачи, которые нужно рассказать, то он рассказывает одну задачу. В противном случае он будет пропущен. если у члена жюри 𝑝2 остались задачи, которые нужно рассказать, то он рассказывает одну задачу. В противном случае он будет пропущен. ... если у члена жюри 𝑝𝑛 остались задачи, которые нужно рассказать, то он рассказывает одну задачу. В противном случае он будет пропущен. если остались члены жюри, которые еще не рассказали все свои задачи, то процесс повторяется с самого начала. В противном случае обсуждение заканчивается.

Перестановка 𝑝 хорошая, если никто из членов жюри не будет рассказывать две или более задач подряд.

Подсчитайте количество хороших перестановок. Ответ может быть очень большим, поэтому выведите его по модулю

**Входные данные**

Первая строка содержит одно целое число 𝑡 (1≤𝑡≤104) — количество наборов входных данных.

Первая строка набора входных данных содержит одно целое число 𝑛 (2≤𝑛≤2⋅105) — количество членов жюри.

Вторая строка содержит 𝑛 целых чисел 𝑎1,𝑎2,…,𝑎𝑛 (1≤𝑎𝑖≤109) — количество задач, которые придумал 𝑖-й член жюри.

Сумма 𝑛 по всем наборам входных данных не превосходит 2⋅105.

**Выходные данные**  
Для каждого набора входных данных выведите одно целое число — количество хороших перестановок, взятое по модулю 998244353.

**Примеры**  
входные данные  
4  
2  
1 2  
3  
5 5 5  
4  
1 3 3 7  
6  
3 4 2 1 3 3  
выходные данные  
1  
6  
0  
540

**Примечание**  
Объяснение первого примера:

Существуют две возможные перестановки, 𝑝=[1,2] и 𝑝=[2,1]. Для 𝑝=[1,2] процесс происходит следующим образом:

первый член жюри рассказывает задачу;  
второй член жюри рассказывает задачу;

у первого члена жюри не осталось задач, которые нужно было бы рассказать, поэтому он будет пропущен; второй член жюри рассказывает задачу. Второй член жюри рассказал две задачи подряд, поэтому перестановка не является хорошей.

Для 𝑝=[2,1] процесс происходит следующим образом:

второй член жюри рассказывает задачу;

первый член жюри рассказывает задачу;

второй член жюри рассказывает задачу. Эта перестановка является хорошей.

Код для решения задачи представлен в Листинге А.8. Результат работы программы представлен на Рисунке 7:

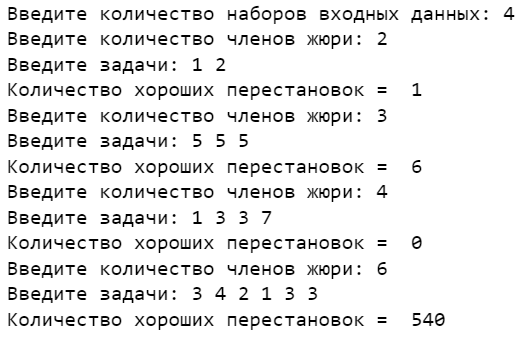


Рисунок 7 – Результат работы программы

# **Задание №8**

У Аюба был массив целых чисел 𝑎 длины 𝑛 и у этого массива было два необычных свойства:

Все целые числа в этом массиве были от 𝑙 до 𝑟 (включительно). Сумма всех элементов в массиве делилась на 3. К сожалению, Аюб потерял свой массив, но он помнит размер массива 𝑛, а также числа 𝑙 и 𝑟, так что он попросил вас посчитать количество способов восстановить массив по этим данным.

Так как ответ может быть очень большим, выведите его по модулю 109+7 (то есть остаток при делении на 109+7). В случае если не существует ни одного подходящего массива (Аюб что-то перепутал) — выведите 0.

**Входные данные**  
Первая и единственная строка содержит три целых числа 𝑛, 𝑙 и 𝑟 (1≤𝑛≤2⋅105,1≤𝑙≤𝑟≤109) — размер массива и диапазон чисел в нём.

**Выходные данные**  
Выведите остаток при делении на 109+7 количества способов восстановить массив.

**Примеры**  
входные данные  
2 1 3  
выходные данные  
3  
входные данные  
3 2 2  
выходные данные  
1  
входные данные  
9 9 99  
выходные данные  
711426616

**Примечание**

В первом примере возможны массивы [1,2], [2,1], [3,3].

Во втором примере единственный возможный массив — [2,2,2].

Код для решения задачи представлен в Листинге А.9. Результат работы программы представлен на Рисунке 8:

C:\Users\User\Desktop\учеба\4 сем\скрины\8.PNG

Рисунок 8 – Результат работы программы

# **Задание №9**

Задано два массива 𝑎 и 𝑏, каждый состоит из 𝑛 целых чисел.

Вы хотите создать новый массив 𝑐 следующим образом: выбрать какое-то вещественное (то есть не обязательно целое) число 𝑑, и для каждого 𝑖∈[1,𝑛] присвоить 𝑐𝑖:=𝑑⋅𝑎𝑖+𝑏𝑖.

Ваша задача — максимизировать количество нулей в массиве 𝑐. Чему равен максимально возможный ответ, если вы выберете значение 𝑑 оптимально?

**Входные данные**  
Первая строка входных данных содержит одно целое число 𝑛 (1≤𝑛≤2⋅105) — количество элементов в обоих массивах.

Вторая строка входных данных содержит 𝑛 целых чисел 𝑎1, 𝑎2, ..., 𝑎𝑛 (−109≤𝑎𝑖≤109).

Третья строка входных данных содержит 𝑛 целых чисел 𝑏1, 𝑏2, ..., 𝑏𝑛 (−109≤𝑏𝑖≤109).

**Выходные данные**

Выведите одно целое число — максимальное количество нулей в массиве 𝑐, если вы выберете 𝑑 оптимально.

**Примеры**  
входные данные  
5  
1 2 3 4 5  
2 4 7 11 3  
выходные данные  
2  
входные данные  
3  
13 37 39  
1 2 3  
выходные данные  
2  
входные данные  
0 0 0 0  
1 2 3 4  
выходные данные  
0  
входные данные  
3  
1 2 -1  
-6 -12 6  
выходные данные  
3

**Примечание**

В первом тестовом примере мы можем выбрать 𝑑=−2.

Во втором тестовом примере мы можем выбрать 𝑑=−113.

В третьем тестовом примере мы не можем получить ни одного нуля в массиве 𝑐, какое бы значение 𝑑 мы ни выбрали.

В четвертом тестовом примере мы можем выбрать 𝑑=6.

Код для решения задачи представлен в Листинге А.10. Результат работы программы представлен на Рисунке 9:

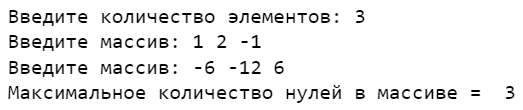


Рисунок 9 – Результат работы программы

# **Задание №10**

Число называется сильным, если оно является степенью двойки или факториалом. Другими словами, число 𝑚 сильное, если существует неотрицательное целое число 𝑑 такое, что 𝑚=2𝑑 или 𝑚=𝑑!, где 𝑑!=1⋅2⋅…⋅𝑑 (в частности, 0!=1). Например, 1, 4 и 6 являются сильными, потому что 1=1!, 4=22, 6=3!, а 7, 10 или 18 не являются.

Вам дано положительное число 𝑛. Найдите минимальное число 𝑘 такое, что 𝑛 можно представить в виде суммы 𝑘 различных сильных чисел, или определите, что такого 𝑘 не существует.

**Входные данные**  
Во входных данных находятся несколько наборов входных данных. В первой строке находится одно целое число 𝑡 (1≤𝑡≤100) — количество наборов входных данных. Далее следуют наборы входных данных.

Описание одного набора входных данных содержит одно целое число 𝑛 (1≤𝑛≤1012).

**Выходные данные**  
Для каждого набора входных данных выведите ответ на отдельной строке.

Если 𝑛 не может быть представлено как сумма различных сильных чисел, выведите −1.

Иначе выведите одно положительное число — минимально возможное число 𝑘.

**Примеры**  
входные данные  
4  
7  
11  
240  
17179869184  
выходные данные  
2  
3  
4  
1

**Примечание**

В первом примере 7 может быть представлена как 7=1+6, где 1 и 6 являются сильными. Так как 7 не является сильным числом, мы знаем, что минимальное значение 𝑘 в этом случае 𝑘=2.

Во втором примере один из способов разложить 11 в сумму трех сильных чисел такой: 11=1+4+6. Можно показать, что нельзя представить 11 в виде суммы двух или менее сильных чисел.

В третьем примере 240 может быть представлено как 240=24+32+64+120. Обратите внимание, что 240=120+120 не является корректным представлением, т. к. сильные числа должны быть различны.

В четвертом примере 17179869184=234, поэтому 17179869184 является сильным числом, и минимальное 𝑘 в этом случае 𝑘=1.

Код для решения задачи представлен в Листинге А.11. Результат работы программы представлен на Рисунке 10:

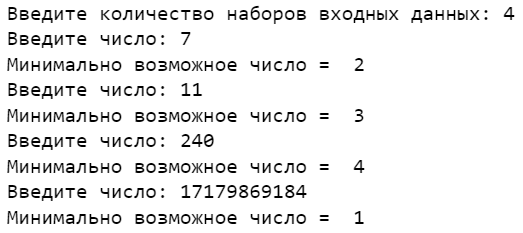


Рисунок 10 – Результат работы программы

**Вывод:** в ходе выполнения курсовой работы были разработаны алгоритмы для решения 10 задач.

# **Приложение А**

## Листинг А.1

**-** Код используемых библиотек и функций проверок

import sys

from operator import contains

# функция для проверки вхождения элемента в диапазон чисел

def check\_range(n, a, b):

if not a <= n <= b:

return False

return True

# функция для проверки равенства количества элементов массива заданному значению

def check\_equal(n, arr):

if not n == len(arr):

return False

return True

# функция для проверки латинских прописных и строчных букв

def check\_ascii(arr):

for i in range(len(arr)):

if not (65<=ord(arr[i])<=90 or 97<=ord(arr[i])<=122):

return False

return True

# функция для проверки латинских строчных букв 'a' и 'b'

def check\_ascii\_limited(arr):

for i in range(len(arr)):

if not 97<=ord(arr[i])<=98:

return False

return True

# функция для проверки нахождения каждого элемента массива в диапазоне чисел

def check\_arr(arr, a, b):

for elem in arr:

if not a <= elem <= b:

return False

return True

# функция для проверки является ли число вещественным и принадлежит диапазону [0,1]

def check\_float(num):

if abs(1-num) <= 1:

return True

return False

# функция для проверки расположения элементов массива в убывающем порядке

def check\_decrease(arr):

for i in range(len(arr)-1):

if not arr[i] >= arr[i+1]:

return False

return True

## Листинг А.2

**-** Код для решения задачи №1

# функция для проверки нахождения элемента в массиве

def check(temp, el):

for i in temp:

if i == el:

return 1

def flats(s):

# запоминаем первый элемент массива, инициализируем точки входа и выхода

temp, ent, out = [s[0]], 0, 0

for i in range(len(s)-1):

if s[i] == s[i+1] : # сравниваем соседние элементы

if temp[0] == s[i+1]: # если встречался только данный элемент

ent = i+1 # смещаем точку входа

elif check(temp, s[i+1]) != 1: # проверяем, находится элемент в массиве

out = i+1 # если нет, то смещаем точку выхода

temp.append(s[i+1]) # и добавляем в массив

return out-ent+1

try:

n = int(input('Введите количество квартир: '))

if not check\_range(n, 1, 100000): sys.exit('Выход за пределы диапазона! 1 <= n <= 100000')

s = list(map(str, input('Введите покемонов в квартирах: ')))

if not check\_ascii(s): sys.exit('Входные данные должны состоять из строчных и прописных букв английского алфавита!')

if not check\_equal(n, s): sys.exit('Количество введенных покемонов не соответствует номеру квартир!')

else: print('Минимальное число квартир для посещения = ', flats(s))

except ValueError:

print('Некорректные входные данные!')

except SystemExit as e:

print('%s' % e.code)

## Листинг А.3

**-** Код для решения задачи №2

def func(n, arr):

diff = []

for i in range(1, n): # заносим в массив разность между соседними элементами

diff.append(arr[i]-arr[i-1])

# создаем словарь из массива с полученными разностями и считаем количество повторений

frequency = {diff.count(x):x for x in set(diff)}

# если кол-во макс-ых разностей = длине массива

if max(frequency.keys()) == len(diff):

# массив последовательный и менять элементы не нужно

count = 0

# если разница длины массива и кол-ва макс-ых разностей = 1

elif len(diff) - max(frequency.keys()) == 1:

# счетчик равен количеству чисел с неповторяющейся разницей + 1

count = len(diff) - max(frequency.keys()) + 1

else:

# счетчик равен количеству чисел с неповторяющейся разницей

count = len(diff) - max(frequency.keys())

return count

try:

t = int(input('Введите количество наборов входных данных: '))

if not check\_range(t, 1, 100): sys.exit('Выход за пределы диапазона! 1 <= t <= 100')

for i in range(t):

n = int(input('Введите количество элементов массиве: '))

if not check\_range(n, 1, 70): sys.exit('Выход за пределы диапазона! 1 <= n <= 70')

arr = list(map(int, input('Введите массив: ').split()))

if not check\_arr(arr, -100, 100): sys.exit('Выход за пределы диапазона! -100 <= a[i] <= 100')

if not check\_equal(n, arr): sys.exit('Количество элементов массива не соответствует заданному номеру!')

elif len(arr) > 1: print('Минимальное количество элементов, которые нужно заменить = ', func(n, arr))

else: print('Минимальное количество элементов, которые нужно заменить = 0')

except ValueError:

print('Некорректные входные данные!')

except SystemExit as e:

print('%s' % e.code)

## Листинг А.4

**-** Код для решения задачи №3

def change(one, two):

while len(one) != len(two): # пока массивы не равны

one.append(two[0]) # добавляем в меньший элемент из большего

del two[0] # и удаляем его из большего

return one, two

def check(one, two):

for temp in one[:]: # проходимся по образам массивов

for elem in two[:]:

if temp == elem: # если находим одинаковые

two.remove(elem) # то удаляем из обоих массивов

one.remove(temp)

break

return one, two

def fill(s, c):

# заносим цвета носков в 'левый' и 'правый' массивы

count, j = 0, - 1

l, r = [], []

for i in range(int(s[1])):

l.append(int(c[i]))

j = i

for i in range(int(s[2])):

r.append(int(c[j+1]))

j += 1

a, b = max(l, r), min(l, r) # определяем, макс и мин массив

count = (len(a)-len(b))//2 # считаем предварительный ответ

change(b, a) # делаем 'левый' и 'правый' массивы равными

check(a, b) # убираем носки одного цвета в разных массивах

return count + len(l) # добавляем к счетчику оставшуюся длину одного массива

try:

t = int(input("Введите количество наборов входных данных: "))

if not check\_range(t, 1, 1000): sys.exit('Выход за пределы диапазона! 1 <= t <= 1000')

for i in range(t):

s = list(map(int, input("Введите общее количество носков, количество левых и правых носков: ").split()))

if not check\_range(s[0], 2, 2\*pow(10,5)) or s[0]%2 != 0:

sys.exit('Выход за пределы диапазона! n - четное число и 2 <= n <= 2\*10^5')

if not (check\_range(s[1], 1, s[0]) or check\_range(s[2], 1, s[0])) or s[1]+s[2]!=s[0]:

sys.exit('Выход за пределы диапазона! 0 <= l,r <=n; l+r=n')

c = list(map(int, input("Введите цвета носков: ").split()))

if not check\_arr(c, 1, s[0]): sys.exit('Выход за пределы диапазона! 1 <= c[i] <= n')

if not check\_equal(s[0], c): sys.exit('Количество элементов массива не соответствует заданному номеру!')

print('Минимальное количество денег, которое нужно заплатить = ', fill(s, c))

except ValueError:

print('Некорректные входные данные!')

except SystemExit as e:

print('%s' % e.code)

## Листинг А.5

**-** Код для решения задачи №4

def same(n, s, t, a1, a2):

difa, difb, pos = [], [], []

for i in range(n):

if s[i] != t[i]: # проходимя по строкам

if s[i] == 'a':

difa.append(i+1) # запоминаем позиции с буквой 'a'

else:

difb.append(i+1) # и 'b'

while len(difa) >= 2: # пока есть >= 2 позиций 'a' для изменения

a1, a2 = difa.pop(0), difa.pop(0) # убираем их

pos.append((a1, a2)) # и заносим в массив позиций

while len(difb) >= 2: # пока есть >= 2 позиций 'b' для изменения

a1, a2 = difb.pop(0), difb.pop(0) # убираем их

pos.append((a1, a2)) # и заносим в массив позиций

if len(difa) + len(difb) == 0: # если массивы пусты

return len(pos), pos # возвращаем ответ

else: # иначе запоминаем первые позиции 'a' или 'b'

a1 = difa[0]

a2 = difb[0]

pos.append((a1, a1)) # заполняем изменения для данного случая

pos.append((a1, a2))

return len(pos), pos # возвращаем ответ

try:

n = int(input("Введите длину строк: "))

if not check\_range(n, 1, 2\*pow(10,5)): sys.exit('Выход за пределы диапазона! 1 <= n <= 2\*10^5')

s = list(map(str, input("Введите первую строку: ")))

t = list(map(str, input("Введите вторую строку: ")))

if not check\_equal(n, t) or not check\_equal(n, s):

sys.exit('Количество элементов строки не соответствует заданному номеру!')

if not check\_ascii\_limited(s) or not check\_ascii\_limited(t):

sys.exit('Строки должны состоять из латинских букв «a» и «b»')

a1 = s.count('a') # считаем кол-во букв 'a' в обеих строках

a2 = t.count('a')

# если нечетное кол-во букв 'a' или 'b'

if (a1+a2)%2 != 0 or (2\*n-a1-a2)%2 != 0: print('Строки невозможно сделать одинаковыми: -1')

else:

print('Минимальное количество операций, которые нужно сделать = ', same(n, s, t, a1, a2)[0])

print('Позиции для обмена букв:')

for elem in same(n, s, t, a1, a2)[1]: # построчно выводим позиции для обмена

print(\*elem)

except ValueError:

print('Некорректные входные данные!')

except SystemExit as e:

print('%s' % e.code)

## Листинг А.6

**-** Код для решения задачи №5

def func(s, arr, rule):

prob = 1.0 # задаем максимальную вероятность

order = s[0] # переменной порядка присваиваем значение количества элементов

# пока порядок < длины массива и элемент = своему индексу + 1

while order <= s[0] and order == arr[order-1]:

order -= 1 # уменьшаем переменную порядка

if order == 0: # если прошли по всему массиву и не нашли несоответствий

prob = 0.0 # присваиваем минимальную вероятность

for elem in rule:

r = int(elem[0]) # длина сортировки

p = float(elem[1]) # вероятность данной сортировки

if order <= r: # если длина сортировки покрывает место найденного несоответствия

prob = prob \* (1-p) # считаем вероятность

return 1-prob

try:

t = int(input('Введите количество наборов входных данных: '))

if not check\_range(t, 1, 100): sys.exit('Выход за пределы диапазона! 1 <= t <= 100')

for i in range(t):

s = list(map(int, input('Введите количество элементов и экспериментов: ').split()))

if not check\_range(s[0], 1, pow(10,5)) or not check\_range(s[1], 1, pow(10,5)):

sys.exit('Выход за пределы диапазона! 1 <= n,m <= 10^5')

arr = list(map(int, input('Введите массив: ').split()))

if not check\_arr(arr, 1, s[0]): sys.exit('Выход за пределы диапазона! 1 <= a[i] <= n')

if not check\_equal(s[0], arr): sys.exit('Количество элементов массива не соответствует заданному номеру!')

rule = []

for i in range(s[1]):

k = input('Введите правило: ').split()

if not (k[0].isdigit() and check\_range(int(k[0]), 1, s[0])):

sys.exit('В правиле должно быть целое число r: 1<=r<=n')

if not check\_float(float(k[1])):

sys.exit('В правиле должно быть вещественное/целое число p: 0<=p<=1')

rule.append(k)

print('Вероятность, что перестановка окажется отсортированной = {:6f}'.format(func(s, arr, rule)))

except ValueError:

print('Некорректные входные данные!')

except SystemExit as e:

print('%s' % e.code)

## Листинг А.7

**-** Код для решения задачи №6

def count(n, p):

medals = [0, 0, 0] # заполняем массив медалей нулями

half = n//2-1 # ищем середину массива

while p[half] == p[half+1]: # если средний элемент массива равен следующему

half -= 1 # уменьшаем середину (медали которые будем распределять)

# считаем золотые медали

for i in range(1, half):

if i\*3 <= half+1-i:

medals[0] += 1

# считаем серебряные медали

for i in range(medals[0], n):

if (half+2-medals[0]-medals[1])//2 > medals[0]:

medals[1] += 1

# считаем бронзовые медали

medals[2] = half+1-medals[0]-medals[1]

return medals

try:

t = int(input('Введите количество наборов входных данных: '))

if not check\_range(t, 1, 10000): sys.exit('Выход за пределы диапазона! 1 <= t <= 10000')

for i in range(t):

n = int(input('Введите количество участников: '))

if not check\_range(n, 1, 4\*pow(10,5)): sys.exit('Выход за пределы диапазона! 1 <= n <= 4\*10^5')

p = list(map(int, input('Введите массив: ').split()))

if not check\_arr(p, 0, pow(10,6)): sys.exit('Выход за пределы диапазона! 0 <= p[i] <= 10^6')

if not check\_equal(n, p): sys.exit('Количество элементов массива не соответствует заданному номеру!')

if not check\_decrease(p): sys.exit('Элементы массива должны быть в убывающем порядке!')

# исключаем случай, когда медали нельзя будет распределить согласно условиям

if n<6: print('Решения не существует: 0 0 0')

else: print('Количество золотых, серебряных и бронзовых медалей = ', \*count(n, p))

except ValueError:

print('Некорректные входные данные!')

except SystemExit as e:

print('%s' % e.code)

## Листинг А.8

**-** Код для решения задачи №7

def good(a, n):

diff = [] # заполняем массив модулями разностей соседних эл-ов

for i in range(1, n):

diff.append(abs(a[i]-a[i-1]))

# если массив состоит из 2 чисел и разница элементов = 1

if len(diff) == 1 and diff[0] == 1:

return 1

# если все элементы массива одинаковые

elif diff.count(0) == len(diff):

return fact(n)

# если несколько раз встречается одинаковая разность

elif contains(a, max(a)-min(a)):

return rearr(n, a)

else: return 0

def fact(n): # считаем факториал = длине массива

res = 1

for i in range(1, n+1):

res = res\*i

return res

def rearr(n, a):

res = 1

a.sort(reverse=True) # сортируем массив по возрастанию

frequency = {a.count(x):x for x in set(a)} # создаем словарь значениq и их количества

temp = max(frequency.keys()) # выбираем максимальное

for i in range(1, n):

if a[i] == temp: # если элемент повторяется

res = res\*temp # умножаем результат на сам элемент массива

else: res = res\*i # иначе считаем как обычный факториал

return res

try:

t = int(input('Введите количество наборов входных данных: '))

if not check\_range(t, 1, pow(10,4)): sys.exit('Выход за пределы диапазона! 1 <= t <= 10^4')

for i in range(t):

n = int(input('Введите количество членов жюри: '))

if not check\_range(n, 2, 2\*pow(10,5)): sys.exit('Выход за пределы диапазона! 2 <= n <= 2\*10^5')

a = list(map(int, input('Введите задачи: ').split()))

if not check\_arr(a, 1, pow(10,9)): sys.exit('Выход за пределы диапазона! 1 <= a[i] <= 10^9')

if not check\_equal(n, a): sys.exit('Количество элементов массива не соответствует заданному номеру!')

m = 998244353

print('Количество хороших перестановок = ', good(a, n)%m)

except ValueError:

print('Некорректные входные данные!')

except SystemExit as e:

print('%s' % e.code)

## Листинг А.9

**-** Код для решения задачи №8

def remake(t):

rest = [0, 0, 0] # заполняем массив остатков нулями

m = pow(10,9)+7 # задаем число для деления

k = t[2]-t[1]+1 # считаем кол-во элементов в диапазоне

for i in range(3): # заполняем массив остатков количеством

rest[i] = k//3 # чисел с данным остатком от деления на 3

if k%3 != 0:

temp = t[1] % 3

rest[temp] += 1

cnt = [rest[0], rest[1], rest[2]] # заполняем массив-счетчик полученными значениями

for i in range(2, t[0]+1): # считаем количество перестановок для остатков

cnt0 = cnt[0]\*rest[0] + cnt[1]\*rest[2] + cnt[2]\*rest[1]

cnt1 = cnt[2]\*rest[2] + cnt[0]\*rest[1] + cnt[1]\*rest[0]

cnt2 = cnt[1]\*rest[1] + cnt[0]\*rest[2] + cnt[2]\*rest[0]

cnt[0] = cnt0 % m

cnt[1] = cnt1 % m

cnt[2] = cnt2 % m

return cnt[0]

try:

t = list(map(int, input('Введите размер массива и диапазон чисел: ').split()))

if not check\_range(t[0], 1, 2\*pow(10,5)): sys.exit('Выход за пределы диапазона! 1 <= n <= 2\*10^5')

if not check\_range(t[1], 1, pow(10,9)): sys.exit('Выход за пределы диапазона! 1 <= l <= 10^9')

if not check\_range(t[2], 1, pow(10,9)): sys.exit('Выход за пределы диапазона! 1 <= t <= 10^9')

print('Количество способов восстановить массив = ', remake(t))

except ValueError:

print('Некорректные входные данные!')

except SystemExit as e:

print('%s' % e.code)

## Листинг А.10

**-** Код для решения задачи №9

def zeros(n, a, b):

temp = []

for i in range(n): # заполняем массив частными != 0

if a[i] != 0:

temp.append(b[i]/a[i])

if temp == []: # если массив пустой

cnt = 0 # максимизировать нули не получится

else:

# создаем словарь частныхи и их количества

frequency = {temp.count(x):x for x in set(temp)}

# выбираем самое частое повторение

cnt = max(frequency.keys())

return cnt

try:

n = int(input('Введите количество элементов: '))

if not check\_range(n, 1, 2\*pow(10,5)): sys.exit('Выход за пределы диапазона! 1 <= n <= 2\*10^5')

a = list(map(int, input('Введите массив: ').split()))

if not check\_arr(a, pow(-10,9), pow(10,9)): sys.exit('Выход за пределы диапазона! -10^9 <= a[i] <= 10^9')

if not check\_equal(n, a): sys.exit('Количество элементов массива не соответствует заданному номеру!')

b = list(map(int, input('Введите массив: ').split()))

if not check\_arr(a, pow(-10,9), pow(10,9)): sys.exit('Выход за пределы диапазона! -10^9 <= b[i] <= 10^9')

if not check\_equal(n, a): sys.exit('Количество элементов массива не соответствует заданному номеру!')

print('Максимальное количество нулей в массиве = ', zeros(n, a, b))

except ValueError:

print('Некорректные входные данные!')

except SystemExit as e:

print('%s' % e.code)

## Листинг А.11

**-** Код для решения задачи №10

def strong(n):

if n%2 == 0: # если число четное

# выбираем минимум из результатов выполнения функций

res = min(degree(n), fact(n))

else: # если нечетное

# выбираем максимум из результатов выполнения функций

res = max(degree(n), fact(n))

return res if res > 0 else '-1' # если вариантов решения нет выводим -1

# проверяем является число степенью 2 и сколько элементов нужно для его представления

def degree(n):

res, cnt = 1, 0

while res < n:

res \*= 2

cnt += 1

# если является возвращаем 1 иначе количество эл-ов - 1

return 1 if res == n else cnt-1

# проверяем является число факториалом и сколько элементов нужно для его представления

def fact(n):

res, k, cnt = 1, 2, 0

while res < n:

res = res\*k

k += 1; cnt += 1

# если является возвращаем 1 иначе количество эл-ов - 1

return 1 if res == n else cnt-1

try:

t = int(input('Введите количество наборов входных данных: '))

if not check\_range(t, 1, 100): sys.exit('Выход за пределы диапазона! 1 <= t <= 100')

for i in range(t):

n = int(input('Введите число: '))

if not check\_range(n, 1, pow(10,12)): sys.exit('Выход за пределы диапазона! 1 <= n <= 10^12')

print('Минимально возможное число = ', strong(n))

except ValueError:

print('Некорректные входные данные!')

except SystemExit as e:

print('%s' % e.code)