**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 «Кафедра «Математическая кибернетика и информационные технологии**»**



**Курсовая работа**

по дисциплине «Структуры и алгоритмы обработки данных»

на тему:

«Решение задач**»**

Выполнил: студент группы

БВТ2005

Быковская Арина

Александровна

Руководитель: Симонов С.Е.

Москва 2022

**Содержание**

[1.1 Постановка задачи 10](#_Toc103642389)

[1.3 Результат 11](#_Toc103642398)

[2.1 Постановка задачи 11](#_Toc103642399)

[2.3 Результат 13](#_Toc103642400)

[3.1 Постановка задачи 14](#_Toc103642401)

[3.3 Результат 15](#_Toc103642402)

[4.1 Постановка задачи 16](#_Toc103642403)

[4.3 Результат 18](#_Toc103642404)

[5.1 Постановка задачи 19](#_Toc103642405)

[5.3 Результат 22](#_Toc103642406)

[6.1 Постановка задачи 24](#_Toc103642407)

[6.3 Результат 26](#_Toc103642408)

[7.1 Постановка задачи 28](#_Toc103642409)

[7.3 Результат 30](#_Toc103642410)

[8.1 Постановка задачи 31](#_Toc103642411)

[8.3 Результат 32](#_Toc103642412)

[9.1 Постановка задачи 33](#_Toc103642413)

[9.3 Результат 34](#_Toc103642414)

[10.1 Постановка задачи 35](#_Toc103642415)

[10.3 Результат 37](#_Toc103642416)

[Вывод 38](#_Toc103642417)

**Функции к заданиям.**

# к заданию 1

def check\_t1(t1):

if not (1 <= t1 <= 15):

raise ValueError('нарушены условия 1 <= t1 <= 15')

return 0

def check\_n1(n1):

if not (2 <= n1 <= 2000):

raise ValueError('нарушены условия 2<=n1<=2000')

return 0

def check\_a1(a1):

for item in a1:

if not (0 <= item <= 230):

raise ValueError('нарушены условия 0<=item<=230')

return 0

def calculations1(a1,n1,xor):

for i in range(n1):

xor ^= a1[i]

if xor == 0:

mess='YES'

else:

cnt = 0

xr = 0

for i in range(n1):

xr ^= a1[i]

if xor == xr:

cnt += 1

xr = 0

if cnt >= 2:

mess='YES'

else:

mess='NO'

return mess

# к заданию 2

def check\_n2\_k2(n2,k2):

if not (1 <= n2 <= 100000):

raise ValueError('нарушены условия 1<=n2<=100000')

if not (0 <= k2 <= n2):

raise ValueError('нарушены условия 0<=k2<=n2')

return 0

def calculations2(k2,c2,d2,b2,a2):

for i in c2:

d2[i] += 12

if min(d2.values()) > k2:

d2[c2[b2]] -= 1

b2 += 1

else:

a2 += 1

return a2

# к заданию 3

def check\_t3(t3):

if not (1 <= t3 <= 10 \*\* 4):

raise ValueError('нарушены условия 1 <= t3 <= 10^4')

return 0

def check\_a3\_b3(a3,b3):

if not (1 <= a3 < b3 <= 10 \*\* 9):

raise ValueError('1 <= a3<b3 <= 10^9')

return 0

def calculations3(s3,b3,a3):

while b3 > 0:

s3 += b3 - a3

a3 //= 10

b3 //= 10

return s3

# к заданию 4

def check\_t4(t4):

if not (1 <= t4 <= 105):

raise ValueError('нарушены условия 1 <= t4 <= 105')

return 0

def check\_n4\_k4(n4,k4):

if not (1 <= k4 < n4 <= 2 \* (10 \*\* 5)):

raise ValueError('нарушены условия 1 <= k4<n4 <= 2\*(10^5)')

return 0

def calculations4(a4,s4,n4,k4):

for i in range(0, k4):

b4 = [0] \* 26

for j4 in range(i, n4, k4):

b4[ord(s4[j4]) - 97] += 1

b4[ord(s4[k4 - 1 - j4]) - 97] += 1

a4 += sum(b4) - max(b4)

return a4

# к заданию 8

def check\_t8(t8):

if not (1 <= t8 <= 10 \*\* 4):

raise ValueError('нарушены условия 1 <= t8 <= 10 \*\* 4')

return 0

def check\_n8(n8):

if not (1 <= n8 <= 10 \*\* 9):

raise ValueError('нарушены условия 1<=n8<=10\*\*9')

return 0

def calculations8(n8):

s8 = int((((n8 \* 2 - 1) \*\* 0.5) - 1) / 2)

return s8

# к заданию 9

def check\_n9(n9):

if not (3 <= n9 <= 10 \*\* 5):

raise ValueError('нарушены условия 3<=n9<=10\*\*5')

return 0

def check\_ar9(ar9):

if not (1 <= len(ar9) <= 10 \*\* 9):

raise ValueError('нарушены условия 1 <= len(ar9) <= 10\*\*9')

return 0

def calculations9(ar9,n9):

ar9.sort()

c9 = int(ar9[-1] \*\* (1 / (n9 - 1)) + 0.5)

count9 = 0

for i in range(n9):

count9 += abs(ar9[i] - c9 \*\* i)

return count9

# к заданию 10

def check\_p10(p10):

if not (1 <= p10 <= 2 \* (10 \*\* 4)):

raise ValueError('нарушены условия 1 <= p10 <= 2\*(10\*\*4)')

return 0

def check\_n10(n10):

if not (1 <= n10 <= 2 \* (10 \*\* 5)):

raise ValueError('нарушены условия 1<=n10<=2\*(10\*\*5)')

return 0

def check\_item(item):

for item in item:

if not (0 <= item <= 1):

raise ValueError('нарушены условия 0<=l<=1')

return 0

def calculations10(n10,c10,ans10,l10):

for i in range(1, n10):

if (l10[i] == 1):

c10 += 1

else:

ans10 += c10 // 3

c10 = 0

ans10 += c10 // 3

return ans10

1.1 Постановка задачи

Требуется найти оптимальное решение задачи в среде разработки Jupyter Notebook.

Ехаб известен своей любовью к определенным операциям. Сейчас у него есть массив 𝑎 длины 𝑛, и он может делать на нем следующую операцию:

Он выбирает 2 соседних элемента; Ехаб удаляет их из массива и вставляет на их место одно число: их исключающее ИЛИ. Обратите внимание, длина массива уменьшается на один.  
Теперь он спрашивает вас: возможно ли сделать все элементы массива равными? Так как он хочет сделать вашу жизнь сложнее, то вводит дополнительное ограничение: в массиве должно остаться хотя бы 2 элемента.

Входные данные  
В первой строке записано одно целое число 𝑡 (1≤𝑡≤15) — количество наборов входных данных

В первой строке каждого набора входных данных записано целое число 𝑛 (2≤𝑛≤2000) — размер массива 𝑎.

Во второй строке каждого набора входных данных записано 𝑛 целых чисел 𝑎1, 𝑎2, …, 𝑎𝑛 (0≤𝑎𝑖<230) — элементы массива 𝑎.

Выходные данные  
Если Ехаб может сделать все элементы массива равными, применяя описанную операцию, так, чтобы в массиве осталось хотя бы 2 элемента, выведите «YES». Иначе выведите «NO».

Пример  
входные данные  
2  
3  
0 2 2  
4  
2 3 1 10  
выходные данные  
YES  
NO  
Примечание  
В первом примере мы можем удалить первые 2 элемента, 0 и 2, и вставить вместо них 0⊕2=2. Получится массив [2,2], в котором все элементы равны.

Во втором примере ответа не существует.

**1.2 Решение задачи**

Код представлен ниже:

t1=int(input("Введите число: "))

check\_t1(t1)

for p1 in range(t1):

xor = 0

n1 = int(input("Введите число - размер массива: "))

a1 = list(map(int, input("Введите числа - элементы массива: ").split()))

check\_a1(a1)

print(calculations1(a1, n1, xor))

1.3 Результат

Введите число: 2

Введите число - размер массива: 3

Введите числа - элементы массива: 0 2 2

YES

Введите число - размер массива: 4

Введите числа - элементы массива: 2 3 1 10

NO

2.1 Постановка задачи

Требуется найти оптимальное решение задачи в среде разработки Jupyter Notebook.

Школьник Вася получил в подарок на день рождения строку длины n, состоящую из букв «a» и «b». Вася называет привлекательностью строки максимальную длину подстроки (последовательности соседних символов), состоящей из одинаковых символов.

Вася может поменять в исходной строке не более k символов. Какой максимальной привлекательности данной строки он сможет добиться?

Входные данные

В первой строке входных данных содержатся два числа n и k (1 ≤ n ≤ 100 000, 0 ≤ k ≤ n) — длина строки Васи и максимальное разрешённое количество изменений.

Во второй строке следует строка Васи, состоящая из букв «a» и «b».

Выходные данные

Выведите единственное целое число — максимальная привлекательность строки, которую Вася может получить, изменив в исходной строке не более k символов.

Примеры

входные данные

4 2

abba

выходные данные

4

входные данные

8 1

aabaabaa

выходные данные

5

Примечание

В первом примере Вася может получить как строку «aaaa», так и строку «bbbb».

Во втором примере оптимальный ответ достигается на строке «aaaaabaa» или на строке «aabaaaaa».

**2.2 Решение задачи**

Код представлен ниже:

n,k = map(int, input("Введите числа: ").split())

check\_n2\_k2(n,k)

c = input("Введите строку: ")

d = {'a':0, 'b':0}

b = 0

a = 0

print("Результат: ",calculations2(k,c,d,b,a))

2.3 Результат

Введите числа: 4 2

Введите строку: abba

Результат: 4

3.1 Постановка задачи

Требуется найти оптимальное решение задачи в среде разработки Jupyter Notebook.

Вам заданы два целых числа 𝑙 и 𝑟, где 𝑙<𝑟. Будем прибавлять 1 к 𝑙 до тех пор, пока результат не окажется равным 𝑟. Таким образом, будет сделано ровно 𝑟−𝑙 прибавлений единицы. Для каждого такого прибавления посмотрим на количество цифр, которые будут изменены при этом прибавлении. Например:

* если 𝑙=909, то прибавление единицы приведёт к результату 910 и будут изменены 2 цифры;
* если к 𝑙=9 прибавить единицу, то результат будет равен 10 и будут изменены тоже 2 цифры;
* если к 𝑙=489999 прибавить единицу, то результат будет равен 490000 и будут изменены 5 цифр.
* Изменённые цифры всегда образуют некоторый суффикс десятичной записи результата.

Выведите суммарное количество изменённых цифр, если требуется из 𝑙 получить 𝑟, прибавляя каждый раз 1.

Входные данные  
В первой строке находится целое число 𝑡 (1≤𝑡≤104). Далее следуют 𝑡 наборов входных данных.

Каждый дабор входных данных характеризуется двумя целыми числами 𝑙 и 𝑟 (1≤𝑙<𝑟≤109).

Выходные данные  
Для каждого набора входных данных вычислите суммарное количество изменённых цифр, если требуется из 𝑙 получить 𝑟, прибавляя каждый раз единицу.

Пример  
Входные данные  
4  
1 9  
9 10  
10 20  
1 1000000000  
Выходные данные  
8  
2  
11  
1111111110

**3.2 Решение задачи**

Код представлен ниже:

t3=int(input("Введите число: "))

check\_t3(t3)

for p in range(t3):

a3, b3 = map(int, input("Введите донабор чисел: ").split())

check\_a3\_b3(a3, b3)

s3 = 0

print("Результат: ", calculations3(s3,b3,a3))

3.3 Результат

Введите число: 4

Введите донабор чисел: 1 9

Результат: 8

Введите донабор чисел: 9 10

Результат: 2

Введите донабор чисел: 10 20

Результат: 11

Введите донабор чисел: 1 100000000

Результат: 111111110

4.1 Постановка задачи

Требуется найти оптимальное решение задачи в среде разработки Jupyter Notebook.

Слово 𝑠 длины 𝑛 называется 𝑘-полным, если

𝑠 — палиндром, то есть 𝑠𝑖=𝑠𝑛+1−𝑖 для всех 1≤𝑖≤𝑛; 𝑠 имеет период 𝑘, то есть 𝑠𝑖=𝑠𝑘+𝑖 для всех 1≤𝑖≤𝑛−𝑘. Например, «abaaba» — это 3-полное слово, а «abccba» нет.

Бобу вручили слово 𝑠 длины 𝑛, состоящее только из строчных букв латинского алфавита, и целое число 𝑘 такое, что 𝑛 делится на 𝑘. Он хочет превратить слово 𝑠 в любое 𝑘-полное слово.

Для этого Боб может выбирать некоторую позицию 𝑖 (1≤𝑖≤𝑛) и заменять букву на позиции 𝑖 на любую другую строчную букву латинского алфавита.

Поэтому теперь Боба интересует минимальное количество позиций, буквы на которых ему необходимо заменить, чтобы превратить 𝑠 в любое 𝑘-полное слово.

Обратите внимание, что Боб может сделать ноль изменений, если слово 𝑠 уже 𝑘-полное.

Требуется ответить на 𝑡 независимых наборов входных данных.

Входные данные  
В первой строке записано одно целое число 𝑡 (1≤𝑡≤105) — количество наборов входных данных.

В первой строке каждого набора входных данных записаны два целых числа 𝑛 и 𝑘 (1≤𝑘<𝑛≤2⋅105, 𝑛 делится на 𝑘).

Во второй строке записано слово 𝑠 длины 𝑛.

Гарантируется, что слово 𝑠 состоит только из строчных букв латинского алфавита. Также гарантируется, что сумма 𝑛 по всем наборам входных данных не превосходит 2⋅105.

Выходные данные  
Для каждого набора входных данных выведите одно целое число, обозначающее минимальное количество позиций, буквы на которых ему придется заменить, чтобы превратить 𝑠 в любое 𝑘-полное слово.

Пример  
входные данные  
4  
6 2  
abaaba  
6 3  
abaaba  
36 9  
hippopotomonstrosesquippedaliophobia  
21 7  
wudixiaoxingxingheclp  
выходные данные  
2  
0  
23  
16  
Примечание  
В первом наборе входных данных одно из оптимальных решений — это «aaaaaa».

Во втором наборе входных данных слово уже 𝑘-полное.

**4.2 Решение задачи**

Код представлен ниже:

t4 = int(input("Введите число: "))

check\_t4(t4)

for p in range(t4):

a4 = 0

n4, k4 = map(int, input("Введите числа: ").split())

check\_n4\_k4(n4,k4)

s4 = input("Введите слово: ")

print("Результат: ", calculations4(a4,s4,n4,k4)//2)

4.3 Результат

Введите число: 4

Введите числа: 6 2

Введите слово: abaaba

Результат: 2

Введите числа: 6 3

Введите слово: abaaba

Результат: 0

Введите числа: 36 9

Введите слово: hippopotomonstrosesquippedaliophobia

Результат: 23

Введите числа: 21 7

Введите слово: wudixiaoxingxingheclp

Результат: 16

5.1 Постановка задачи

Требуется найти оптимальное решение задачи в среде разработки Jupyter Notebook.

Гильдонг — владелец ресторана Пулькоги. Ресторан пользуется большим спросом, и часто посетители хотят забронировать столик до посещения.

Гильдонг так старается угодить всем посетителям, что он даже запомнил предпочитаемые диапазоны температуры всех посетителей! Смотря на список бронирований, он хочет удовлетворить всех посетителей, контролируя температуру ресторана.

В ресторане стоит кондиционер, у которого есть три состояния: выключенное, нагрев, и охлаждение. Когда кондиционер выключен, температура в ресторане не изменяется. Когда включен нагрев, температура увеличивается на 1 каждую минуту. Наконец, когда включено охлаждение, температура уменьшается на 1 каждую минуту. Гильдонг может переключать состояние сколько угодно раз, в любые целочисленные минуты. Кондиционер исходно выключен.

Каждый посетитель характеризуется тремя значениями: 𝑡𝑖 — момент визита (в минутах) 𝑖-го посетителя, 𝑙𝑖 — нижняя граница предпочитаемого диапазона температур, and ℎ𝑖 — верхняя граница предпочитаемого диапазона температур.

Каждый посетитель удовлетворен, если температура находится в их предпочтительном диапазоне в момент их посещения ресторана. Формально, 𝑖-й посетитель удовлетворён, если и только если в минуту 𝑡𝑖 температура находится в отрезке от 𝑙𝑖 до ℎ𝑖, включительно.

Вам дана исходная температура, список времен посещения посетителей и их предпочитаемые диапазоны температуры, помогите ему удовлетворить всех посетителей.

Входные данные Каждый тест состоит из одного или большего числа наборов входных данных. В первой строке записано количество наборов входных данных 𝑞 (1≤𝑞≤500). Затем следуют их описания.

В первой строке каждого набора входных данных записаны два целых числа 𝑛 и 𝑚 (1≤𝑛≤100, −109≤𝑚≤109), где 𝑛 это число посетителей и 𝑚 это изначальная температура ресторана.

Затем следуют 𝑛 строк. 𝑖-я из них содержит три целых числа 𝑡𝑖, 𝑙𝑖, и ℎ𝑖 (1≤𝑡𝑖≤109, −109≤𝑙𝑖≤ℎ𝑖≤109), где 𝑡𝑖 это момент визита 𝑖-го посетителя, 𝑙𝑖 это нижняя граница предпочитаемого диапазона температур, а ℎ𝑖 это верхняя граница предпочитаемого диапазона температур. Все границы диапазонов — включительны.

Посетители даны в порядке неубывания момента посещения, а текущее время — 0.

Выходные данные  
Для каждого набора входных данных выведите «YES» если возможно удовлетворить потребности всех посетителей. В противном случае, выведите «NO».

Вы можете выводить все буквы в любом регистре (верхнем или нижнем).

Пример  
входные данные  
4  
3 0  
5 1 2  
7 3 5  
10 -1 0  
2 12  
5 7 10  
10 16 20  
3 -100  
100 0 0  
100 -50 50  
200 100 100  
1 100  
99 -100 0  
выходные данные  
YES  
NO  
YES  
NO  
Примечание  
В первом наборе входных данных примера Гильдонг может контролировать кондиционер, чтобы удовлетворить всех посетителей, следующим образом:

* В 0-ю минуту, поменяйте состояние на нагрев (температура 0).
* Во 2-ю мунуту, поменяйте состояние на выключенное (температура 2)
* В 5-ю минуту, поменяйте состояние на нагрев (температура 2, 1-й посетитель уже удовлетворен).
* В 6-ю минуту, поменяте состояние на выключенное (температура 3).
* В 7-ю минуту, поменяйте состояние на охлаждение (температура 3, 2-й посетитель удовлетворен).
* В 10-ю минуту, температура будет 0, что удовлетворит последнего посетителя.
* В третьем наборе входных данных примера поменяйте состояние на выключенное в 0-ю муинуту и оставьте его таким. Тогда все посетители будут удовлетворены. Обратите внимание, что момент посещения 1-го посетителя совпадает с моментом посещения 2-го посетителя.

Во втором и четвертом наборах входных данных примера Гильдонгу придется оставить хотя бы одного посетителя неудовлетворенным.

**5.2 Решение задачи**

Код представлен ниже:

def x(a,b,c,d):

return([max(a,c),min(b,d)])

q=int(input("Введите число: "))

if not (1 <= q <= 500):

raise ValueError('нарушены условия 1 <= q <= 500')

for l in range(q):

n,m=map(int,input("Введите числа: ").split())

if not (1 <= n <= 100):

raise ValueError('нарушены условия 1 <= n <= 100')

if not (-109 <= m <= 109):

raise ValueError('нарушены условия -109 <= m <= 109')

a=m

b=m

k=0

z=0

for l in range(n):

t,l,h=map(int,input("Введите числа: ").split())

if not (1 <= t,l,h <= 100):

raise ValueError('нарушены условия 1 <= t,l,h <= 100')

if not (-109 <= t,l,h <= 109):

raise ValueError('нарушены условия -109 <= t,l,h <= 109')

a,b=x(a-(t-z),b+(t-z),l,h)

z=t

if b<a: k=1

if(a<l and b<l) or (a>h and b>h):

k=1

if k==1:print("Результат: ",'NO')

else: print("Результат: ",'YES')

5.3 Результат

Введите число: 4

Введите числа: 3 0

Введите числа: 5 1 2

Введите числа: 7 3 5

Введите числа: 10 -1 0

Результат: YES

Введите числа: 2 12

Введите числа: 5 7 10

Введите числа: 10 16 20

Результат: NO

Введите числа: 3 -100

Введите числа: 100 0 0

Введите числа: 100 -50 50

Введите числа: 200 100 100

Результат: YES

Введите числа: 1 100

Введите числа: 99 -100 0

Результат: NO

6.1 Постановка задачи

Требуется найти оптимальное решение задачи в среде разработки Jupyter Notebook.

Бинарная строка это строка, состоящая только из символов 0 и 1. Бинарная строка называется 𝑘-сбалансированной, если каждая подстрока длины 𝑘 этой бинарной строки содержит равное количество символов 0 и 1 (𝑘2 каждого).

Вам дается целое число 𝑘 и строка 𝑠, состоящая только из символов 0, 1 и ?. Вам необходимо определить, можно ли получить 𝑘-сбалансированную бинарную строку, заменив каждый символ ? в 𝑠 либо на 0, либо на 1.

Строка 𝑎 является подстрокой 𝑏, если 𝑎 может быть получена из 𝑏 удалением нескольких (возможно, ни одного или всех) символов из начала и нескольких (возможно, ни одного или всех) символов из конца.

Входные данные  
Каждый тест содержит несколько наборов входных данных. В первой строке указано количество наборов входных данных 𝑡 (1≤𝑡≤104). Описание наборов входных данных приведено ниже.

Первая строка каждого набора входных данных содержит два целых числа 𝑛 и 𝑘 (2≤𝑘≤𝑛≤3⋅105, 𝑘 чётное) — длина строки и параметр для сбалансированной бинарной строки.

Следующая строка содержит строку 𝑠 (|𝑠|=𝑛). При этом 𝑠 состоит только из 0, 1 и ?.

Гарантируется, что сумма 𝑛 по всем наборам входных данных не превышает 3⋅105.

Выходные данные  
Для каждого набора входных данных выведите YES, если мы можем заменить каждый ? в 𝑠 на 0 или 1 так, чтобы получившаяся бинарная строка была 𝑘-сбалансированной, или NO если это невозможно.

Пример  
входные данные  
9  
6 4  
100110  
3 2  
1?1  
3 2  
1?0  
4 4  
????  
7 4  
1?0??1?  
10 10  
11??11??11  
4 2  
1??1  
4 4  
?0?0  
6 2  
????00  
выходные данные  
YES  
NO  
YES  
YES  
NO  
NO  
YES  
NO  
Примечание  
В первом наборе входных данных строка уже является 4-сбалансированной бинарной строкой.

Во втором наборе входных данных строка может быть преобразована в 101.

В третьем наборе входных данных строку можно преобразовать в 0110.

В четвертом наборе входных данных строку можно преобразовать в 1100110.

**6.2 Решение задачи**

Код представлен ниже:

for \_ in range(int(input("Введите число: "))):

n,k=map(int,input("Введите числа: ").split())

if not (2 <= k <= n <= 3\*(10\*\*5)):

raise ValueError('нарушены условия 2 <= k <= n <= 3\*(10^5)')

s=input("Введите число: ")

def solve(n,k,s):

l=list(s)

for i in range(k):

t=s[i]

for j in range(i,n,k):

if s[j]!='?':

if t!='?' and s[j]!=t:

return False

t=s[j]

for j in range(i,n,k):

l[j]=t

return max(l[:k].count('1'),l[:k].count('0')) <= k//2

if solve(n,k,s):

print("Результат: ",'YES')

else:

print("Результат: ",'NO')

6.3 Результат

Введите число: 9

Введите числа: 6 4

Введите число: 100110

Результат: YES

Введите числа: 3 2

Введите число: 1?1

Результат: YES

Введите числа: 3 2

Введите число: 1?0

Результат: NO

Введите числа: 4 4

Введите число: ????

Результат: YES

Введите числа: 7 4

Введите число: 1?0??1?

Результат: YES

Введите числа: 10 10

Введите число: 11??11??11

Результат: NO

Введите числа: 4 2

Введите число: 1??1

Результат: NO

Введите числа: 4 4

Введите число: ?0?0

Результат: YES

Введите числа: 6 2

Введите число: ????00

Результат: NO

7.1 Постановка задачи

Требуется найти оптимальное решение задачи в среде разработки Jupyter Notebook.

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

Рассмотрим перестановку 𝑝 длины 𝑛. Построим граф на 𝑛 вершинах, используя перестановку следующим образом:

* Для каждого 1≤𝑖≤𝑛 найдите наибольшее значение 𝑗, для которого 1≤𝑗<𝑖 и 𝑝𝑗>𝑝𝑖, и добавьте неориентированное ребро между вершинами 𝑖 и 𝑗.
* Для каждого 1≤𝑖≤𝑛 найдите наименьшее значение 𝑗, для которого 𝑖<𝑗≤𝑛 и 𝑝𝑗>𝑝𝑖, и добавьте неориентированное ребро между вершинами 𝑖 и 𝑗
* В тех случаях, когда таких 𝑗 не существует, мы не добавляем ребер.

Также обратите внимание, что мы проводим ребра между соответствующими индексами, а не значениями в этих индексах.

Например, рассмотрим случай 𝑛=4 и 𝑝=[3,1,4,2]; здесь ребрами графа являются (1,3),(2,1),(2,3),(4,3).

Перестановка 𝑝 является циклической, если граф, построенный с использованием 𝑝, имеет хотя бы один простой цикл.

Для данного 𝑛, найдите число циклических перестановок длины 𝑛. Поскольку число может быть очень большим, выведите его по модулю 109+7.

Пожалуйста, обратитесь к разделу Примечания для формального определения простого цикла.

Входные данные  
Первая и единственная строка содержит одно целое число 𝑛 (3≤𝑛≤106).

Выходные данные  
Выведите единственное целое число 0≤𝑥<109+7, количество циклических перестановок длины 𝑛 по модулю 109+7.

Примеры  
входные данные  
4  
выходные данные  
16  
входные данные  
583291  
выходные данные  
135712853  
Примечание  
Для 𝑛=4 существует 16 циклических перестановок. [4,2,1,3] — одна из таких перестановок, она содержит цикл длины четыре: 4→3→2→1→4.

Вершины 𝑣1, 𝑣2, …, 𝑣𝑘 образуют простой цикл, если выполняются следующие условия:

𝑘≥3. 𝑣𝑖≠𝑣𝑗 для любой пары индексов 𝑖 и 𝑗. (1≤𝑖<𝑗≤𝑘) Между 𝑣𝑖 и 𝑣𝑖+1 есть ребро для всех 𝑖 (1≤𝑖<𝑘), как и между 𝑣1 и 𝑣𝑘

**7.2 Решение задачи**

Код представлен ниже:

def fact(n):

r = 1

for i in range(2, n + 1):

r = r \* i

r %= 1000000007

return r

n = int(input("Введите число: "))

if not (3 <= n <= 10\*\*6):

raise ValueError('нарушены условия 3 <= n <= 10\*\*6')

def pow(n):

r = 1

for i in range(n):

r \*= 2

r %= 1000000007

return r

print("Результат: ", (fact(n) - pow(n - 1) + 1000000007) % 1000000007)

7.3 Результат

Введите число: 583291

Результат: 135712853

8.1 Постановка задачи

Требуется найти оптимальное решение задачи в среде разработки Jupyter Notebook.

Пифагорова тройка — это тройка целых чисел (𝑎,𝑏,𝑐) таких, что можно образовать прямоугольный треугольник с длинами первого катета, второго катета и гипотенузы, равными 𝑎, 𝑏 и 𝑐 соответственно. Примером пифагоровой тройки является (3,4,5).

Вася изучает свойства прямоугольных треугольников, и он использует формулу, которая определяет, является ли некоторая тройка целых чисел пифагоровой. К сожалению, он забыл точную формулу; он помнит только, что формула была каким-то уравнением с квадратами. Поэтому он придумал следующую формулу: 𝑐=𝑎2−𝑏.

Очевидно, что это неправильная формула для проверки, является ли тройка чисел пифагоровой. Но, к удивлению Васи, его формула сработала на тройке (3,4,5): 5=32−4 поэтому по формуле Васи, это пифагорова тройка.

Когда Вася нашел правильную формулу (и понял, что его формула неверна), он задался вопросом: сколько существует троек целых чисел (𝑎,𝑏,𝑐) 1≤𝑎≤𝑏≤𝑐≤𝑛 таких, что они являются пифагоровыми как по его формуле, так и по настоящему определению? Он попросил вас посчитать количество таких троек.

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

Каждый набор состоит из одной строки, содержащей одно целое число 𝑛 (1≤𝑛≤109).

Выходные данные  
Для каждого теста выведите одно целое число — количество троек целых чисел (𝑎,𝑏,𝑐) 1≤𝑎≤𝑏≤𝑐≤𝑛 таких, чтобы они являются пифагоровыми как по настоящему определению, так и по формуле, которую придумал Вася.

Пример  
входные данные  
3  
3  
6  
9  
выходные данные  
0  
1  
1  
Примечание  
Единственная пифагорова тройка, удовлетворяющая 𝑐=𝑎2−𝑏 и 1≤𝑎≤𝑏≤𝑐≤9, является (3,4,5); поэтому ответ для 𝑛=3 равен 0, а ответ для 𝑛=6 (и для 𝑛=9) равен 1.

**8.2 Решение задачи**

Код представлен ниже:

t8=int(input("Введите число: "))

check\_t8(t8)

for i in range(t8):

n8=int(input("Введите донабор: "))

for i in range(t8):

check\_n8(n8)

print("Результат: ", calculations8(n8))

8.3 Результат

Введите число: 3

Введите донабор: 3

Результат: 0

Введите донабор: 6

Результат: 1

Введите донабор: 9

Результат: 1

9.1 Постановка задачи

Требуется найти оптимальное решение задачи в среде разработки Jupyter Notebook.

Назовем последовательность положительных чисел 𝑎0,𝑎1,...,𝑎𝑛−1 степенной последовательностью, если найдется такое положительное целое число 𝑐, что для всех 0≤𝑖≤𝑛−1, 𝑎𝑖=𝑐𝑖.

Вам дана последовательность из 𝑛 положительных чисел 𝑎0,𝑎1,...,𝑎𝑛−1, вам разрешается:

Переупорядочить последовательность (иначе говоря, выбрать перестановку 𝑝 из {0,1,...,𝑛−1} и заменить 𝑎𝑖 на 𝑎𝑝𝑖), и затем Выполнить следующую операцию любое количество раз: выбрать индекс 𝑖 и заменить 𝑎𝑖 на 𝑎𝑖−1 или 𝑎𝑖+1 (иначе говоря, уменьшить или увеличить 𝑎𝑖 на 1) за стоимость 1. Найдите минимальную стоимость, необходимую для превращения 𝑎0,𝑎1,...,𝑎𝑛−1 в степенную последовательность.

Входные данные  
В первой строке записано одно целое число 𝑛 (3≤𝑛≤105).

Во второй строке записаны 𝑛 целых чисел 𝑎0,𝑎1,...,𝑎𝑛−1 (1≤𝑎𝑖≤109).

Выходные данные  
Выведите минимальную стоимость, необходимую для превращения 𝑎0,𝑎1,...,𝑎𝑛−1 в степенную последовательность.

Примеры  
входные данные  
3  
1 3 2  
выходные данные  
1  
входные данные  
3  
1000000000 1000000000 1000000000  
выходные данные  
1999982505  
Примечание  
В первом примере сначала можно переупорядочить {1,3,2} в {1,2,3}, затем увеличить 𝑎2 до 4 за стоимость 1, чтобы получить степенную последовательность {1,2,4}.

**9.2 Решение задачи**

Код представлен ниже:

n9 = int(input("Введите число: "))

check\_n9(n9)

ar9 = list(map(int, input("Введите числа через пробел: ").split()))

check\_ar9(ar9)

print("Результат: ",calculations9(ar9,n9))

9.3 Результат

Введите число: 3

Введите числа через пробел: 1 3 2

Результат: 1

10.1 Постановка задачи

Требуется найти оптимальное решение задачи в среде разработки Jupyter Notebook.

Вы и ваш друг играете в Mortal Kombat XI. Вы пытаетесь пройти башню испытаний. Всего в башне есть 𝑛 боссов, пронумерованных от 1 до 𝑛. Тип 𝑖-го босса равен 𝑎𝑖. Если 𝑖-й босс является легким, то его тип равен 𝑎𝑖=0, иначе этот босс является сложным и его тип равен 𝑎𝑖=1.

В течение одной игровой сессии вы или ваш друг можете убить одного или двух боссов (ни вы, ни ваш друг не можете пропускать сессию, поэтому минимальное количество боссов, убитых в течение сессии, равно хотя бы одному). После сессии вашего друга начинается ваша сессия, затем опять сессия вашего друга, затем опять ваша, и так далее. Первая сессия — сессия вашего друга.

Вашему другу надо научиться играть лучше, потому что на самом деле он не может убивать сложных боссов. Чтобы убивать их, он использует очки пропуска. Одно очко пропуска может быть использовано для того, чтобы убить одного сложного босса.

Ваша задача — найти минимальное количество очков пропуска, которое ваш друг должен использовать для того, чтобы вы с вашим другом убили всех 𝑛 боссов в заданном порядке.

Например: предположим, что 𝑛=8, 𝑎=[1,0,1,1,0,1,1,1]. Тогда лучшей последовательностью действий является следующая:

* ваш друг убивает первых двух боссов, используя одно очко пропуска для первого босса;
* вы убиваете третьего и четвертого боссов;  
  ваш друг убивает пятого босса;
* вы убиваете шестого и седьмого боссов;
* ваш друг убивает последнего босса, используя одно очко пропуска, таким образом, башня проходится с использованием двух очков пропуска. 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.

Первая строка набора входных данных содержит одно целое число 𝑛 (1≤𝑛≤2⋅105) — количество боссов. Вторая строка набора входных данных содержит 𝑛 целых чисел 𝑎1,𝑎2,…,𝑎𝑛 (0≤𝑎𝑖≤1), где 𝑎𝑖 равно типу 𝑖-го босса.

Гарантируется, что сумма 𝑛 не превосходит 2⋅105 (∑𝑛≤2⋅105).

Выходные данные  
Выведите ответ на каждый набор тестовых данных: минимальное количество очков пропуска, которое ваш друг должен использовать для того, чтобы вы с вашим другом убили всех 𝑛 боссов в заданном порядке.

Пример  
входные данные  
6  
8  
1 0 1 1 0 1 1 1  
5  
1 1 1 1 0  
7  
1 1 1 1 0 0 1  
6  
1 1 1 1 1 1  
1  
1  
1  
0  
выходные данные  
2  
2  
2  
1  
0

**10.2 Решение задачи**

Код представлен ниже:

p10=int(input("Введите число: "))

check\_p10(p10)

for t in range(p10):

n10 = int(input("Введите число: "))

check\_n10(n10)

l10 = list(map(int, input("Введите числа через пробел: ").split()))

check\_item(l10)

ans10 = l10[0]

c10 = 0

print("Результат: ", calculations10(n10,c10,ans10,l10))

10.3 Результат

Введите число: 6

Введите число: 8

Введите числа через пробел: 1 0 1 1 0 1 1 1

Результат: 2

Введите число: 5

Введите числа через пробел: 1 1 1 1 0

Результат: 2

Введите число: 7

Введите числа через пробел: 1 1 1 1 0 0 1

Результат: 2

Введите число: 6

Введите числа через пробел: 1 1 1 1 1 1

Результат: 2

Введите число: 1

Введите числа через пробел: 1

Результат: 1

Введите число: 3

Введите числа через пробел: 1 1 1

Результат: 1

Вывод

В ходе данной курсовой работы, я успешно выполнила поставленные задачи и закрепила полученные знания по данной дисциплине.