**Федеральное агентство связи**

**Ордена Трудового Красного Знамени**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

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

Кафедра “Математическая кибернетика и информационные технологии”

**Отчет по курсовой работе**

по дисциплине «Структуры и алгоритмы обработки данных»

Вариант №2.

Выполнил: студент группы БВТ2004

Станолевич Владислав Станиславович

Проверил:

Мкртчян Грач Маратович

Москва, 2022

# Задачи

## Задача №1

Вам задано целое положительное число 𝑛. За один ход вы можете увеличить 𝑛 на единицу (то есть сделать 𝑛:=𝑛+1). Ваша задача — найти минимальное количество ходов, которое надо совершить, чтобы сделать сумму цифр 𝑛 не превышающей 𝑠.

Вам необходимо ответить на 𝑡 независимых наборов тестовых данных.

*Входные данные:* первая строка входных данных содержит одно целое число 𝑡 (1≤𝑡≤2⋅104) — количество наборов тестовых данных. Затем следуют 𝑡 наборов тестовых данных. Единственная строка набора тестовых данных содержит два целых числа 𝑛 и 𝑠 (1≤𝑛≤1018; 1≤𝑠≤162).

*Выходные данные:* на каждый набор тестовых данных выведите ответ: минимальное количество ходов, которое надо совершить, чтобы сделать сумму цифр 𝑛 не превышающей 𝑠.

## Задача №2

Назовем левым циклическим сдвигом некоторой строки 𝑡1𝑡2𝑡3…𝑡𝑛−1𝑡𝑛 следующую строку: 𝑡2𝑡3…𝑡𝑛−1𝑡𝑛𝑡1. Аналогично, назовем правым циклическим сдвигом строки 𝑡 строку 𝑡𝑛𝑡1𝑡2𝑡3…𝑡𝑛−1.

Скажем, что строка 𝑡 является хорошей, если ее левый циклический сдвиг равен правому циклическому сдвигу.

Вам дана строка 𝑠, состоящая из цифр 0–9.

Какое минимальное количество символов необходимо удалить из строки 𝑠, чтобы она стала хорошей?

*Входные данные:* первая строка содержит единственное целое число 𝑡 (1≤𝑡≤1000) — количество наборов входных данных. Следующие 𝑡 строк содержат описание наборов входных данных. Первая и единственная строка каждого набора содержит строку 𝑠 (2≤|𝑠|≤2⋅105). Каждый символ 𝑠𝑖 является цифрой 0–9.

Гарантируется, что суммарная длина строк не превышает 2⋅105.

*Выходные данные:* для каждого набора входных данных выведите минимальное количество символов, которое необходимо удалить из строки 𝑠, чтобы она стала хорошей.

## Задача №3

Вам задана бинарная строка 𝑠, состоящая из 𝑛 нулей или единиц.

Ваша задача – разделить заданную строку на минимальное число подпоследовательностей таким образом, что каждый символ строки принадлежит ровно одной подпоследовательности и каждая подпоследовательность выглядит подобно «010101 ...» или «101010 ...» (т.е. подпоследовательность не должна содержать два соседних нуля или единицы).

Напомним, что подпоследовательность — это последовательность, которая может быть получена путем удаления из заданной последовательности с помощью удаления нуля или более элементов без изменения порядка остальных элементов. Например, подпоследовательностями «1011101» являются «0», «1», «11111», «0111», «101», «1001», но не «000», «101010» и «11100».

Вам необходимо ответить на 𝑡 независимых наборов тестовых данных.

*Входные данные:* первая строка теста содержит одно целое число 𝑡 (1≤𝑡≤2⋅104) — количество наборов тестовых данных. Затем следуют 𝑡 наборов тестовых данных. Первая строка набора тестовых данных содержит одно целое число 𝑛 (1≤𝑛≤2⋅105) — длину 𝑠. Вторая строка набора тестовых данных содержит 𝑛 символов '0' и '1' — строку 𝑠. Гарантируется, что сумма всех 𝑛 не превосходит 2⋅105 (∑𝑛≤2⋅105).

*Выходные данные:* Для каждого набора тестовых данных выведите ответ на него: первой строкой выведите одно целое число 𝑘 (1≤𝑘≤𝑛) — минимальное количество подпоследовательностей, на которые вы можете разделить строку 𝑠. Второй строкой выведите 𝑛 целых чисел 𝑎1,𝑎2,…,𝑎𝑛 (1≤𝑎𝑖≤𝑘), где 𝑎𝑖 — номер подпоследовательности, к которой принадлежит 𝑖-й символ строки 𝑠. Если существует несколько ответов, вы можете вывести любой.

## Задача №4

Патрик любит играть в бейсбол, но иногда он тратит так много часов на пробежки, что его разум начинает затуманиваться! Патрик уверен, что его набранные очки за 𝑛 игр соответствуют тождественной перестановке (т.е. в первой игре он набирает 1, во второй игре он набирает 2 и так далее). Однако, когда он посмотрел на свои записи, он увидел, что все значения перепутаны!

Определим специальный обмен следующим образом: выберите любой подмассив очков и переставьте местами его элементы так, чтобы ни один элемент не оказался в той же позиции, где он был до обмена. Например, выполнение специального обмена на [1,2,3] может дать [3,1,2], но не может дать [3,2,1], так как 2 находится в той же позиции.

Вам дана перестановка из 𝑛 целых чисел. Пожалуйста, помогите Патрику найти минимальное количество специальных обменов, необходимых для того, чтобы сделать ее отсортированной! Можно доказать, что при данных ограничениях это число не превышает 1018.

Массив 𝑎 является подмассивом массива 𝑏, если 𝑎 можно получить из 𝑏, удалив несколько (возможно, ноль или все) элементов из начала и несколько (возможно, ноль или все) элементов с конца.

*Входные данные:* каждый тест содержит несколько наборов входных данных. Первая строка содержит количество наборов входных данных 𝑡 (1≤𝑡≤100). Описание наборов входных данных приведено ниже. Первая строка каждого набора входных данных содержит целое число 𝑛 (1≤𝑛≤2⋅105) — длину данной перестановки.тВторая строка каждого набора входных данных содержит 𝑛 целых чисел 𝑎1,𝑎2,...,𝑎𝑛 (1≤𝑎𝑖≤𝑛) — начальную перестановку.

*Выходные данные:* для каждого набора входных данных выведите одно целое число: минимальное количество специальных обменов, необходимых для сортировки перестановки.

## Задача №5

Вам дана строка 𝑠[1…𝑛], состоящая из строчных латинских букв. Гарантируется, что 𝑛=2𝑘 для некоторого целого числа 𝑘≥0. Строка 𝑠[1…𝑛] называется 𝑐-хорошей, если выполняется как минимум одно из следующих условий:

* Длина строки 𝑠 равна 1 и она состоит из единственного символа 𝑐 (то есть 𝑠1=𝑐);
* Длина строки 𝑠 больше 1, первая половина строки состоит только из символа 𝑐 (то есть 𝑠1=𝑠2=⋯=𝑠𝑛2=𝑐), а вторая половина строки (то есть строка 𝑠𝑛2+1𝑠𝑛2+2…𝑠𝑛) является (𝑐+1)-хорошей строкой;
* Длина строки 𝑠 больше 1, вторая половина строки состоит только из символа 𝑐 (то есть 𝑠𝑛2+1=𝑠𝑛2+2=⋯=𝑠𝑛=𝑐), а первая половина строки (то есть строка 𝑠1𝑠2…𝑠𝑛2) является (𝑐+1)-хорошей строкой.

За один ход вы можете выбрать один индекс 𝑖 от 1 до 𝑛 и заменить 𝑠𝑖 на любую строчную латинскую букву (любой символ от 'a' до 'z').

Ваша задача — найти минимальное количество ходов, необходимое, чтобы получить 'a'-хорошую строку из 𝑠 (т.е. 𝑐-хорошую строку для 𝑐= 'a'). Гарантируется, что ответ всегда существует.

Вам нужно ответить на 𝑡 независимых наборов тестовых данных.

*Входные данные*: первая строка теста содержит одно целое число 𝑡 (1≤𝑡≤2⋅104) — количество наборов тестовых данных. Затем следуют 𝑡 наборов тестовых данных. Первая строка набора тестовых данных содержит одно целое число 𝑛 (1≤𝑛≤131 072) — длину 𝑠. Гарантируется, что 𝑛=2𝑘 для некоторого целого числа 𝑘≥0. Вторая строка набора тестовых данных содержит строку 𝑠, состоящую из 𝑛 строчных латинских букв.

*Выходные данные:* для каждого набора тестовых данных выведите ответ на него — наименьшее количество ходов, необходимое, чтобы получить 'a'-хорошую строку 𝑠 (т.е. 𝑐-хорошую строку для 𝑐= 'a'). Гарантируется, что ответ существует.

## Задача №6

Рассмотрим следующий процесс. У вас есть бинарная строка (строка, состоящая только из символов 0 и 1) 𝑤 длины 𝑛 и число 𝑥. Вы создаете новую бинарную строку 𝑠 длины 𝑛; 𝑖-й символ новой строки 𝑠 выбирается следующим образом:

* если символ 𝑤𝑖−𝑥 существует и равен 1, то 𝑠𝑖 равно 1 (более формально, если 𝑖>𝑥 и 𝑤𝑖−𝑥= 1, то 𝑠𝑖= 1);
* если символ 𝑤𝑖+𝑥 существует и равен 1, то 𝑠𝑖 равно 1 (более формально, если 𝑖+𝑥≤𝑛 и 𝑤𝑖+𝑥= 1, то 𝑠𝑖= 1);
* если оба описанных выше условия неверны, то 𝑠𝑖 равно 0.

Вам заданы число 𝑥 и строка 𝑠. Восстановите изначальную строку 𝑤.

*Входные данные:* первая строка содержит число 𝑡 (1≤𝑡≤1000) — количество наборов входных данных.

Каждый набор входных данных содержит две строки. Первая строка содержит строку 𝑠 (2≤|𝑠|≤105, каждый символ строки 𝑠 равен либо 0, либо 1). Вторая строка содержит целое число 𝑥 (1≤𝑥≤|𝑠|−1).Суммарная длина всех длин строк 𝑠 во входных данных не превосходит 105.

*Выходные данные*: на каждый набор входных данных выведите ответ:

* если не существует строки 𝑤, которая может породить строку 𝑠, то выведите −1;
* иначе, выведите бинарную строку 𝑤 состоящую из |𝑠| символов. Если возможных ответов несколько — вы можете вывести любой из них.

## Задача №7

В линию выстроены 𝑛 настоек, причем настойка 1 находится слева, а настойка 𝑛 — справа. Каждая настойка увеличит ваше здоровье на 𝑎𝑖, если ее выпить. 𝑎𝑖 может быть отрицательным, что означает, что настойка уменьшит ваше здоровье.

Вы начинаете с 0 здоровья и будете идти слева направо, от первой настойки до последней. Для каждой настойки вы можете выбрать, выпить ли ее. Вы должны следить за тем, чтобы ваше здоровье всегда было неотрицательным.

Какое наибольшее количество настоек вы можете выпить?

*Входные данные:* первая строка содержит одно целое число 𝑛 (1≤𝑛≤2000) — количество настоек. Следующая строка содержит 𝑛 целых чисел 𝑎1, 𝑎2, ... ,𝑎𝑛 (−109≤𝑎𝑖≤109), которые обозначают изменения здоровья после употребления данных настоек.

*Выходные данные:* выведите одно целое число — максимальное количество настоек, которое вы можете выпить, чтобы ваше здоровье всегда было неотрицательным.

## Задача №8

Задан массив 𝑎=[𝑎1,𝑎2,…,𝑎𝑛] (1≤𝑎𝑖≤𝑛). Его элемент 𝑎𝑖 называется особым, если существует такая пара индексов 𝑙 и 𝑟 (1≤𝑙<𝑟≤𝑛), что 𝑎𝑖=𝑎𝑙+𝑎𝑙+1+…+𝑎𝑟. Иными словами, элемент называется особым, если он представим в виде суммы двух или более подряд идущих элементов массива (не важно, особых или нет).

Выведите количество особых элементов заданного массива 𝑎.

Обратите внимание, что среди элементов массива 𝑎 могут быть равные — если несколько элементов равны и являются особыми, то все они должны быть посчитаны в ответе.

*Входные данные:* в первой строке записано целое число 𝑡 (1≤𝑡≤1000) — количество наборов входных данных в тесте. Далее следуют описания 𝑡 наборов входных данных. Каждый набор задается двумя строками. В первой строке записано целое число 𝑛 (1≤𝑛≤8000) — длина массива 𝑎. Во второй строке записаны целые числа 𝑎1,𝑎2,…,𝑎𝑛 (1≤𝑎𝑖≤𝑛). Гарантируется, что сумма значений 𝑛 по всем наборам входных данных не превосходит 8000.

*Выходные данные:* выведите 𝑡 чисел — количества особых элементов для каждого из заданных массивов.

## Задача №9

Вам дан массив 𝑎1,𝑎2,…,𝑎𝑛.

За одну операцию вы можете взять любые два элемента 𝑎𝑖 и 𝑎𝑗 (𝑖≠𝑗) и уменьшить каждый из них на единицу.

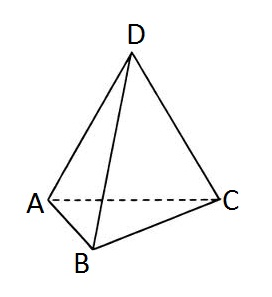
Вам нужно проверить можно ли сделать все элементы массива равными нулю или нет.

*Входные данные:* Первая строка содержит одно целое число 𝑛 (2≤𝑛≤105) — размер массива. Вторая строка содержит 𝑛 целых чисел 𝑎1,𝑎2,…,𝑎𝑛 (1≤𝑎𝑖≤109) — элементы массива.

*Выходные данные:* Выведите «YES», если можно сделать все элементы массива равными нулю. Иначе выведите «NO».

## Задача №10

Вам задан тетраэдр. Обозначим его вершины буквами A, B, C и D соответственно.



В вершине тетраэдра D находится муравей. Муравей очень подвижный и не любит стоять на месте. В каждый момент времени он совершает один шаг от одной вершины к другой по некоторому ребру тетраэдра, оставаться на месте он не может.

От Вас в этой задаче требуется совсем немногое: нужно посчитать каким количеством способов муравей может прийти из исходной вершины D в себя ровно за n шагов. Другими словами, Вас просят узнать количество различных циклических путей длины n из вершины D в себя. Поскольку это количество может быть достаточно большим, ответ требуется посчитать по модулю

1000000007 (109 + 7).

*Входные данные:* в первой строке записано единственное целое число n (1 ≤ n ≤ 107) — требуемая длина циклического пути.

*Выходные данные:* выведите единственное целое число — искомое количество способов по модулю - 1000000007 (109 + 7).

# Выполнение

## Листинг кода задачи №1

## t = int(input())

**for** i **in** range(t):

n, s **=** list(map(int, input()**.**split())) *#Ввод n и s в одну строку*

ans **=** [] *#Массив для ответа*

**while** sum(n) **>** s:

**if** n **%** 10 **!=** 0: *#Если последняя цифра n не равна нулю, мы добавляем в начало ответа число ходов*

ans**.**insert(0, str(10 **-** n**%10**)) #нужное для увеличения на 10

n **=** n**//**10 **+** 1 *#Заменяем разряд n единичкой*

**else**:

ans**.**insert(0, str(0)) *#А если нет, добавляем в начало ноль и уменьшаем n на разряд*

n **=** n**//**10

**if** len(ans) **==** 0:

print(0)

print(''**.**join(ans))

## Листинг кода задачи №2

**from** collections **import** Counter

**def** goodLine(s, a, b):

c **=** 0

**for** i **in** s: *#условие для хорошей строки - каждый 1 и 3 символ - равны, каждый 2 и 4 символ - равны, и т.д.*

**if** i **==** a:

a, b **=** b, a

c **=** c **+** 1

**if** c**%2** != 0: #если длина строки нечётная - все символы условия должны быть равны между собой

c **=** c **-** 1 *#иначе все символы на четных позициях должны быть равны,*

*#и все символы на нечетных позициях должны быть равны*

**return** c

t **=** int(input())

**for** i **in** range(t):

s **=** list(input())

l **=** len(s)

maxx **=-** 1

**for** k **in** range(10): *#существует всего 10 типов таких символов, и мы их перебираем*

**for** j **in** range(10): *#для каждой комбинации набирается подпоследовательность s,*

**if** k **==** j: *#начинающаяся с этих двух символов*

x **=** s**.**count(str(k))

**else**:

x **=** goodLine(s, str(k), str(j))

maxx **=** max(maxx,x)

print(l **-** maxx)

## Листинг кода задачи №3

**def** solve(s, n):

cur **=** [set(), set()] *#Массив из двух множеств нулей и единиц*

used **=** 1 *#Кол-во использованных подпоследовательностей*

ans **=** [1] *#Массив номеров подпоследовательностей, относительно цифр в s*

**if** s[0] **==** "0":

cur[0]**.**add(0) *#заносим первый элемент в соответствующее множество*

**else**:

cur[1]**.**add(0)

**for** c **in** s[1:]: *#проверяем элементы после первого*

**if** c **==** "0" **and** len(cur[1]) **!=** 0: *#сравниваем этот элемент с предыдущим, и определяем возможна ли неповторяющаяся*

i **=** cur[1]**.**pop() *#подпоследовательность, до тех пор, пока цифры не повторятся, постепенно заменяя*

cur[0]**.**add(i) *#первый элемент в множестве нулей и единиц сравнение идёт всегда с ним*

ans**.**append(i **+** 1) *#пишем, к какой подпослеовательности относится элемент*

**elif** c **==** "1" **and** len(cur[0]) **!=** 0:

i **=** cur[0]**.**pop()

cur[1]**.**add(i)

ans**.**append(i **+** 1)

**else**:

used **+=** 1 *#в случае, если значения не совпали, мы увеличиваем кол-во подпоследовательностей*

ans**.**append(used)

**if** c **==** "0":

cur[0]**.**add(used **-** 1)

**else**:

cur[1]**.**add(used **-** 1)

print(used)

print(**\***ans)

t **=** int(input())

**for** i **in** range(t):

n **=** int(input())

s **=** input()

solve(s, n)

## Листинг кода задачи №4

t **=** int(input())

**for** i **in** range(t):

n **=** int(input())

arr **=** list(map(int, input()**.**split()))

arrSort **=** sorted(arr)

count **=** 0 *#счётчик элементов подпоследовательности*

k **=** 0 *#счётчик перестановок*

**for** i **in** range(len(arr)):

**if** arr[i] **!=** arrSort[i]: *#условия для существования перестановки*

count **+=** 1

**elif** arr[i] **==** arrSort[i] **and** count **!=** 0: *#если нашли совпадающие элементы, то подпоследовательность с уникальными*

count **=** 0 *#позициями перестановки закончилась, увеличиваем счётчик перестановок*

k **+=** 1

**if** count **!=** 0:

k **+=** 1

print(min(k, 2))

## Листинг кода задачи №5

**def** goodStr(s, n, c): *#определяет минимальное кол-во изменений, требуемое для создания хорошей строки*

**if** n **==** 1:

**if** s[0] **!=** c:

**return** 1 *#1-e условие*

**else**:

**return** 0

mid **=** n **//** 2

s1 **=** s[:n **//** 2] *#проверяем первую половину*

l1**=**(s1**.**count(c)) *#кол-во вхождений символа c в s*

s2 **=** s[n **//** 2:] *#проверяем вторую половину*

r2 **=** (s2**.**count(c))

secondCon **=** mid **-** l1 **+** goodStr(s2,n **//** 2,chr(ord(c) **+** 1)) *#2-е условие*

thirdCon **=** mid **-** r2 **+** goodStr(s1,n **//** 2,chr(ord(c) **+** 1)) *#3-е условие*

**return**(min(secondCon,thirdCon))

t **=** int(input())

**for** i **in** range(t):

n **=** int(input())

s **=** input()

print(goodStr(s, n, 'a'))

## Листинг кода задачи №6

**def** solve(s, x):

n **=** len(s)

res **=** ['1'] **\*** n

**for** j **in** range(n):

**if** s[j] **==** '0':

**if** j **-** x **>** **-**1:

res[j **-** x] **=** '0' *#"воссоздаём" старую строку по заданному условия*

**if** j **+** x **<** n:

res[j **+** x] **=** '0'

**for** j **in** range(n):

**if** s[j] **==** '1':

**if** j **-** x **>** **-**1 **and** res[j **-** x] **==** '1':

**continue** *#условия схоядтся*

**if** j **+** x **<** n **and** res[j **+** x] **==** '1':

**continue**

print('-1') *#строки не существует*

**return**

print(''**.**join(res))

t **=** int(input())

**for** i **in** range(t):

s **=** input()

x **=** int(input())

solve(s, x)

## Листинг кода задачи №7

**import** heapq *#куча*

**def** healthPoint(i, num):

count **=** 0 *#счётчик значения здоровья*

total **=** 0 *#сколько всего настоек*

heap **=** list() *#куча выпитого*

**for** i **in** range(n): *#изначально пьём всё подряд*

count **+=** a[i]

total **+=** 1

heapq**.**heappush(heap, a[i]) *#помещаем значение настойки*

**while** count **<** 0: *#а есть счётчик здоровья упал ниже нуля, вычисляем, что было ядом, и что надо вылить*

poison **=** heapq**.**heappop(heap) *#удаляем яд из выпитого*

count **-=** poison *#вычитаем из здорья показатели яда*

total **-=** 1 *#и удаляем банку*

print(total)

n **=** int(input())

a **=** list(map(int, input()**.**split()))

healthPoint(n**-**1, 0)

## Листинг кода задачи №8

t **=** int(input())

**for** i **in** range(t):

n **=** int(input())

arr **=** list(map(int,input()**.**split()))

c **=** 0

m **=** [0]**\***(n**+**1)

**for** i **in** range(n):

s **=** arr[i] *#выделяем проверяемое число*

**for** j **in** range(i**+**1,n): *#проверка выполнения "особости" цифры*

s **+=** arr[j]

**if** s **<=** n:

m[s] **=** 1 *#собираем список особых чисел*

**else**:

**break**

**for** k **in** arr: *#считаем кол-во особых чисел*

**if** m[k] **>**0:

c **+=** 1

print(c)

## Листинг кода задачи №9

n **=** int(input())

a **=** list(map(int, input()**.**split()))

summa **=** sum(a)

**if** max(a) **<=** summa **and** summa **%** 2 **==** 0: *#условие выполняется, в случае если максимальный элемент*

print('YES') *#меньше или равен сумме введённых цифр и если сумма цифр - чётное число*

**else**:

print('NO')

## Листинг кода задачи №10

n **=** int(input())

mod **=** int(1e9**+**7) *#модуль*

D **=** 1

ABC **=** 0

**while** n **>** 0:

n **-=** 1 *#счётчик шагов*

D, ABC **=** (3 **\*** ABC), (2 **\*** ABC **+** D) *#За N ходов попасть в D из D - утроенное количество способов*

D **%=** mod *#попасть из не-D в D за n-1 ход. За N ходов попасть в D из не-D -*

ABC **%=** mod *#удвоенное количество способов попасть из не-D в не-D за n-1 ход*

*#+ количество способов попасть за n-1 ход из D в D.*

print(D)

# Вывод

В ходе проделанной курсовой работы, я выполнил 10 алгоритмических задач на языке Python.