**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”**

### **Отчет по курсовой работе**

### 

### по дисциплине “Структуры и алгоритмы обработки данных”

Выполнил: студент гр. БВТ2005

Голубков А.Ю.

Проверил: Симонов С.Е.

Москва 2022 г.

### **Задача 1**

На координатной плоскости стоит робот. Изначально он расположен в точке (0,0). Его путь описывается строкой 𝑠 длины 𝑛, состоящей из символов 'L', 'R', 'U', 'D'.

Каждый из этих символов соответствует некоторому ходу:

'L' (влево): робот перемещается из точки (𝑥,𝑦) в точку (𝑥−1,𝑦);

'R' (вправо): робот перемещается из точки (𝑥,𝑦) в точку (𝑥+1,𝑦);

'U' (вверх): робот перемещается из точки (𝑥,𝑦) в точку (𝑥,𝑦+1);

'D' (вниз): робот перемещается из точки (𝑥,𝑦) в точку (𝑥,𝑦−1).

Компания, которая создала этого робота, попросила вас оптимизировать путь робота некоторым образом. Чтобы сделать это, вы можете удалить любую непустую подстроку пути. Но эта компания не хочет, чтобы покупатели заметили перемену в поведении робота. Это означает, что если до оптимизации робот заканчивал свой путь в точке (𝑥𝑒,𝑦𝑒), то после оптимизации (т.е. удаления некоторой единственной подстроки из 𝑠) робот тоже должен заканчивать свой путь в точке (𝑥𝑒,𝑦𝑒).

Эта оптимизация — малобюджетный проект, поэтому вам нужно удалить кратчайшую возможную непустую подстроку, чтобы оптимизировать путь робота так, что конечная точка его пути не изменится. Возможна ситуация, когда вы не можете оптимизировать путь. Кроме того, возможно, что после оптимизации полученный путь станет пустой строкой (т.е. удаленная подстрока — это вся строка 𝑠).

Напомним, что подстрока строки 𝑠 — это такая строка, которая может быть получена из 𝑠 с помощью удаления некоторого числа символов (возможно, нулевого) из префикса и некоторого количества символов (возможно, нулевого) из суффикса. Например, подстроками «LURLLR» являются «LU», «LR», «LURLLR», «URL», но не «RR» и «UL».

Вам нужно ответить на 𝑡 независимых наборов входных данных.

Входные данные

Первая строка входных данных содержит одно целое число 𝑡 (1≤𝑡≤1000) — количество наборов входных данных.

Следующие 2𝑡 строк описывают наборы входных данных. Каждый набор задан двумя строками. Первая строка набора содержит одно целое число 𝑛 (1≤𝑛≤2⋅105) — длину пути робота. Вторая строка набора содержит одну строку 𝑠, состоящую из 𝑛 символов 'L', 'R', 'U', 'D' — путь робота.

Гарантируется, что сумма всех 𝑛 по всем наборам входных данных не превосходит 2⋅105 (∑𝑛≤2⋅105).

Выходные данные

Для каждого набора входных данных выведите ответ на него. Если вы не можете удалить такую непустую подстроку, что конечная точка пути робота не изменится, выведите -1. В противном случае выведите два целых числа 𝑙 и 𝑟 такие, что 1≤𝑙≤𝑟≤𝑛 — границы подстроки, которую вы удаляете. Значение 𝑟−𝑙+1 должно быть минимальным возможным. Если существует несколько ответов, выведите любой из них.

Пример

входные данные

4

4

LRUD

4

LURD

5

RRUDU

5

LLDDR

выходные данные

1 2

1 4

3 4

-1

**Решение:**

t = int(input('Введите количество наборов:' + '\t\t')); print()

while ((t < 1) or (t > 1000)):

t = int(input('Необходимо ввести целое число от 1 до 1000!'))

for j in range (1, t + 1):

n = int(input('Введите длину пути робота ' + str(j) + ' набора:' + '\t'))

while ((n < 1) or (n > 2\*105)):

n = int(input('Необходимо ввести целое число от от 1 до 2\*105 !'))

s = input('Введите строку ' + str(j) + ' набора:' + '\t\t')

while (len(s) != n):

s = input('Количество символов должно быть равно длине пути робота!')

x = 0; y = 0

l = 0; r = 10000000

m = {str(x)+str(y):1}

for i in range (n):

if s[i] == 'R':

x += 1

if s[i] == 'L':

x -= 1

if s[i] == 'U':

y += 1

if s[i] == 'D':

y -= 1

try:

if m[str(x)+str(y)]:

if(r-l+1 > i+2-m[str(x)+str(y)]):

r = i+1; l = m[str(x)+str(y)]

except KeyError:

pass

m[str(x)+str(y)]=i+2;

if(r==10000000):

print('Результат:' + '\t\t\t\t' + '-1' + '\n')

else:

print('Результат:' + '\t\t\t\t' + str(l) + ' ' + str(r) + '\n')

### **Задача 2**

Для некоторой 01-строки 𝑠 (то есть такой, что каждый символ 𝑠𝑖 — это либо '0', либо '1') выписали все пары подряд идущих символов (все подстроки длины 2). Для каждой пары (подстроки длины 2) посчитали количество символов '1' (единиц) в ней.

Вам заданы три числа:

𝑛0 — количество пар подряд идущих символов (подстрок), в которых 0 единиц;

𝑛1 — количество пар подряд идущих символов (подстрок), в которых 1 единица;

𝑛2 — количество пар подряд идущих символов (подстрок), в которых 2 единицы.

Например, для строки 𝑠=«1110011110» были бы выписаны следующие подстроки:

«11», «11», «10», «00», «01», «11», «11», «11», «10». Таким образом, 𝑛0=1, 𝑛1=3, 𝑛2=5.

Ваша задача — по заданным 𝑛0,𝑛1,𝑛2 восстановить любую подходящую 01-строку (бинарную строку) 𝑠. Гарантируется, что хотя бы одно из чисел 𝑛0,𝑛1,𝑛2 отлично от 0. Также гарантируется, что решение существует.

Входные данные

В первой строке записано целое число 𝑡 (1≤𝑡≤1000) — количество наборов входных данных в тесте. Далее во входных данных записаны сами наборы.

Каждый набор состоит из одной строки, которая содержит три целых числа 𝑛0,𝑛1,𝑛2 (0≤𝑛0,𝑛1,𝑛2≤100; 𝑛0+𝑛1+𝑛2>0). Гарантируется, что ответ для заданных 𝑛0,𝑛1,𝑛2 существует.

Выходные данные

Выведите 𝑡 строк. Каждая из строк должна содержать ответ в виде 01-строки (бинарной строки) на соответствующий набор входных данных. Если возможных ответов несколько, выведите любой из них.

Пример

входные данные

7

1 3 5

1 1 1

3 9 3

0 1 0

3 1 2

0 0 3

2 0 0

выходные данные

1110011110

0011

0110001100101011

10

0000111

1111

000

**Решение:**

t = int(input('Введите количество наборов:' + '\t'))

while ((t < 1) or (t > 1000)):

print('Необходимо ввести целое число от 1 до 1000!'); print()

t = int(input('Введите количество наборов:' + '\t'))

print()

for i in range(t):

n0, n1, n2 = input('Введите ' + str(i+1) + ' набор:' + '\t\t').split()

n0 = int(n0); n1 = int(n1); n2 = int(n2)

while (((n0<0)or(n0>100)) or ((n1<0)or(n1>100)) or ((n2<0)or(n2>100))):

n0, n1, n2 = input('Введите числа от 0 до 100!' + '\t').split()

n0 = int(n0); n1 = int(n1); n2 = int(n2)

n = '1'

for j in range(n2):

n += '1'

n += '0'

for j in range(n0):

n += '0'

for j in range(n1//2):

n += '10'

if n1 % 2 == 0:

n = n [:-1]

print('Результат:' + '\t\t\t' + n + '\n')

### **Задача 3**

У вашего одноклассника, которого вы не очень любите за его занудство, но уважаете за его ум, были обнаружены две строки: строка 𝑠 длины 𝑛 и строка 𝑡 длины 𝑚.

Последовательность индексов 𝑝1,𝑝2,…,𝑝𝑚, где 1≤𝑝1<𝑝2<…<𝑝𝑚≤𝑛, называется хорошей, если 𝑠𝑝𝑖=𝑡𝑖 для всех 𝑖 от 1 до 𝑚. Шириной последовательности называется величина max1≤𝑖<𝑚(𝑝𝑖+1−𝑝𝑖).

Помогите однокласснику найти хорошую последовательность индексов с максимальной шириной. Одноклассник обещал вам, что строки 𝑠 и 𝑡 таковы, что хотя бы одна хорошая последовательность точно существует.

Входные данные

Первая строка входных данных содержит два числа 𝑛 и 𝑚 (2≤𝑚≤𝑛≤2⋅105) — длины строк 𝑠 и 𝑡 соответственно.

Во второй строке входных данных задана строка 𝑠, состоящая из 𝑛 строчных букв латинского алфавита.

В третьей строке задана строка 𝑡, состоящая из 𝑚 строчных букв латинского алфавита.

Гарантируется, что существует хотя бы одна хорошая последовательность индексов.

Выходные данные

Выведите одно число — максимальную ширину хорошей последовательности.

Примеры

входные данные

5 3

abbbc

abc

выходные данные

3

входные данные

5 2

aaaaa

aa

выходные данные

4

входные данные

5 5

abcdf

abcdf

выходные данные

1

входные данные

2 2

ab

ab

выходные данные

1

**Решение:**

n, m = input('Введите длины строк s и t соответственно:' + '\t').split()

n = int(n); m = int(m)

while ((n < 2) or (n > 2\*105) or (m < 2) or (m > 2\*105) or (m > n)):

print('Данные введены некорректно!')

n, m = input('Введите длины строк s и t соответственно:' + '\t').split()

s = input('Введите строку s:' + '\t\t\t\t')

while(len(s) != n):

print('Длина строки s должна соответствовать введённой!')

s = input('Введите строку s:' + '\t\t\t\t')

s += ' '

t = input('Введите строку t:' + '\t\t\t\t')

while(len(t) != m):

print('Длина строки t должна соответствовать введённой!')

t= input('Введите строку t:' + '\t\t\t\t')

good = []; no\_repeat = []; j = 0

for i in range(n):

if (s[i] in no\_repeat) and (s[i+1] == s[i]):

continue

no\_repeat += s[i]

if s[i] == t[j]:

good.append(i)

j += 1

max = 1

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

if (good[i+1] - good[i]) > max:

max = good[i+1] - good[i]

print ('Результат:' + '\t\t\t\t\t' + str(max))

### **Задача 4**

В Семи Королевствах n городов и n - 1 дорога, каждая дорога соединяет два города, и возможно достичь каждый город из каждого, передвигаясь по дорогам.

Теон и Яра Грейджой начинают путешествовать на лошади по дорогам из первого города. Из-за тумана они не видят, куда лошадь их везет. Когда лошадь входит в город (в том числе в первый), она направляется в один из городов, соединенных с ним. Но это странная лошадь, поэтому она пойдет только в город, в котором до этого не была. Она идет равновероятно в каждый из таких городов и останавливается, если таких городов нет.

Пусть длина каждой дороги 1. Путешествие начинается в городе 1. Чему равно математическое ожидание длины их путешествия? Формально, математическое ожидание — это ожидаемое (то есть среднее) значение.

Входные данные

Первая строка содержит одно целое число n (1 ≤ n ≤ 100000) — количество городов.

Далее следует n - 1 строк. i-я из них содержит два целых числа ui и vi (1 ≤ ui, vi ≤ n, ui ≠ vi) — номера городов, соединенных i-й дорогой.

Гарантируется, что возможно достичь каждый город из каждого, передвигаясь по дорогам.

Выходные данные

Выведите одно число — математическое ожидание длины путешествия. Путешествие начинается в городе 1.

Ваш ответ будет считаться правильным, если его абсолютная или относительная точность не превосходит 10 - 6.

А именно, если ваш ответ равен a, а ответ жюри равен b, то ваш ответ будет зачтен, если .

Примеры

входные данные

4

1 2

1 3

2 4

выходные данные

1.500000000000000

входные данные

5

1 2

1 3

3 4

2 5

выходные данные

2.000000000000000

**Решение:**

n = int(input('Введите количество городов:' + '\t\t\t'))

while ((n < 1) or (n > 100000)):

print('Необходимо ввести число от 1 до 100000')

n = int(input('Введите количество городов:' + '\t'))

r = [] #roads

for i in range (n - 1):

b = input('Введите номера городов, соединённых ' + str(i + 1) + ' дорогой' + '\t').split()

b = [int(item) for item in b]

while((len(b) != 2) or (b[0] < 1) or (b[0] > n) or (b[1] < 1) or (b[1] > n) or (b[0] == b[1])):

print('Данные о дороге введены некорректно!')

b = input('Введите номера городов, соединённых ' + str(i + 1) + ' дорогой' + '\t').split()

b = [int(item) for item in b]

r.append(b)

count = 1; trip = 0; res = []; copy = r; restart = True; exit = False

while ((len(r) != 0) and (restart)):

restart = False; exit = False

for i in range(len(r)):

if ((r[i][0] == count) or (r[i][1] == count)):

if (r[i][0] > r[i][1]):

count = r[i][0]

else:

count = r[i][1]

trip += 1

r.pop(i)

restart = True

exit = True

break

if (exit):

continue

res.append(trip); count = 1; trip = 0; restart = True

res.append(trip)

sum = 0

for i in range (len(res)):

sum += res[i]

print('Математическое ожидание длины путешествия' + '\t' + str(sum/len(res)))

### **Задача 5**

Дарк собирается посетить день рождения Мотарака. Дарк решил, что он хочет подарить Мотараку массив 𝑎, состоящий из 𝑛 неотрицательных целых чисел.

Дарк создал этот массив еще 1000 лет назад, поэтому некоторые его элементы пропали. Дарк знает, что Мотарак не любит, когда у двух соседних элементов большая абсолютная разница. У него не так много времени, поэтому он хочет выбрать целое число 𝑘 (0≤𝑘≤109) и заменить все пропущенные элементы в массиве 𝑎 числом 𝑘.

Обозначим за 𝑚 максимальную абсолютную разницу между всеми парами соседних элементов (то есть максимальное значение |𝑎𝑖−𝑎𝑖+1| по всем 1≤𝑖≤𝑛−1) в массиве 𝑎 после того, как Дарк заменит все пропавшие элементы на число 𝑘.

Дарк хочет выбрать число 𝑘 так, что 𝑚 будет минимально. Можете ли вы помочь ему?

Входные данные

Входные данные состоят из нескольких тестовых случаев. Первая строка содержит единственное целое число 𝑡 (1≤𝑡≤104) — количество тестовых случаев. Далее следует описание тестовых случаев в следующем формате.

В первой строке описания каждого тестового случае содержится одно целое число 𝑛 (2≤𝑛≤105) — размер массива 𝑎.

Во второй строке описания каждого тестового случая содержится 𝑛 целых чисел 𝑎1,𝑎2,…,𝑎𝑛 (−1≤𝑎𝑖≤109). Если 𝑎𝑖=−1, то 𝑖-е число пропало. Гарантируется, что хотя бы одно число пропало в массиве в каждом тестовом случае.

Гарантируется, что сумма 𝑛 по всем тестовым случаям не превосходит 4⋅105.

Выходные данные

Выведите ответы для каждого тестового случая в следующем формате:

Вы должны вывести два целых числа, минимальное возможное значение 𝑚 и целое число 𝑘 (0≤𝑘≤109), которое делает максимальную абсолютную разницу между соседними элементами массива 𝑎 равной 𝑚.

Обратите внимание, что после замены всех пропавших элементов на 𝑘 максимальная абсолютная разница между всеми парами соседних элементов должна стать равной 𝑚.

Если существует более, чем одно подходящее 𝑘, вы можете вывести любое из них.

Пример

входные данные

7

5

-1 10 -1 12 -1

5

-1 40 35 -1 35

6

-1 -1 9 -1 3 -1

2

-1 -1

2

0 -1

4

1 -1 3 -1

7

1 -1 7 5 2 -1 5

выходные данные

1 11

5 35

3 6

0 42

0 0

1 2

3 4

**Решение:**

t = int(input('Введите количество тестовых случаев:' + '\t'))

while ((t < 1) or (t > 104)):

print('Необходимо ввести целое число от 1 до 104!')

t = int(input('Введите количество тестовых случаев:' + '\t'))

print()

for j in range(t):

n = int(input('Введите размер ' + str(j+1) + ' массива:' + '\t\t'))

while ((n < 2) or (n > 105)):

print ('Размер массива должен быть от 2 до 105')

n = int(input('Введите размер ' + str(j+1) + ' массива:' + '\t\t'))

check = True

while(check):

a = input('Введите числа в этом массиве:' + '\t\t').split()

a = [int(item) for item in a]

check = False

for i in range (len(a)):

if ((a[i] < -1) or (a[i] > 109)):

print ('Элемент должен находится в промежутке от -1 до 109!')

check = True

break

a.insert(0, 1000005)

Max=0; Min=1000000000; res=0;

for i in range(2, n):

if(a[i] == -1):

Max = max(Max,a[i-1])

if (a[i-1] != -1):

Min = min(Min,a[i-1])

for i in range(1, n):

if(a[i] == -1):

Max=max(Max,a[i+1])

if (a[i+1] != -1):

Min = min(Min,a[i+1])

for i in range(1, n):

if ((a[i]!=-1) and (a[i+1]!=-1)):

res = max(res,abs(a[i+1]-a[i]))

res=max(res,(Max-Min+1)//2)

print ('Результат, т.е числа m, k' + '\t\t' + str(res) + ' ' + str((Max+Min)//2) + '\n')

### **Задача 6**

Вам задан массив 𝑎, состоящий из 𝑛 целых чисел.

Вы можете удалить не более одного элемента из этого массива. Таким образом, после этой операции длина массива равна либо 𝑛−1, либо 𝑛.

Ваша задача — посчитать максимально возможную длину строго возрастающего последовательного подмассива получившегося массива.

Напомним, что последовательным подмассивом 𝑎 с индексами от 𝑙 до 𝑟 называется 𝑎[𝑙…𝑟]=𝑎𝑙,𝑎𝑙+1,…,𝑎𝑟. Массив 𝑎[𝑙…𝑟] называется строго возрастающим, если 𝑎𝑙<𝑎𝑙+1<⋯<𝑎𝑟.

Входные данные

Первая строка входных данных содержит одно целое число 𝑛 (2≤𝑛≤2⋅105) — количество элементов в 𝑎.

Вторая строка входных данных содержит 𝑛 целых чисел 𝑎1,𝑎2,…,𝑎𝑛 (1≤𝑎𝑖≤109), где 𝑎𝑖 равно 𝑖-му элементу 𝑎.

Выходные данные

Выведите одно целое число — максимально возможную длину строго возрастающего последовательного подмассива массива 𝑎 после удаления не более одного элемента.

Примеры

входные данные

5

1 2 5 3 4

выходные данные

4

входные данные

2

1 2

выходные данные

2

входные данные

7

6 5 4 3 2 4 3

выходные данные

2

**Решение:**

n = int(input('Введите количество элементов массива:' + '\t'))

while ((n < 2) or (n > 2\*105)):

print('Необходимо ввести число от 2 до 2\*105!')

n = int(input('Введите количество элементов массива:' + '\t'))

check = True

while(check):

a = input('Введите числа в этом массиве:' + '\t\t').split()

a = [int(item) for item in a]

check = False

for i in range (len(a)):

if ((a[i] < 1) or (a[i] > 109)):

print ('Элемент должен находится в промежутке от 1 до 109!')

check = True

break

max = 1; t = 1; skip = False

for i in range(n-1):

if skip:

skip = False

continue

if (a[i+1] == a[i] + 1):

t += 1

else:

try:

if (a[i+2] == a[i] + 1):

t += 1

skip = True

continue

except IndexError: pass

if (t > max):

max = t

t = 1

if (t > max):

max = t

print('\n' + 'Результат' + '\t\t\t\t' + str(max))

### **Задача 7**

У Коли есть массив целых чисел 𝑎1,𝑎2,…,𝑎𝑛. Числа в массиве могут быть и положительными, и отрицательными, но так как Коля не любит число 0, то таких чисел в его массиве нет.

Коле не нравится, что сумма элементов на некоторых подотрезках его массива может быть равна 0. Под подотрезком массива следует понимать некоторую непустую последовательность подряд идущих элементов массива.

Вы должны помочь Коле и сделать так, чтобы в его массиве не было подотрезков с суммой элементов равной 0. Для этого вы можете вставлять в его массив произвольные целые числа между любыми соседними элементами массива (числа могут быть любые: положительные, отрицательные, 0, любые по абсолютной величине, даже не представимые в стандартных типах данных во многих языках программирования).

Определите минимальное количество произвольных целых чисел, которые нужно добавить в массив Коли таким образом, чтобы после этого в массиве не было подотрезков с суммой элементов равной 0.

Входные данные

В первой строке следует целое число 𝑛 (2≤𝑛≤200000) — количество элементов в массиве Коли.

Во второй строке следует 𝑛 целых чисел 𝑎1,𝑎2,…,𝑎𝑛 (−109≤𝑎𝑖≤109,𝑎𝑖≠0) — описание массива Коли.

Выходные данные

Выведите минимальное количество произвольных целых чисел, которые нужно добавить в массив Коли таким образом, чтобы после этого в массиве не было подотрезков с суммой элементов равной 0.

Примеры

входные данные

4

1 -5 3 2

выходные данные

1

входные данные

5

4 -2 3 -9 2

выходные данные

0

входные данные

9

-1 1 -1 1 -1 1 1 -1 -1

выходные данные

6

входные данные

8

16 -5 -11 -15 10 5 4 -4

выходные данные

3

**Решение**:

n = int(input('Введите количество элементов массива:' + '\t'))

while ((n < 2) or (n > 200000)):

print('Необходимо ввести число от 2 до 200000!')

n = int(input('Введите количество элементов массива:' + '\t'))

check = True

while(check):

a = input('Введите числа в этом массиве:' + '\t\t').split()

a = [int(item) for item in a]

check = False

for i in range (len(a)):

if ((a[i] < -109) or (a[i] > 109) or a[i] == 0):

print ('Элемент должен находится в промежутке от -109 до 109, за исключением нуля!')

check = True

break

k = 0;

def func(a, j):

global k

sum = a[j]

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

sum += a[i+1]

if (sum == 0):

a.insert(i+1, 11000000)

k += 1

j = 0

while (j != (len(a) - 1)):

func(a, j)

j += 1

print('\n' + 'Результат' + '\t\t\t\t' + str(k))

### **Задача 8**

В линию выстроены 𝑛 монстров, пронумерованных числами с 1 по 𝑛. У 𝑖-го монстра есть ℎ𝑖 очков здоровья (hp). Вы можете атаковать с силой, равной 𝑎 hp, а ваш соперник может атаковать с силой, равной 𝑏 hp.

Вы и ваш соперник сражаетесь с этими монстрами. Сначала вы и ваш соперник идете к первому монстру и сражаетесь с ним до тех пор, пока он не будет мертв, затем вы и ваш соперник отправляетесь ко второму монстру и сражаетесь с ним до тех пор, пока и он не будет мертв, и так далее. Смерть монстра наступает, когда значение его hp становится меньшим или равным 0.

Сражение с монстром происходит поочередно.

Вы атакуете монстра с силой 𝑎 hp. Если он погибает после вашего удара, вы получаете одно очко и затем вы и соперник переходите к следующему монстру. Ваш соперник атакует монстра с силой 𝑏 hp. Если монстр погибает после этой атаки, никто не получает очко и вы с соперником переходите к следующему монстру.

У вас есть некоторая секретная техника, с помощью которой можно заставить вашего соперника пропустить свой ход. Вы можете использовать эту технику не более, чем 𝑘 раз всего (например, если есть два монстра и 𝑘=4, то вы можете использовать эту технику 2 раза во время сражения с первым монстром и 1 один раз — со вторым, но не 2 раза с первым монстром и 3 раза со вторым).

Ваша задача — определить максимальное число очков, которое вы можете получить, если будете использовать секретную технику оптимально.

Входные данные

Первая строка входных данных содержит четыре целых числа 𝑛,𝑎,𝑏 and 𝑘 (1≤𝑛≤2⋅105,1≤𝑎,𝑏,𝑘≤109) — количество монстров, сила вашей атаки, сила атаки соперника и количество раз, которое вы можете использовать секретную технику.

Вторая строка входных данных содержит 𝑛 целых чисел ℎ1,ℎ2,…,ℎ𝑛 (1≤ℎ𝑖≤109), где ℎ𝑖 количество очков здоровья у 𝑖-го монстра.

Выходные данные

Выведите одно целое число — максимальное число очков, которое вы можете получить, если будете использовать секретную технику оптимально.

Примеры

входные данные

6 2 3 3

7 10 50 12 1 8

выходные данные

5

входные данные

1 1 100 99

100

выходные данные

1

входные данные

7 4 2 1

1 3 5 4 2 7 6

выходные данные

6

**Решение:**

n, a, b, k = input('Введите данные по игре' + '\t\t\t\t\t').split()

n = int(n); a = int(a); b = int(b); k = int(k)

while ((n < 1) or (n > 2\*105) or (a < 1) or (a > 109) or (b < 1) or (b > 109) or (k < 1) or (k > 109)):

print('Необходимо ввести данные по следующим правилам: 1≤𝑛≤2⋅105,1≤𝑎,𝑏,𝑘≤109')

n, a, b, k = input('Введите данные по игре' + '\t\t\t\t\t').split()

n = int(n); a = int(a); b = int(b); k = int(k)

check = True

while(check):

h = input('Введите данные о здоровье монстров:' + '\t\t\t').split()

h = [int(item) for item in h]

check = False

for i in range (len(h)):

if ((h[i] < 1) or (h[i] > 109)):

print ('Элемент должен находится в промежутке от 1 до 109!')

check = True

break

for i in range(n):

h[i] = ((h[i] - 1) % (a + b)) // a;

h.sort()

s = 0;

for i in range(n):

s += h[i];

if (s > k):

print('Максимальное число очков, которое вы можете получить:' + '\t' + str(i))

break

### **Задача 9**

Олимпийские игры только что начались, и Федерико с нетерпением ждет начала марафонского забега.

В марафоне будут участвовать 𝑛 спортсменов, от 1 до 𝑛, и все они в прошлом участвовали в 5 важных марафонах, пронумерованных от 1 до 5. Для каждого 1≤𝑖≤𝑛 и 1≤𝑗≤5 Федерико помнит, что спортсмен 𝑖 занял место 𝑟𝑖,𝑗 в марафоне 𝑗 (например, 𝑟2,4=3 означает, что спортсмен 2 был третьим в марафоне 4).

Федерико считает, что спортсмен 𝑥 превосходит спортсмена 𝑦, если спортсмен 𝑥 занял лучшее место, чем спортсмен 𝑦, по крайней мере в 3 прошлых марафонах, т.е. 𝑟𝑥,𝑗<𝑟𝑦,𝑗 по крайней мере для 3 различных значений 𝑗.

Федерико считает, что у спортсмена хорошие шансы получить золотую медаль на Олимпийских играх, если он превосходит всех остальных спортсменов.

Найдите любого спортсмена, у которого хорошие шансы получить золотую медаль (то есть спортсмена, который превосходит всех остальных спортсменов), или определите, что такого спортсмена нет.

Входные данные

Первая строка содержит одно целое число 𝑡 (1≤𝑡≤1000) — количество наборов входных данных. Затем следуют 𝑡 наборов входных данных.

Первая строка каждого набора входных данных содержит одно целое число 𝑛 (1≤𝑛≤50000) — количество спортсменов.

Затем следуют 𝑛 строк, каждая из которых описывает места одного спортсмена.

𝑖-я из этих строк содержит 5 целых чисел 𝑟𝑖,1,𝑟𝑖,2,𝑟𝑖,3,𝑟𝑖,4,𝑟𝑖,5 (1≤𝑟𝑖,𝑗≤50000) — места спортсмена 𝑖 в последних 5 марафонах. Гарантируется, что в каждом из 5 прошедших марафонов 𝑛 спортсменов заняли различные места, т.е. для каждого 1≤𝑗≤5, 𝑛 значений 𝑟1,𝑗,𝑟2,𝑗,…,𝑟𝑛,𝑗 попарно различны.

Гарантируется, что сумма 𝑛 по всем наборам входных данных не превышает 50000.

Выходные данные

Для каждого набора входных данных выведите одно целое число — номер спортсмена, у которого хорошие шансы получить золотую медаль (то есть спортсмена, который превосходит всех остальных спортсменов). Если таких спортсменов нет, выведите −1. Если таких спортсменов больше, чем один, выведите любого из них.

Пример

входные данные

4

1

50000 1 50000 50000 50000

3

10 10 20 30 30

20 20 30 10 10

30 30 10 20 20

3

1 1 1 1 1

2 2 2 2 2

3 3 3 3 3

6

9 5 3 7 1

7 4 1 6 8

5 6 7 3 2

6 7 8 8 6

4 2 2 4 5

8 3 6 9 4

выходные данные

1

-1

1

5

**Решение:**

t = int(input('Введите количество наборов:' + '\t'))

while ((t < 1) or (t > 1000)):

print('Необходимо ввести целое число от 1 до 1000!')

t = int(input('Введите количество наборов:' + '\t'))

for l in range(t):

n = int(input('\n' + 'Введите количество спортсменов:' + '\t'))

while ((n < 1) or (n > 50000)):

print('Необходимо ввести целое число от 1 до 50000!')

n = int(input('Введите количество спортсменов:' + '\t'))

a = []

for p in range(n):

check = True

while(check):

b = input('Введите места ' + str(p+1) + ' спортсмена:' + '\t').split()

b = [int(item) for item in b]

check = False

for i in range (len(b)):

if ((b[i] < 1) or (b[i] > 50000)):

print ('Место должно находится в промежутке от 1 до 50000!')

check = True

break

a.append(b)

l = [] #Список, содержащий числа, показывающие скольких спортсменов данный спортсмен лучше

m = [0,0]

for i in range(n):

k = 0 #Счётчик(скольких спорсменов превосходит вообще)

for j in range(n):

if i == j:

continue

sum = 0 #Счётчик соревнований, по которым он превосходит одного спортсмена

for p in range (5):

if (a[i][p] < a[j][p]):

sum += 1

if (sum >= 3):

k += 1

if k > m[1]:

m[0] = i

m[1] = k

l.append(k)

if len(set(l)) == 1:

print('Результат: ' + str(-1))

else:

print('Результат: ' + str((m[0]+1)))

### **Задача 10**

Лео-младший рисует в тетради в клеточку (каждая страница размечена квадратной сеткой). Можно считать, что страницы бесконечно большие в любом направлении.

Лео-младший закрашивает некоторые клетки на странице серым цветом. Он считает получившийся рисунок красивым, если выполнены следующие условия:

Рисунок связен, то есть, из любой серой клетки можно попасть в любую другую по цепочке из серых клеток, в которой каждые две последовательные клетки — соседи (то есть, имеют общую сторону).

Каждая серая клетка имеет чётное количество серых соседей. В рисунке ровно 𝑛 серых клеток, у которых все соседи серые. Количество остальных серых клеток может произвольным (но разумным, чтобы их можно было перечислить).

Лео-младший пытается нарисовать красивый рисунок для выбранного значения 𝑛. Помогите ему, найдите любой пример красивого рисунка.

Чтобы вывести координаты клеток в ответе, предполагайте, что на странице введена декартова система координат, в которой одна из клеток является началом координат (0,0), оси 0𝑥 и 0𝑦 перпендикулярны и сонаправлены линиям сетки, и шаг единичной длины в любом направлении вдоль любой оси ведёт в соседнюю клетку.

Входные данные

В единственной строке записано одно целое число 𝑛 (1≤𝑛≤500) — количество серых клеток, у которых все соседи серые.

Выходные данные

В первой строке выведите одно целое число 𝑘 — количество серых клеток в вашем рисунке. По техническим причинам 𝑘 не должно превосходить 5⋅105.

Каждая из следующих 𝑘 строк должна содержать две целых числа — координаты очередной серой клетки в вашем рисунке. Все перечисленные клетки должны быть различны, и рисунок должен отвечать всем требованиям, перечисленным выше. Все координаты не должны превосходить 109 по абсолютной величине.

Можно показать, что существует ответ, который удовлеторяет всем требованиям и укладывается в ограничение на 𝑘.

Пример

входные данные

4

выходные данные

12

1 0

2 0

0 1

1 1

2 1

3 1

0 2

1 2

2 2

3 2

1 3

2 3

**Решение:**