**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 №13.

Выполнил: студент группы БВТ2004

Зинов Т.В.

Проверил:

Мкртчян Грач Маратович

Москва, 2022

# Задачи

## Задача №1

Назовем лесом неориентированный граф без циклов (петли и кратные рёбра также запрещены). Однажды Миша играл с лесом из n вершин. Для каждой вершины v от 0 до n - 1 он записал два числа degreev и sv, где первое число — количество вершин, смежных с вершиной v, а второе — XOR-сумма номеров вершин, смежных с v (в случае, если смежных вершин не было, он записал 0).

На следующий день Миша не смог вспомнить, какой граф у него был изначально. У Миши остались значения degreev и sv. Помогите ему найти количество ребер и сами ребра исходного графа. Гарантируется, что существует лес, которому соответствуют выписанные Мишей числа. Входные данные В первой строке находится целое число n (1 ≤ n ≤ 216), количество вершин в графе.

В i-й из последующих строк находятся числа degreei и si (0 ≤ degreei ≤ n - 1, 0 ≤ si < 216), разделенные пробелом.

Выходные данные В первой строке выведите число m, количество ребер графа.

Далее выведите m строк, каждая из которых содержит два различных числа a и b (0 ≤ a ≤ n - 1, 0 ≤ b ≤ n - 1), соответствующие ребру (a, b).

Рёбра могут быть выведены в любом порядке; вершины одного ребра также могут быть выведены в любом порядке. Примеры входные данные: 3 2 3 1 0 1 0 выходные данные: 2 1 0 2 0

## Задача №2

Дома у Фродо собираются ночевать n хоббитов. У Фродо есть n кроватей, стоящих в ряд, и m подушек (n ≤ m). Каждому хоббиту для сна нужна кровать и хотя бы одна подушка, однако, каждый хочет как можно больше подушек. Понятно, что подушки не всегда можно поделить поровну, однако любой хоббит обидится, если у него будет меньше подушек, чем у соседа, хотя бы на две.

Фродо будет спать на k-й с краю кровати. Какое максимальное число подушек он может положить себе, чтобы у каждого хоббита была хотя бы одна подушка, все подушки были розданы хоббитам, и никто из них не обиделся?

Входные данные В единственной строке находятся три целых числа n, m и k (1 ≤ n ≤ m ≤ 109, 1 ≤ k ≤ n) — число хоббитов, число подушек и номер кровати Фродо.

Выходные данные Выведите одно число — максимальное число подушек, которые Фродо может положить себе, чтобы никто не обиделся.

Примеры входные данные 4 6 2 выходные данные 2

## Задача №3

Недавно Вы купили новую умную лампу с программируемыми функциями. Первым делом Вы выставили расписание на ней. Каждый день лампа будет включать питание в момент 0 и выключать питание в момент 𝑀. Более того, данная лампа позволяет Вам записать в нее программу, следуя которой, она будет менять свое состояние (состояниями считается «свет включен» и «свет выключен»). К сожалению, в лампу уже предустановлена некоторая программа.

Лампа принимает только хорошие программы. Хорошая программа может быть представлена непустым массивом 𝑎, где 0<𝑎1<𝑎2<⋯<𝑎|𝑎|<𝑀. Все 𝑎𝑖 должны быть целыми числами. Конечно, предустановленная программа является хорошей.

Лампа выполняет заданную программу 𝑎 следующим образом: в момент 0 она включает и питание, и свет. Далее, в момент 𝑎𝑖 лампа переключает свое состояние на противоположное (если свет был включен, то он выключается, и наоборот). Свет переключается моментально, т. е., например, если переключить свет в момент времени 1 и дальше ничего не делать, то итоговое время горения лампы будет равно 1. Наконец, в момент 𝑀 лампа отключает питание независимо от того, был ли свет включен.

Так как Вы не из тех, кто читает инструкции, да и написана она на неизвестном языке, Вы (методом проб и ошибок) находите единственный способ изменить предустановленную программу. Вы можете вставить не более одного числа в программу 𝑎 таким образом, что она все еще останется хорошей. Вставка может быть осуществлена в любое место, как между любой парой соседних элементов в 𝑎, так и в начало или конец массива 𝑎.

Найдите такой способ изменить программу, что суммарное время горения лампы будет максимально. Возможно, что иногда лучше оставить программу нетронутой. (Если у лампы включен свет с момента 𝑥 по момент 𝑦, тогда лампа будет гореть время, равное 𝑦−𝑥.

Отрезки включенного света суммируются).

Входные данные Первая строка содержит два числа 𝑛 и 𝑀 (1≤𝑛≤105, 2≤𝑀≤109) — длина предустановленной программы 𝑎 и момент отключения питания лампы.

Вторая строка содержит 𝑛 чисел через пробел 𝑎1,𝑎2,…,𝑎𝑛 (0<𝑎1<𝑎2<⋯<𝑎𝑛<𝑀) — программа 𝑎.

Выходные данные Выведите единственное число — максимально возможное время горения, если Вы можете сделать не более одного изменения, описанного выше.

Примеры входные данные 3 10 4 6 7 выходные данные 8

## Задача №4

У Васи есть n предметов, которые лежат в ряд. Предметы последовательно пронумерованы числами от 1 до n таким образом, что самый левый предмет имеет номер 1, самый правый — номер n. Каждый предмет имеет свой вес, i-тый из них весит wi килограмм.

Васе нужно собрать все эти предметы, однако он не будет делать этого сам, а использует своего нового робота. У робота есть две разные руки — левая и правая. Робот может последовательно выполнять действия:

Взять левой рукой самый левый из предметов, затратив на это wi · l единиц энергии (где wi — вес самого левого предмета, l — заданный параметр), если предыдущее действие было таким же (левой рукой), то робот тратит дополнительные Ql единиц энергии. Взять правой рукой самый правый из предметов, затратив на это wj · r единиц энергии (где wj — вес самого правого предмета, r — заданный параметр), если предыдущее действие было таким же (правой рукой), то робот тратит дополнительные Qr единиц энергии. Разумеется, Вася хочет запрограммировать робота так, чтобы тот потратил как можно меньше энергии. С этой задачей он и обратился за помощью к вам. Ваша задача — найти минимальное количество энергии, которое потратит робот, чтобы собрать все предметы.

Входные данные В первой строке содержится пять целых чисел n, l, r, Ql, Qr (1 ≤ n ≤ 105; 1 ≤ l, r ≤ 100; 1 ≤ Ql, Qr ≤ 104).

Во второй строке содержится n целых чисел w1, w2, ..., wn (1 ≤ wi ≤ 100).

Выходные данные В единственной строке выведите целое число — ответ на задачу.

Примеры входные данные 3 4 4 19 1 42 3 99 выходные данные 576

## Задача №5

Будем называть неориентированный граф из n вершин p-интересным, если выполнены условия:

граф содержит ровно 2n + p ребер; граф не содержит петель и кратных ребер; для любого целого k (1 ≤ k ≤ n) любой подграф, состоящий из k вершин, содержит не более 2k + p ребер. Подграфом графа будем называть некоторое множество вершин графа и некоторое множество ребер графа. Причем множество ребер должно удовлетворять условию: оба конца каждого ребра из множества должны принадлежать выбранному множеству вершин.

Ваша задача отыскать p-интересный граф, состоящий из n вершин.

Входные данные В первой строке задано единственное целое число t (1 ≤ t ≤ 5) — количество тестовых данных. В следующих t строках задано по два целых числа: n, p (5 ≤ n ≤ 24; p ≥ 0; ) — количество вершин в графе и параметр интересности для соответствующего теста.

Гарантируется, что искомый граф существует.

Выходные данные Для каждого из t тестов выведите 2n + p строк, содержащих описание ребер p-интересного графа: i-я строка должна содержать два целых числа через пробел ai, bi (1 ≤ ai, bi ≤ n; ai ≠ bi) — две вершины, соединенные ребром в результирующем графе. Считайте, что вершины графа пронумерованы целыми числами от 1 до n.

Ответы для тестов выводите в том порядке, в котором тесты заданы во входных данных. Если существует несколько решений, разрешается вывести любое из них.

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

## Задача №6

Недавно Дима познакомился с Сашей в филателистическом магазине, и с тех пор они собирают монеты вместе. Их любимое занятие — сортировать коллекции монет. Саше нравится порядок, поэтому он хочет, чтобы монеты были расположены в ряд, причём сначала шли монеты, вышедшие из обращения, а потом монеты, всё ещё находящиеся в обращении.

Для упорядочивания монет Дима использует алгоритм, один шаг которого выглядит следующим образом:

Дима просматривает все монеты слева направо; если он видит, что i-я монета ещё в ходу, а (i + 1)-я уже вышла из обращения, то он меняет эти две монеты местами, и продолжает смотреть на монеты дальше, начиная с (i + 1)-й. Дима повторяет этот шаг до тех пор, пока не окажется, что на очередном шаге не произошло ни одного обмена. Сложностью упорядочивания Дима называет количество шагов, которые ему требуются в соответствии с процедурой, описанной выше, то есть, количество раз, которое он будет начинать просматривать монеты с начала. В частности, для уже упорядоченной исходной последовательности монет сложность упорядочивания равна единице.

Сегодня Саша в очередной раз позвал Диму в гости, чтобы предложить ему следующую игру. Сначала он выложил в ряд перед Димой n монет, все из которых вышли из обращения. Затем Саша n раз выбирает какую-то из монет, которая вышла из обращения, и заменяет её на монету, которая находится в обращении. В ходе этого процесса Саша постоянно интересуется у Димы, какова на данный момент сложность упорядочивания последовательности.

Задачу усложняет тот факт, что Диме нельзя трогать монеты, и поэтому определять сложность упорядочивания ему приходится в уме. Помогите Диме справиться с этим заданием.

Входные данные В первой строке задано целое число n (1 ≤ n ≤ 300 000) — количество монет, которые Саша выложил в ряд перед Димой.

Следующая строка содержит n целых различных чисел p1, p2, ..., pn (1 ≤ pi ≤ n) — позиции монет, если смотреть слева направо, которые Саша меняет на монеты, находящиеся в ходу. Сначала Саша заменяет монету, находящуюся на позиции p1, затем монету, находящуюся на позиции p2 и так далее.

Выходные данные Выведите n + 1 чисел a0, a1, ..., an, где a0 — сложность упорядочивания последовательности в начале, a1 — сложность упорядочивания после замены одной монеты и так далее.

Примеры входные данные 4 1 3 4 2 выходные данные 1 2 3 2 1

## Задача №7

Маленький Слоник очень любит суммы на промежутках.

В этот раз у него есть пара целых чисел l и r (l ≤ r). Маленькому Слонику нужно найти количество таких целых чисел x (l ≤ x ≤ r), что первая цифра числа x равна последней (в десятичном представлении). Например, такие числа как 101, 477474 или 9 он будет включать в ответ, в то время как 47, 253 или 1020 — нет.

Помогите ему, посчитайте описанное количество чисел x для заданной пары l и r.

Входные данные В единственной строке задана пара целых чисел l и r (1 ≤ l ≤ r ≤ 1018) — границы промежутка.

Пожалуйста, не используйте спецификатор %lld для чтения или записи 64-х битовых чисел на С++. Рекомендуется использовать потоки cin, cout или спецификатор %I64d.

Выходные данные В единственной строке выведите одно целое число — ответ на задачу.

Примеры входные данные 2 47 выходные данные 12

## Задача №8

После возвращения из армии Макес получил в подарок массив a, состоящий из n целых положительных чисел. Так как он очень давно не решал задачи, его заинтересовал следующий вопрос: сколько существует таких упорядоченных троек (i,  j,  k), что i < j < k, а ai·aj·ak — минимально. Помогите ему в этом!

Входные данные В первой строке входных данных задано целое положительное число n (3 ≤ n ≤ 105) — количество элементов в массиве a. Во второй строке задано n положительных целых чисел ai (1 ≤ ai ≤ 109) — элементы массива.

Выходные данные Выведите единственное число — количество упорядоченных троек (i,  j,  k), что i,  j и k — попарно различны, а ai·aj·ak — минимально.

Примеры входные данные 4 1 1 1 1 выходные данные 4

## Задача №9

Дана строка S, состоящая из N символов. Требуется найти количество упорядоченных пар целых чисел i и j таких, что:

1 ≤ i, j ≤ N

S[i] = S[j], то есть i-ый символ строки S равен j-ому.

Входные данные Единственная строка входа содержит S, состоящую из строчных букв латинского алфавита и цифр. Гарантируется, что строка S непуста и её длина не превосходит 105.

Выходные данные Выведите одно число — количество пар i и j с требуемым свойством. Пары (x, y) и (y, x) следует считать различными, т. е. считаются упорядоченные пары.

Примеры входные данные great10 выходные данные 7 входные данные aaaaaaaaaa выходные данные 100

## 1.10Задача №10

Алиса и Боб играют в игру на прямой с 𝑛 ячейками. Есть 𝑛 ячеек, пронумерованных от 1 до 𝑛. Для каждого 𝑖 от 1 до 𝑛−1 ячейки 𝑖 и 𝑖+1 являются смежными.

У Алисы изначально есть фишка в какой-то ячейке на прямой, а Боб пытается угадать, где она находится.

Боб спрашивает последовательность номеров ячеек в таком порядке: 𝑥1,𝑥2,…,𝑥𝑘. В 𝑖-м вопросе Боб спрашивает Алису, находится ли ее фишка в ячейке 𝑥𝑖. То есть, Алиса ответит либо «нет», либо «да» на каждый вопрос Боба.

Не более одного раза в этом процессе, до или после ответа на вопрос, Алиса может переместить свою фишку из своей текущей ячейки в некоторую смежную ячейку. Алиса действует так, чтобы она могла ответить «нет» на все вопросы Боба.

Обратите внимание, что Алиса даже может переместить свою фигурку, прежде чем ответить на первый вопрос или после того, как ответит на последний вопрос. Алиса также может вообще не передвигать ее.

Вам дано число 𝑛 и вопросы Боба 𝑥1,…,𝑥𝑘. Вы хотели бы посчитать количество сценариев, которые позволяют Алисе ответить «нет» на все вопросы Боба.

Пусть (𝑎,𝑏) обозначает сценарий, в котором Алиса начинает в ячейке 𝑎 и заканчивает в ячейке 𝑏. Два сценария (𝑎𝑖,𝑏𝑖) и (𝑎𝑗,𝑏𝑗) различны, если 𝑎𝑖≠𝑎𝑗 или 𝑏𝑖≠𝑏𝑗.

Входные данные Первая строка содержит два целых числа 𝑛 и 𝑘 (1≤𝑛,𝑘≤105) — количество ячеек и количество вопросов Боба.

Вторая строка содержит 𝑘 целых чисел 𝑥1,𝑥2,…,𝑥𝑘 (1≤𝑥𝑖≤𝑛) — вопросы Боба.

Выходные данные Выведите единственное целое число — количество сценариев, которые позволяют Алисе ответить «нет» на все вопросы Боба.

Примеры входные данные 5 3 5 1 4 выходные данные 9

# 2.Выполнение

## 2.1Листинг кода задачи №1

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

**assert** 1 **<** n **<** 216, "Входные данные не удовлетворяют условию"

v **=** []

deg1 **=** []

degsum **=** 0

has\_v **=** [**True**]**\***n

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

d, s **=** map(int, input()**.**split())

degsum **+=** d

**if** d **==** 1: deg1**.**append(i)

**if** d **==** 0: has\_v[i] **=** **False**

v**.**append([d, s])

**assert** 0**<**d**<**n**-**1,"Входные данные не удовлетворяют условию"

**assert** 0**<=**s**<=**216,"Входные данные не удовлетворяют условию"

edge **=** []

**while** deg1 **!=** []:

f **=** deg1**.**pop()

**if** has\_v[f] **is** **False**: **continue**

v[f][0] **-=** 1

has\_v[f] **=** **False**

t **=** v[f][1]

edge**.**append([f, t])

v[t][0] **-=** 1

v[t][1] **^=** f

**if** v[t][0] **==** 1:

deg1**.**append(t)

**elif** v[t][0] **==** 0:

has\_v[t] **=** **False**

print("---result------")

print(degsum**//**2)

**for** f, t **in** edge:

print(f, t)

## 2.2Листинг кода задачи №2

n, m, k **=** map(int, input()**.**split())

**assert** 1**<=**n**<=**m**<=**10**\*\***9 ,"Входные данные не удовлетворяют условию"

**assert** 1**<=**k**<=**n,"Входные данные не удовлетворяют условию"

m **-=** n

d **=** 0

k **-=** 1

result **=** 1

**while** m **>** 1 **and** d **!=** max(k, n **-** k **-** 1):

result **+=** 1

m **-=** 1

left **=** min(d, k)

right **=** min(d, n **-** k **-** 1)

d **+=** 1

m **-=** left

m **-=** right

result **+=** m **//** n

print("---result------")

print(result)

## 2.3Листинг кода задачи №3

m,n**=**map(int,input()**.**split())

a**=**[0]**+**[int(x) **for** x **in** input()**.**split()]**+**[n]

s**=**k**=**t**=**0

abool**=False**

**assert** 1**<=**n**<=**10**\*\***5,"Входные данные не удовлетворяют условию"

**for** i **in** range (len(a)**-**1):

**for** j **in** range (i**+**1,len(a)):

**assert** a[i]**<**a[j],"Входные данные не удовлетворяют условию"

**for** i **in** range(1,len(a),2):

ti**=**a[i]**-**a[i**-**1]

t**+=**ti

s**=**t

**for** i **in** range(2,len(a),2):

k**+=**a[i**-**1]**-**a[i**-**2]

**if** a[i]**>**a[i**-**1]**+**1:

ki**=**t**-**k

s**=**max(s,k**+**n**-**a[i**-**1]**-**1**-**ki)

print("---result------")

print(s)

## 2.4Листинг кода задачи №4

n,l,r,ql,qr **=** map(int,input()**.**split())

wbool**=False**

**assert** 1**<=**n**<=**10**\*\***5,"Входные данные не удовлетворяют условию"

**assert** 1**<=**l**<=**100,"Входные данные не удовлетворяют условию"

**assert** 1**<=**ql,"Входные данные не удовлетворяют условию"

**assert** qr**<=**104,"Входные данные не удовлетворяют условию"

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

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

**assert** 1**<=**w[i]**<=**100,"Входные данные не удовлетворяют условию"

s **=** [0, **\***w]

**for** i **in** range(1, n**+**1): s[i] **+=** s[i**-**1]

c\_min **=** s[n]**\***r **+** (n**-**1)**\***qr

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

cc **=** l**\***s[i] **+** (s[n]**-** s[i])**\***r

**if** 2**\***i **<** n:

cc **+=** (n**-**2**\***i**-**1)**\***qr

**elif** 2**\***i **>** n:

cc **+=** (i**-**(n**-**i) **-**1)**\***ql

c\_min **=** min(c\_min, cc)

print("---result------")

print(c\_min)

## 2.5Листинг кода задачи №5

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

**assert** 1**<=**t**<=**5

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

n, p **=** map(int, input()**.**split())

**assert** 5**<=**n**<=**24,"Входные данные не удовлетворяют условию"

**assert** p**>=**0,"Входные данные не удовлетворяют условию"

i, j **=** 1, 2

print("---result------")

**for** k **in** range(2 **\*** n **+** p):

print(i, j)

j **+=** 1

**if** j **>** n:

i **+=** 1

j **=** i **+** 1

## 2.6Листинг кода задачи №6

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

**assert** 1**<=**n**<=**300000,"Входные данные не удовлетворяют условию"

plist **=** list(map(int, input()**.**split(' ')))

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

**assert** 1**<=**plist[i]**<=**n,"Входные данные не удовлетворяют условию"

pos **=** n

a **=** [0 **for** i **in** range(n**+**1)]

res **=** 1

ans **=** [1]

**for** x **in** plist:

a[x] **=** 1

res **+=** 1

**while** a[pos]**==**1:

pos **-=** 1

res **-=** 1

ans**.**append(res)

print("---result------")

print (' '**.**join(map(str, ans)))

## 2.7Листинг кода задачи №7

**def** find(x):

**if**(x**<**10):

**return** x

ans**=**x**//**10**-**1

ans**+=**9

**if**(str(x)[0]**<=**str(x)[**-**1]):

ans**+=**1

**return** ans

l,r**=**map(int,input()**.**split())

**if** l**<**1 **or** l**>**r **or** r**>**1018:

**assert** **False**,"Входные данные не удовлетворяют условию"

**else**:

print("---result------")

print( find(r) **-** find(l**-**1))

## 2.8Листинг кода задачи №8

a**=**int(input())

**assert** 3**<=**a**<=**10**\*\***5,"Входные данные не удовлетворяют условию"

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

s**=**[]

**assert** a**==**len(b),"Входные данные не удовлетворяют условию"

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

**assert** 1**<=**b[i]**<=**10**\*\***9,"Входные данные не удовлетворяют условию"

s**.**append(b[i])

s**.**sort()

f1**=**b**.**count(s[0])

b**.**remove(s[0])

f2**=**b**.**count(s[1])

b**.**remove(s[1])

f3**=**b**.**count(s[2])

b**.**remove(s[2])

f4**=**f1**\***f2**\***f3

**if** s[1]**==**s[2] **and** s[2]**==**s[0]:

f4**=**f4**//**6

**if** s[1]**==**s[2] **and** s[2]**!=**s[0]:

f4**=**f4**//**2

**if** s[1]**==**s[0] **and** s[0]**!=**s[2]:

f4**=**f4**//**2

**if** s[2]**==**s[0] **and** s[1]**!=**s[2]:

f4**=**f4**//**2

print("---result------")

print(f4)

## 2.9Листинг кода задачи №9

a**=**input()

**assert** len(a)**<=**105,"Входные данные не удовлетворяют условию"

b**=**0

c**=**set(a)

**for** i **in** c:

b**+=**a**.**count(i)**\*\***2

print("---result------")

print(b)

## 2.10Листинг кода задачи №10

n, k **=** list(map(int, input()**.**split()))

**if** n**<**1 **or** k**>**105:

**assert** **False**,"Входные данные не удовлетворяют условию"

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

**for** a **in** x:

**assert** 1**<=**a**<=**n,"Входные данные не удовлетворяют условию"

ml **=** []

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

ml**.**append([0])

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

ml[x[i]**-**1] **+=** [i**+**1]

c **=** 0

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

**if** ml[i] **==** [0]:

c **+=** 1

**if** i **!=** 0:

**if** ml[i] **==** [0] **or** ml[i**-**1] **==** [0] **or** min(ml[i][1:]) **>** max(ml[i**-**1]):

c **+=** 1

**if** i **!=** n**-**1:

**if** ml[i] **==** [0] **or** ml[i**+**1] **==** [0] **or** min(ml[i][1:]) **>** max(ml[i**+**1]):

c **+=** 1

print(c)

# 3.Вывод

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