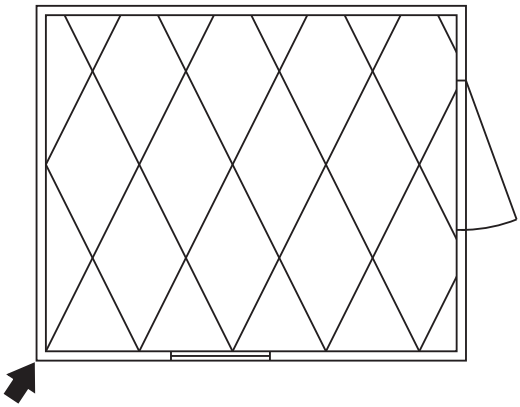
Problems

A. Rhomboid tiles

The floor of a rectangular room must be covered with identical tiles that have the shape of a rhombus, so that their diagonals are parallel to the room walls. You have a laser tile cutter that can cut the tiles with practically no waste. The owner of the room demands that the cuts may only be adjacent to the walls, that the pattern design is matched along the tile sides, and that from one corner of the room the tile halves run along the two adjacent walls (the window wall and the wall opposite the door). In other words, the floor of the room looks as a rectangle on an infinite plane covered with tiles so that:

- all the tiles on the plane have the same orientation and touch each other only along a full side;
- one of the vertices of the rectangle (indicated by arrow in the figure below) is located at a tile vertex.



Find out the minimal number of tiles that you will have to buy to satisfy these requirements.

Input

Four positive integers L, W, a, b on a single line, respectively, the length and width of the room, and the diagonals of the tile rhombuses. All numbers do not exceed 10^4.

Output

Output a single integer, the minimum necessary number of tiles.

Scoring

Subtask 1

All numbers do not exceed 100. Points will be awarded only if all tests from the statement and the subtask pass.

Subtask 2

All numbers do not exceed 200. Points will be awarded only if all tests from the statement and the subtask pass.

Subtask 3

No additional restrictions. Points will be awarded only if all tests the statement and the subtask pass.

Example

tiles.in	tiles.out
220 180 50 100	18

B. Skis sorting

A new fully automated ski factory has recently opened near Yakutsk. Production is divided into several stages, including manufacturing and packaging. Skis are produced in batches consisting of n different pairs. But once there was a failure in management system and the manufacturing robot started to output skis in not sorted order. The order in which the manufacturing robot produced skis was also partially lost.

Because of this, the packaging robot had to sort skis into pairs on his own. He acts in the following manner: in the beginning of each second he takes the next ski from the conveyor belt and checks whether he's already holding the second ski from that pair. If he does, he packages them together and passes the pair to the next robot. Otherwise, he holds this ski until he finds its pair.

Now the packaging robot has to hold more skis than before the failure. Let us say that if the robot is holding x skis between receiving the ith and the $(i+1)$th skis, then its *load* at that moment is equal to x. Define the total load of the robot as sum of all loads in all moments of time from 1 to $2n-1$.

Since the log of the order in which the skis came from conveyor belt is partially lost, it's not possible to calculate the total load of the packaging robot precisely. Instead, the engineers of the factory ask you: what is the *average* total load over all possible outcomes that satisfy given log data? The average

total load is defined as the sum of all total loads of the packaging robot for all possible ski orders satisfying the input data, divided by the number of such orders.

Input

First line of input contains integer n ($1 \le n \le 10^5$) — number of ski pairs produced in the batch. Next line contains $2n$ integers a_i ($0 \le a_i \le n$). If $a_i \ge 1$, then the packaging robot received ski from pair number a_i in i-th second. Otherwise, if $a_i = 0$, then the information about the ski in moment i was lost. Each number, except for 0, is present in this list at most two times.

Output

Output one floating-point number: the average total load of the packaging robot over all possible orders satisfying the input data. Your answer will be considered correct if its relative error does not exceed 10^{-6}.

Scoring

Subtask	Points	Constraints	
		n	Additional
1	20	$n \le 5$	$a_i \ne 0$
2	20	$n \le 5$	—
3	30	$n \le 1000$	—
4	30	$n \le 10^5$	—

Examples

skis.in	skis.out
2 0 0 1 0	3.333333333333333
3 1 3 1 3 2 2	5.0

C. Fire stations

In a certain country all settlements are situated on the one road. The most important problem for these settlements is the fire safety problem, because the majority of the buildings are made of wood. It's necessary to construct a number of fire stations along the road, where fire-engines will based at.

Fire station can be built only in a settlement, and the cost of the construction differs in each settlement. Number all the settlements sequentially from 1 to n and denote by c_i the cost of fire station construction for settlement number i.

Settlements are situated along the road almost uniformly, so we consider that a fire-engine can reach any neighboring settlement in 1 hour. If it takes more then k hours to get to burning building then it can't be saved. Therefore, a ride from a fire station to any settlement mustn't take more than k hours.

Determine the minimal cost of constructing the fire stations in the settlements.

Input

First line of input contains two numbers n and k ($1 \le n \le 10000$, $0 \le k \le 100$) — number of settlements on the road and maximal time for the fire-engine to get to the burning building. Second line contains n numbers c_i ($0 \le c_i \le 10^6$) — fire station constructions cost for i-th settlement.

Output

Output a single number, minimal expense for building all the fire stations.

Scoring

Subtask 1

Additional limitation: $n \le 10$

Points will be awarded only if all tests from the statement and the subtask pass.

Subtask 2

Additional limitation: $k \le 1$

Points will be awarded only if all tests from the statement and the subtask pass.

Subtask 3

There is no additional limitations.

Points will be awarded only if all tests from the statement and the subtask pass.

Examples

fire.in	fire.out
4 2 2 4 3 2	3
3 1 2 3 2	3

D. Labyrinth

Vasya is stuck in a labyrinth. The labyrinth is a non-self-intersecting polygon with N vertices and sides parallel to the coordinate axes. In the evening Vasya has a programming practice, so he wants to escape from the labyrinth as quickly as possible. He is at the point (X_s, Y_s) inside the labyrinth, and the exit is at the point (X_e, Y_e). For simplicity, let us represent Vasya as a square with side of A units. The sides of the square are always parallel to the axes. Vasya is considered to have escaped when his center reaches the exit point. Find the minimal escape time. Vasya's speed is V units/sec. If Vasya cannot escape, output -1.

Input

The first line holds two real numbers X_s and Y_s ($0 \le X_s, Y_s \le 100$). The next line holds two real numbers X_e and Y_e ($0 \le X_e, Y_e \le 100$). The following two lines hold real numbers A and V ($0.5 \le A, V \le 10$), respectively. The following lines describe the polygon: a single integer N ($1 \le N \le 100$) on a line by itself, and each of the following N lines holds a pair of real numbers, X_i and Y_i being the coordinates of a vertex ($0 \le X_i, Y_i \le 100$). All the reals are guaranteed to give an integer when multiplied by 2.

Output

Output the answer on a single line. Your answer will be considered correct if its relative error does not exceed 10^{-6}.

Scoring

Points for all the tests are awarded independently.

Example

labyrinth.in	labyrinth.out
1.5 4.5 4.5 1.5 1 1 6 0 0 0 6 3 6 3 3 6 3 6 0	4.472135955

E. Liberloun

The planet Liberloun has N communication stations, some of which have retranslators around them. At a certain initial moment, a talented astronomer Aytal from Earth sends a message signal to each of the comm stations, which reaches them instantaneously, but with some loss and scatter. It is known that each retranslator receives a fraction p of the initial message (the more retranslators there are, the greater portion of the signal goes to them). Also, a fraction q of the signal is lost. It takes takes 3 hours for a retranslator to forward the message to its comm station. Aytal's computer only retransmits the lost part of the message after 10 hours. The station receives a readable message if the combined fraction of the message that reaches it is no less than S.

Aytal asks you to count how many stations will receive a readable message after H hours.

Input

The first line of input contains an integer N ($1 \le N \le 10^6$) and real numbers $p,\ q$ ($0 \le p,\ q \le 1$), the number of stations and the fractions, respectively. The second line holds N integers $a_1, a_2, \dots, a_n$, where a_i is the number of retranslators around the ith comm station ($0 \le a_i \le 10^6$). The third line contains two real numbers $H,\ S$ ($0 \le H \le 1000,\ 0 \le S \le 1$).

It is guaranteed that for all i the condition $p \times a_i + q \leq 1$ is satisfied.

Output

Output a single integer, the number of stations that will receive a readable message after H hours.

Scoring

Solutions that fail the tests in the examples will be awarded 0 points and not be further tested.

Each test is scored separately.

Examples

liberloun.in	liberloun.out
2 0.01 0.5 0 1 0 0.5	1
4 0.2 0.2 0 1 2 3 8 0.3	4

Scoring

In the first example the first station immediately receives half of the message, and the second station 0.49 of the message. Only the first station receives a readable message.

In the second example, after 3 hours the stations number 2, 3, and 4 will receive messages from their retranslators, so all of the stations will receive 0.8 of the message.

F. Necklace, again

Alyosha Popovich has beads colored in M different colors. He wants to make a necklace from these beads for Vasilisa the Beautiful. He believes that necklaces containing all M available colors in any $M + 1$ adjacent beads are especially *beautiful*. How many different beautiful necklaces of length N can he make? This number can be huge, so you are asked to calculate its remainder modulo $10^9 + 7$.

Input
Input file contains two positive integer numbers M and N ($2 \leq M < N \leq 100000$) separated by space.

Output
Output the number of different necklaces modulo $10^9 + 7$ into output file.

Scoring
Subtask 1
Additional limitation:$N \leq 8$. 20 points.
Points will be awarded only if solution passes all tests from the statement and the subtask.
Subtask 2
Additional limitation: $N \leq 500$. 40 points.
Points will be awarded only if solution passes all tests from the statement and the subtask.
Subtask 3
There is no additional limitations. 40 points.
Points will be awarded only if solution passes all tests from the statement and the subtask.

Examples

necklace.in	necklace.out
2 6	26
5 8	8520

Scoring
Necklace is considered as a sequence of beads of length N.

G. Monetary system of the Land of Fools

It is known that in the Land of Fools there are two types of currency: golden and wooden. There are coins with the values of a_1, a_2, ..., a_K wooden units and a golden coin. By the order of wise Buratino, the ruler of the Land of Fools, the monetary system of the country is organized in such a way that any amount from 1 up to M of wooden money can be paid using no more than N coins. Define the minimal and maximal amount of wooden money that the golden

coin can be worth. In the case when one golden coin is equal to exactly one number of wooden units, print it as both minimal and maximal.

It is guaranteed that the order of Buratino is fulfilled and one golden coin is not worth more than M wooden units.

Input

First line contains three numbers K, M and N ($1 \le K \le 500, 1 \le M, N \le 2000$). Second line contains K integers $a_1, a_2, \ldots, a_K$ ($1 \le a_i \le 500$).

Output

Output two space separated integers, the minimal and maximal wooden value of the golden coin.

Scoring

Solutions that fail the tests in the examples will be awarded 0 points and not be further tested.

Each test is scored separately.

Examples

coins.in	coins.out
3 10 2 1 2 3	7 7
3 15 3 1 2 5	4 13

H. Shopping

Grisha, a young coder, goes shopping for groceries every Saturday. This time he has decided to minimize his spending.

Grisha's town has N grocery shops, and for convenience he has numbered them from 1 to N. Some shops are connected with minibus routes, and the cost of the minibus from the ith shop to the jth one is p_{ij} rubles.

There are K different kinds of groceries. The shops stock limited amount of each. Grisha also knows the price for each kind of groceries in each shop. The prices in the shops may differ.

Grisha lives near the shop number 1, so he spends nothing to get there. He may finish his shopping in any of the N shops. Assume that the return home will also cost him nothing.

Given a shopping list, determine the minimum necessary amount of money that Grisha will have to take with him.

Input

The first line of the input contains a single integer N ($1 \leq N \leq 17$), the number of shops.

The following N lines form an $N \times N$ matrix P. The matrix elements p_{ij} ($0 \leq p_{ij} \leq 2000$) determines the cost of minibus from the ith shop to the jth shop. If $p_{ij} = 0$, it means that there is no direct minibus route connecting the shops number i and j. The matrix is symmetric, and all p_{ii} are zero.

The next line holds a single integer number K ($1 \leq K \leq 50$), the number of types of groceries.

The next line holds K integers Q_i ($1 \leq Q_i \leq 2000$), the required amount of each grocery type on Grisha's shopping list.

What follows is K blocks describing the grocery stocks and prices in the shops. The ith block starts with a line holding a single integer M, the number of shops that have the ith grocery type. Each of the M following lines of the block holds three integers v, p, q ($1 \leq v \leq n, 0 \leq p \leq 2000, 1 \leq q \leq 2000$) meaning that the shop number v sells the ith grocery type for p rubles per piece and it has q pieces of that grocery.

Output

A single number, the minimum sum of money that will allow Grisha to buy all the groceries. If that is impossible, output -1.

Scoring

Solutions that fail the test in the example will be awarded 0 points and not be further tested.

Each test is scored separately.

Example

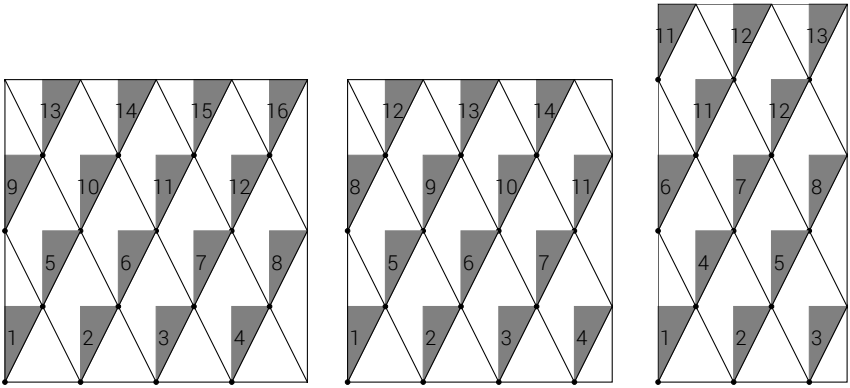
shops.in	shops.out
5 0 1 3 0 2 1 0 5 0 5 3 5 0 7 2 0 0 7 0 2 2 5 2 2 0 3 3 5 5 3 1 3 2 3 2 1 5 4 3 3 2 4 3 3 5 4 5 2 1 4 1 9 1 2 8 2 3 7 3 4 6 1	70

Solutions

A. Rhomboid tiles

Represent the floor of the room as a rectangle on a plane lined by two families of parallel straight lines forming rhombuses as required. Consider all the intersection points of these lines that lie inside the rectangle except on its upper and right side. Denote their number by N. Each of them is touched from above by a part of a tile containing the *same* nonempty region of the lower right quarter. So the minimum required number of tiles is at least N.

Consider first the case when the dimensions of the room are multiples of the halves of the tile diagonals: $L = n\frac{a}{2}$, $W = m\frac{b}{2}$, n, m integral. In this case, one can readily see that those parts not containing any region of the lower right tile quarter can be covered from parts left from opposite walls, irrespective of whether n and m are even or odd. Therefore in this case no additional tiles are necessary and the minimum required number of tiles is exactly N.



$n = 8, m = 4,$ $N = 16$ | $n = 7, m = 4$ $N = 14$ | $n = 5, m = 5$ $N = 13$

In the general case, when L/a and W/b are not integer or half-integer, we can increase the room dimensions so that these numbers reach the nearest integer or half-integer. This will reduce the situation to the above special case, and won't bring any new intersection points inside the rectangle, so the number N will stay the same.

Now N can be counted by examining the cases depending on n and m

being odd or even, or by looping, however these are equivalent to the formula

$$N = \left\lceil \frac{nm}{2} \right\rceil,$$

where $\lceil x \rceil$ denotes the ceiling of x.

We can also satisfy the requirements and perhaps get a smaller number after rotating the rectangle 90 degrees. Therefore the final answer is

$$N = \min\left(\left\lceil \frac{\lceil \frac{2L}{a} \rceil \lceil \frac{2W}{b} \rceil}{2} \right\rceil, \left\lceil \frac{\lceil \frac{2L}{b} \rceil \lceil \frac{2W}{a} \rceil}{2} \right\rceil \right).$$

B. Skis sorting

Let's reformulate the problem as follows: given a sequence of $2n$ integers, where every number from 1 to n is present at most twice. The important observation is that the *the total load* of the robot for the sequence without skipped numbers is equal to the sum of distances between two occurrences of all numbers. That means that, instead of calculating the total average load of the robot over all possible valid orders, we can calculate the contribution of each number from 1 to n independently.

To solve the first subtask, just write a program that calculates the answer using the definition given in the statement. For the second subtask you could write a recursive bruteforce algorithm that iterates over all valid orders. Writing such solutions could help you validate your solution for next subtasks.

Consider a number x. If it's already present twice, then add the distance between its occurrences to the answer and be done with it. For all numbers, that are not present in the input, the answer is the same. For each free position i calculate the count k_i and sum of indices of all position to the left of it s_i. When this position is the second occurrence, the contribution to the answer is equal to $k_i i - s_i$. Sum these values over all free positions and divide it by $\frac{c_0(c_0-1)}{2}$, where c_0 is the total number of free positions.

The remaining case, where the number is only present once is similar. If some number is present at position i, then its contribution is equal to $\frac{1}{c_0}((k_{left} \cdot i - s_{left}) + (s_{right} - k_{right} \cdot i))$, where k_{left}, s_{left}, k_{right} and s_{right} — number and sum of free positions to the left and to the right of i. This solution can be implemented in $O(n)$.

C. Fire stations

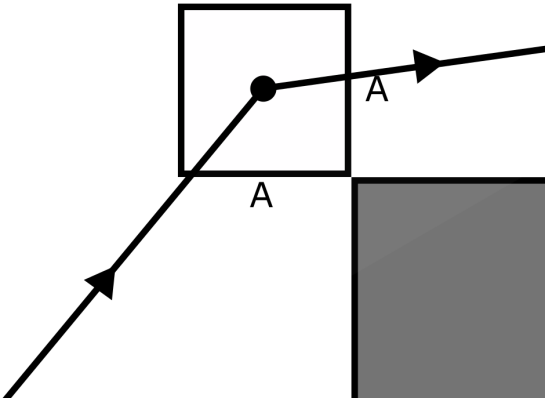
Use dynamic programming. Denote A_i the minimum expenditure to provide fire protection to settlements number 1 to i, while having a fire station in settlement number i. Obviously, $A_1 = C_1$. If we have a station at i, the distance to the previous station must be no greater $2k + 1$. Therefore

$$A_i = \min(A_{\overline{i-2k-1}}, A_{\overline{i-2k}}, \dots, A_{i-1}) + C_i,$$

where $\overline{x}$ denotes $\max(x, 1)$. The answer to the problem is $\min(A_{\overline{n-k}}, A_{\overline{n-k+1}}, \dots, A_n)$.

D. Labyrinth

Vasya's movement to the exit point will be optimal when it is composed of straight line segments, meeting each other at points where he touches the polygon vertices. Due to the restrictions of the problem, he can only touch them at certain points.



Compute all the points where a square vertex can touch a polygon vertex, and for each one check whether Vasya can reach it. Call such a point *regular*. Now for each pair of vertices of the graph including the regular points, the starting point and the exit point, check whether they can be connected by

a straight line segment along which Vasya can move. If so, connect these graph vertices by an edge, the length of which is the time required for Vasya to traverse the corresponding segment. Use Dijkstra's algorithm to find the distance between the starting point and the exit point.

E. Liberloun

Denote by x the missing portion for first station, and represent H as $H = 10 \times k + r$, where k and r are positive integers with $r < 10$. The signal was sent at times $0,\ 10,\ 20, \ldots,\ 10 \times k$, so at that moment we have $k + 1$ signals, with losses, respectively $q,\ q^2, \ldots,\ q^{k+1}$.

Consider the first station. If less than 3 hours passed after the last $(k+1)$th transmission then $x = q^{k+1} + q^k * p * a_1$, where $q^k * p * a_1$ means the part intercepted by retranslators. Otherwise $x = q^{k+1}$, that is a part of message is already received by the station from the retranslators.

Therefore $1 - x$ is the fraction of the message received by the station. Repeating the computation of the fraction received by each station, and compare it with S.

F. Necklace, again

There are beads of k colors. Denote $C(n)$ the number of different beautiful necklaces of length n. Obviously $n \geq k$. We have the following formula:

$$C(m + k + 1) = C(m + 1) + C(m + 2) + \ldots + C(m + k).$$

If we know the values of $C(k+1)$, $C(k+2)$, ..., $C(k+k)$, then we can find $C(n)$ for all subsequent n.

Compute $C(k + 1)$. Consider the last k beads. There are two cases. First, all of them may be unique. The number of possible necklaces for this case is $k \times k!$. If there are two beads of the same color among k beads, then some color does not appear among them, and it is the color of the first bead. Using this, we have the number of beautiful necklaces $C_k^2 \times k \times (k - 1) \times (k - 2)!$. Denote the numbers in the first and second case, respectively, by f_1 and r_1. So $C(k + 1) = f_1 + r_1$.

Compute $C(k + 2)$. The number can increase only due to f_1. Therefore the $k+2$th bead can be any color. We now have that $C(k+2) = C(k+1)+(k-1) \times f_1$.

Compute $C(k+3)$. Again, separate $C(k+2)$ into two parts analogous to f_1 and r_1. Denote them f_2 и r_2. We are only interested in f_2. Let us return to f_1, represent it as $a+d$, where $a = \frac{f_1}{k-1}$, $d = \frac{(k-2)f_1}{k-1}$. Then $f_2 = a + 2d$. Therefore, $C(k+3) = C(k+2) + (k-1) \times f_2$. Denote f_3 the factorial part of $C(k+3)$. $f_3 = a + 4d$. We obtain $C(k+4) = C(k+3) + (k-1) \times f_3$.

In the general case, $f_m = a + 2^{m-1}d$, $C(m+1) = C(m) + (k-1) \times f_m$. We continue the computation until we find $C(2k)$. After that we continue using the formula $C(m+k+1) = C(m+1) + C(m+2) + ... + C(m+k)$. To simplify the computation, split the number $C(k+1)$ as $C(k+1) = a_1 + a_2 + ... + a_k$, where $a_i = C(i)$. Пусть некоторое $C(m) = b_1a_1 + b_2a_2 + ... + b_ka_k$. Тогда $C(m+1) = b_ka_1 + (b_k + b_1)a_2 + (b_k + b_2)a_2 + ... + (b_{k-1} + b_k)a_k$. Using this recurrence we can compute any $C(n)$.

G. Monetary system of the Land of Fools

First, calculate the following value: f_i — minimum number of coins to get sum i. This value can be calculated with dynamic programming in $O(MK)$ as follows: initialize $f_i = 0$ for $i = 0$ and $f_i = +\infty$ for $i > 0$. Then, for all coins a_k and indices i update $f_i = \min(f_i, f_{i-a_k} + 1)$. If you iterate over i in increasing order, you account for the fact that you can take some coin multiple times.

Back to solving the problem. Define sum i as *bad*, if $f_i > N$. The new golden coin should fix all bad sums. If the value of the golden coin is x, then bad position i will be fixed, if there is some $j < i$, such that $j - i$ is divisible by x and $f_i + \frac{i-j}{x} \leq N$. Using this property, we can list all values x such that bad position i will be fixed. To do that, iterate over all good positions $j < i$, iterate over all divisors of $i - j$, check that value of f_i is less than or equal to N.

It's relatively known that $\sum_{i=1}^{n} d(i) = O(n \log n)$, where $d(n)$ is the number of divisors of n. That means that for fixed i we only need to consider $O(M \log M)$ numbers. After that, we can intersect all these sets and take the minimum and maximum value. The running time of the second part and the whole solution is $O(M^2 \log M)$.

H. Shopping

This problem can be split into two subproblems. The first is to find the minimum amount of money to spend on minibuses to visit a given subset of shops. The second subproblem is to determine the minimum amount of money to buy all required groceries from a given subset of shops. , that is to say that . Таким образом, для определенного множества магазинов можно вычислить минимальную сумму денег, которая требуется чтобы купить товары только в магазинах из этого множества. Finding the minimum over all the shop subsets, we can find the answer.

For the first subproblem compute the matrix A of minimum distancesThis can be done using the Floyd–Warshall algorithm.

Then reduce the problem to a dynamic programming one. Introduce an two-dimensional array D with $D[mask][last]$ storing the minimum amount of money for the state $[mask][last]$, necessary to visit all the shops given in the *mask*, and finishing at the shop number *last*. If the ith bit of *mask* is 1 it means that the shop number i is visited. Initialize $D[mask][last]$ where *mask* only contains one shop *last*, to zero 0.

To find $D[mask][last]$ first define *prevMask* for previous step. Because we must finish in the shop number *last*, the shop set for the previous state only differs in not containing the element *last*, ie $prevMask = mask \backslash last$. Let the shop *prevLast* be in *prevMask*, then if we arrive from the state $[prevMask][prevLast]$, total price will be $D[prevMask][prevLast] + A[prevLast][last]$. Checking all *prevLast* from *prevMask*, we find the minimum travel cost. In this manner, for a given set *mask* we can find the minimum travel cost by finding minimum over all $D[mask][last]$.

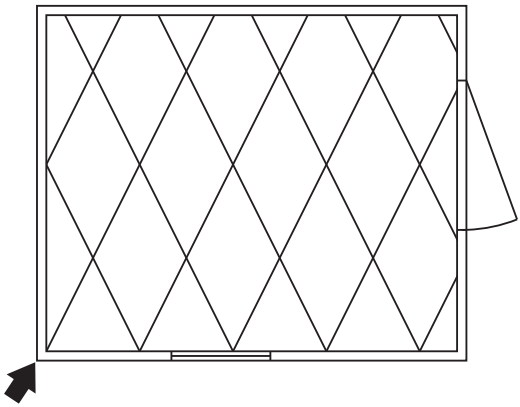
Consider a special case for the second subproblem where we only have one type of groceries. Assume we visit all shops given by *mask*. Sort the shops in the increasing order by price. We go along the sorted list of shops until we can buy the required amount of the grocery. In the general case of multiple grocery types the cost is determined separately for each type.

Задачи

A. Ромбовидная плитка

Пол в прямоугольной комнате требуется выложить одинаковыми плитками, имеющими форму ромба, так, чтобы диагонали каждого ромба были параллельны стенам комнаты. У вас есть лазерный резак, который режет плитку практически без отходов. Заказчик требует, чтобы разрезы примыкали только к стенам, рисунок на плитке совмещался по сторонам, и от угла двух смежных стен (у окна и напротив двери) начинались половинки плиток. Иными словами, пол комнаты должен выглядеть как прямоугольник, вырезанный из бесконечной плоскости, причем:

- плитки на плоскости одинаково ориентированы и прилегают друг к другу только по целым сторонам;
- одна из вершин прямоугольника находится в общем углу двух плиток (показан стрелкой на рисунке ниже).



Подсчитайте минимальное количество плиток, которые потребуется купить, чтобы выполнить все требования.

Формат входных данных
Записанные через пробел в одной строке четыре положительных целых числа L, W, a, b — длина и ширина комнаты, а также диагонали ромбов плитки. Все числа не превышают 10^4.

Формат выходных данных
Одно целое число — минимально необходимое число плиток.

Система оценивания

Подзадача 1

Дополнительное ограничение: все числа не превосходят 100. Баллы начисляются только в том случае, если все тесты данной подзадачи и тесты из примера пройдены.

Подзадача 2

Дополнительное ограничение: все числа не превосходят 200. Баллы начисляются только в том случае, если все тесты данной подзадачи и тесты из примера пройдены.

Подзадача 3

Дополнительных ограничений нет. Баллы начисляются только в том случае, если все тесты данной подзадачи и тесты из примера пройдены.

Пример входного и выходного файлов

tiles.in	tiles.out
220 180 50 100	18

B. Сортировка лыж

В окрестности Якутска начал работу новый полностью автоматизированный завод по производству лыж. Производство разделено на несколько этапов, среди которых изготовление и упаковка. Лыжи на этом заводе выпускаются партиями по n различных пар. Однажды в системе управления произошел сбой, и во время изготовления одной партии робот-изготовитель выдавал лыжи не в отсортированном порядке. Порядок, в котором робот-изготовитель выдавал лыжи, также оказался частично утерян.

В связи с этим, на долю робота-упаковщика выпала тяжелая участь составлять пары лыж самостоятельно. Он действует следующим образом: забирает в начале каждой секунды с конвейера очередную лыжу и смотрит, держит ли он парную к ней. Если он находит пару, он ее упаковывает и отдает дальше. Иначе, он берет эту лыжу и держит, пока не найдет к ней парную.

Из-за неполадок робот-упаковщик теперь вынужден совершать больше работы, чем раньше. Скажем, что если он держит x лыж между получением i-й и $(i+1)$-й лыжи, то его *загруженность* в момент времени

i равна x. Общей загруженностью робота назовем сумму всех загруженностей в моменты времени от 1 до $2n - 1$.

Поскольку журнал событий, где описывались лыжи, которые получал упаковщик, оказался частично утерян, общую загруженность робота точно посчитать не удастся. Вместо этого, инженеры завода спрашивают вас: чему равна общая загруженность *в среднем* по всем возможным исходам, удовлетворяющим данному журналу? Общая загруженность в среднем определяется как сумма общих загруженностей робота для всех возможных порядков выдачи лыж, удовлетворяющих входным данным, деленная на количество таких порядков.

Формат входных данных

В первой строке содержится число n ($1 \le n \le 10^5$) — количество пар лыж, которые были изготовлены. В следующей строке содержится $2n$ целых чисел a_i ($0 \le a_i \le n$). Если $a_i \ge 1$, то известно, что в i-ю секунду робот получил лыжу из комплекта с номером a_i. Иначе, если $a_i = 0$, то информация о том, что получил робот, утеряна. Каждое число, кроме 0, встречается в этом списке не более двух раз.

Формат выходных данных

Выведите единственное число с плавающей точкой: суммарная загруженность робота в среднем по всем возможным порядкам изготовления лыж, согласованных с журналом действий. Ответ будет считаться корректным, если его относительная погрешность не превосходит 10^{-6}.

Система оценивания

Подзадача	Баллы	Ограничения	
		n	Дополнительные
1	20	$n \le 5$	$a_i \ne 0$
2	20	$n \le 5$	—
3	30	$n \le 1000$	—
4	30	$n \le 10^5$	—

Примеры входного и выходного файлов

skis.in	skis.out
2 0 0 1 0	3.333333333333333
3 1 3 1 3 2 2	5.0

C. Пожарная охрана

В одной республике все населенные пункты расположены вдоль одной единственной дороги. Поскольку большинство построек в них деревянные, то одной из важных задач является организация пожарной охраны. Для этого необходимо построить вдоль дороги несколько пожарных станций, в которых будут базироваться пожарные машины.

Станции можно построить только в населенных пунктах, причем затраты на постройку пожарной станции в разных населенных пунктах могут быть различны. Будем считать, что населенные пункты пронумерованы от одного конца дороги к другому последовательно числами от 1 до n, а стоимость постройки пожарной станции в населенном пункте с номером i равна c_i.

Населенные пункты расположены вдоль дороги достаточно равномерно, поэтому будем считать, что расстояние между любыми двумя соседними населенными пунктами пожарная машина проезжает за один час. Если от момента возникновения пожара до момента приезда пожарной машины проходит более k часов, то горящий объект спасти не удается. Поэтому от любого населенного пункта (вне населенных пунктов пожары не возникают) до ближайшей пожарной станции путь должен занимать не более k часов.

Определите наименьшую сумму затрат на строительство пожарных станций.

Формат входных данных

Первая строка входного файла содержит два числа n и k ($1 \le n \le 10000$, $0 \le k \le 100$) — количество населенных пунктов, находящихся на дороге и максимальное допустимое время прибытия пожарной машины k.

Вторая строка содержит n чисел c_i ($0 \le c_i \le 10^6$) — стоимость постройки пожарной станции в населенном пункте с номером i.

Формат выходных данных

В единственной строке выходного файла выведите одно число — минимальную суммарную стоимость постройки всех пожарных станций.

Система оценивания

Подзадача 1

Дополнительное ограничение: $n \le 10$

Баллы начисляются только в том случае, если все тесты данной подзадачи и тесты из примера пройдены.

Подзадача 2

Дополнительное ограничение: $k \le 1$

Баллы начисляются только в том случае, если все тесты данной подзадачи и тесты из примера пройдены.

Подзадача 3

Дополнительных ограничений нет.

Баллы начисляются только в том случае, если все тесты данной подзадачи и тесты из примера пройдены.

Примеры входного и выходного файлов

fire.in	fire.out
4 2 2 4 3 2	3
3 1 2 3 2	3

D. Лабиринт

Мальчик Вася застрял в лабиринте. Лабиринт представляет собою произвольный несамопересекающийся многоугольник с N вершинами и параллельными к осям координат сторонами. Так как у него вечером занятия по программированию, то Вася хочет выбраться как можно быстрее. Вася находится в точке (X_s, Y_s) внутри лабиринта, а выход находится в точке (X_e, Y_e). Для простоты решения представим Васю как квадрат со сторонами A условных единиц. Стороны квадрата всегда параллельны

осям координат. Считается что Вася вышел из лабиринта, если его центр достигнет точки выхода. Найдите минимальное возможное время, за которое Вася сможет выйти из лабиринта. Скорость передвижения Васи равняется V у.е/с. Если Вася не может выбраться из лабиринта, вывести -1.

Формат входных данных

В первой строке заданы вещественные числа X_s и Y_s ($0 \leq X_s, Y_s \leq 100$). В следующей строке заданы вещественные числа X_e, Y_e ($0 \leq X_e, Y_e \leq 100$). В последующих двух строках заданы вещественные числа A и V ($0.5 \leq A, V \leq 10$) соответственно. Далее задан многоугольник: сначала дано целое число N ($1 \leq N \leq 100$), в последующих N строках находятся по два вещественных числа: координаты вершин многоугольника X_i и Y_i ($0 \leq X_i, Y_i \leq 100$). Гарантируется, что при умножении любого вещественного числа из входных данных на 2, получится целое число.

Формат выходных данных

В единственной строке вывести ответ. Ответ считается верным, если его относительная погрешность не превышает 10^{-6}.

Система оценивания

Баллы за каждый тест начисляются независимо.

Пример входного и выходного файлов

labyrinth.in	labyrinth.out
1.5 4.5 4.5 1.5 1 1 6 0 0 0 6 3 6 3 3 6 3 6 0	4.472135955

E. Либерлун

На планете Либерлун есть N станций связи, вокруг которых могут находиться ретрансляторы. Каждой станции в начальный момент времени талантливый астроном Айтал с Земли посылает сообщения в виде сигналов, которые доходят мгновенно, но с некоторыми потерями и отклонениями. Известно, что по p-ой части направленного станции сообщения попадает ретрансляторам (чем больше вокруг ретрансляторов, тем больше сигналов попадает к ним), и q-ая часть теряется. Ретранслятору требуется 3 часа, чтобы передать сообщение адресату. Ту часть сообщения, которая потерялась, компьютер Айтала повторно отправляет лишь через 10 часов. Считается, что станция получила читаемое сообщение, если полученная суммарная доля не меньше S.

Айтал просит вас подсчитать, сколько станций получат читаемое сообщение за H часов.

Формат входных данных

В первой строке записаны числа натуральное число N ($1 \le N \le 10^6$) и вещественные числа p, q ($0 \le p, q \le 1$) — количество станций и соответствующие доли. Во второй строке записаны N целых чисел $a_1, a_2, \ldots, a_n$, где a_i означает количество соседних ретрансляторов у i-й станции ($0 \le a_i \le 10^6$). В третьей строке записаны вещественные числа H, S ($0 \le H \le 1000$, $0 \le S \le 1$).

Гарантируется, что для всех i выполнено условие $p \times a_i + q \le 1$.

Формат выходных данных

Выведите единственное целое число — количество станций, которые получат читаемое сообщение за H часов.

Система оценивания

Представленное на проверку решение сначала проходит тестирование на тестах из примеров, приведенных в условии задачи. Если на этих тестах решение участника выдает правильные ответы, то тогда это решение проверяется с использованием комплекта тестов для этой задачи (при этом баллы за каждый тест начисляются независимо). В противном случае решение участника считается неверным, и за него участнику не начисляются какие-либо баллы.

Примеры входного и выходного файлов

liberloun.in	liberloun.out
2 0.01 0.5 0 1 0 0.5	1
4 0.2 0.2 0 1 2 3 8 0.3	4

Система оценивания

1. Сразу после отправления первая станция получает половину, а вторая — 0.49 объема сообщения. Следовательно, только первая станция получила читаемое сообщение.

2. По прошествии 3 часов 2, 3 и 4 станции получат сообщения от своих соседних ретрансляторов. Соответственно, у всех станций доли полученного сигнала будут равны 0.8.

F. Опять ожерелье

У Алеши Поповича есть бусинки M разных цветов в неограниченном количестве. Сколько различных красивых ожерелий длины N он может подарить Василисе Прекрасной? Ожерелье считается *красивым*, если среди любых $M+1$ подряд идущих бусинок в нем присутствуют бусинки всех M цветов. Это количество может быть большим, поэтому вместо самого количества посчитайте остаток от деления его на $10^9 + 7$.

Формат входных данных

Входной файл содержит записанные через пробел два целых числа M и N ($2 \leq M < N \leq 100000$).

Формат выходных данных

Выведите единственное целое число, остаток от деления количества красивых ожерелий на $10^9 + 7$.

Система оценивания

Подзадача 1

Дополнительное ограничение: $N \leq 8$. 20 баллов.

Баллы начисляются только в том случае, если все тесты данной подзадачи и тесты из примера пройдены.

Подзадача 2

Дополнительное ограничение: $N \leq 500$. 40 баллов.

Баллы начисляются только в том случае, если все тесты данной подзадачи и тесты из примера пройдены.

Подзадача 3

Дополнительных ограничений нет. 40 баллов.

Баллы начисляются только в том случае, если все тесты данной подзадачи и тесты из примера пройдены.

Примеры входного и выходного файлов

necklace.in	necklace.out
2 6	26
5 8	8520

Система оценивания

Ожерельем считается последовательность бусинок длины N.

G. Денежная система Страны Дураков

В Стране Дураков, как известно, в ходу два вида валюты – золотые и деревянные. При этом имеют хождение монеты номиналом a_1, a_2, ..., a_K деревянных, а также 1 золотой. По распоряжению Буратино – мудрого правителя Страны Дураков, денежная система устроена так, что любую сумму от 1 до M деревянных можно заплатить не более чем N монетами. Определите, какое наименьшее и наибольшее количество деревянных может быть в одном золотом, при условии, что золотой не дороже M деревянных? Если вариант только один, укажите его и как наименьший, и как наибольший.

Гарантируется, что указ Буратино выполняется, и золотой не дороже M деревянных.

Формат входных данных

В первой строке записаны три целых числа K, M, N ($1 \leq K \leq 500$, $1 \leq M, N \leq 2000$). Во второй строке – K целых чисел a_1, a_2, ..., a_K ($1 \leq a_i \leq 500$).

Формат выходных данных

Два целых числа через пробел — соответственно, минимально и максимально возможные числа деревянных в одном золотом.

Система оценивания

Представленное на проверку решение сначала проходит тестирование на тестах из примеров, приведенных в условии задачи. Если на этих тестах решение участника выдает правильные ответы, то тогда это решение проверяется с использованием комплекта тестов для этой задачи (при этом баллы за каждый тест начисляются независимо). В противном случае решение участника считается неверным, и за него участнику не начисляются какие-либо баллы.

Примеры входного и выходного файлов

coins.in	coins.out
3 10 2 1 2 3	7 7
3 15 3 1 2 5	4 13

H. Магазины

Молодой программист Гриша каждую субботу ходит за продуктами. В этот раз он решил минимизировать свои расходы.

В городе, где проживает Гриша, имеется N продовольственных магазинов. Для удобства он пронумеровал магазины от 1 до N. Между некоторыми магазинами ходят маршрутные такси. Стоимость проезда с i-го магазина в j-й составляет p_{ij} рублей. Всего продается K разных типов продуктов. В магазинах имеется определенное количество единиц того или иного продукта. Также Гриша знает, сколько стоит каждый продукт в каждом из магазинов. Цены на продукты в различных магазинах могут быть разными.

Гриша живет рядом с магазином с номером 1. Для того, чтобы добраться до этого магазина, он ничего не платит. Конечным магазином может быть любой из N магазинов. На обратную дорогу домой Гриша ничего не тратит.

Имея список покупок, определите наименьшую сумму денег, которую Гриша должен взять с собой.

Формат входных данных

В первой строке находится единственное целое число N ($1 \leq N \leq 17$) — количество магазинов.

Следующие N строк содержат матрицу $N \times N$. Элемент матрицы p_{ij} ($0 \leq p_{ij} \leq 2000$) указывает стоимость проезда с i-го магазина в j-й. Если p_{ij} равно 0, то это значит, что нет прямого маршрута с i-го магазина в j-й. Матрица симметрична. Элементы на главной диагонали всегда равны нулю.

В следующей строке находится единственное целое число K ($1 \leq K \leq 50$) — количество различных продуктов.

Далее идут K целых чисел Q_i ($1 \leq Q_i \leq 2000$) — количество единиц каждого вида продукта, которое нужно купить.

Затем идут K блоков, которые указывают, в каких магазинах продаются товары, и по какой стоимости. Каждый i-й блок начинается со строки, где находится единственное целое M — количество магазинов, где продается i-й продукт. Далее в M строках имеется по три целых числа v p q ($1 \leq v \leq n, 0 \leq p \leq 2000, 1 \leq q \leq 2000$) — в магазине с номером v продается i-й продукт по цене p за штуку, и в этом магазине имеется q товаров этого типа.

Формат выходных данных

В единственной строке выведите наименьшую сумму денег, которую Гриша должен взять с собой. Если нет возможности купить все товары, то надо вывести одно число -1.

Система оценивания

Представленное на проверку решение сначала проходит тестирование на тесте из примера, приведенного в условии задачи. Если на этом тесте решение участника выдает правильный ответ, то тогда это решение проверяется с использованием комплекта тестов для этой задачи (при этом баллы за каждый тест начисляются независимо). В противном случае решение участника считается неверным, и за него участнику не начисляются какие-либо баллы.

Пример входного и выходного файлов

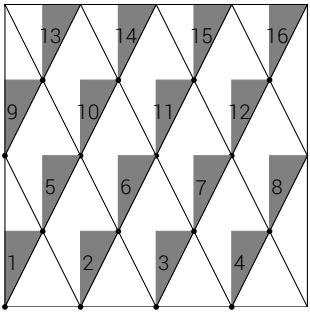
shops.in	shops.out
5 0 1 3 0 2 1 0 5 0 5 3 5 0 7 2 0 0 7 0 2 2 5 2 2 0 3 3 5 5 3 1 3 2 3 2 1 5 4 3 3 2 4 3 3 5 4 5 2 1 4 1 9 1 2 8 2 3 7 3 4 6 1	70

Решения

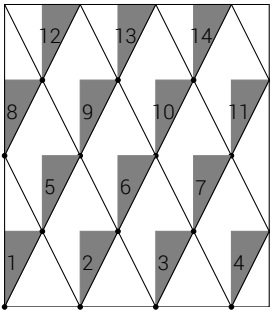
A. Ромбовидная плитка

Представим пол комнаты прямоугольником на плоскости и проведем два семейства параллельных прямых, которые разобьют плоскость на ромбы, как требуется. Рассмотрим все точки пересечений прямых внутри прямоугольника комнаты, кроме тех, что лежат на верхней или правой стороне. Обозначим через N число таких точек. К каждой из них примыкает сверху плитка или ее часть, но в любом случае содержащая один и тот же непустой фрагмент из правой нижней четверти. Поэтому требуемое количество плиток не может быть меньше N.

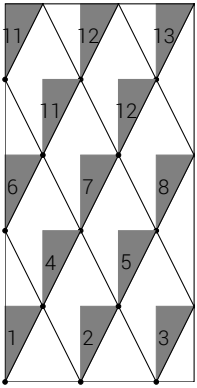
Рассмотрим сначала случай, когда длина и ширины комнаты кратны полудиагоналям плитки, соответственно: $L = n\frac{a}{2}$, $W = m\frac{b}{2}$, n, m — целые. В этом случае части плиток у нижнего и правого края, не содержащие участка правее и выше своего нижнего угла, как нетрудно убедиться, могут быть взяты из половинок, оставшихся от плиток для левого и верхнего края, независимо от четности чисел n и m. Поэтому требуемое число плиток в этом случае в точности равно N.



$n = 8, m = 4,$
$N = 16$



$n = 7, m = 4$
$N = 14$



$n = 5, m = 5$
$N = 13$

В общем случае, когда L и W не кратны полудиагоналям плитки, увеличим размеры комнаты до ближайших кратных, сведя ситуацию к выше рассмотренной. При этом новых точек пересечения прямых не появится, то есть число N не увеличится.

Число N можно вычислять при помощи разбора случаев в зависимо-

сти от четности n и m, однако они сворачиваются в общую формулу

$$N = \left\lceil \frac{nm}{2} \right\rceil,$$

где $\lceil x \rceil$ обозначает результат округления числа x вверх.

Возможно, что, повернув плитку на 90 градусов, мы получим более экономное разбиение, так что окончательный ответ можно представить в виде

$$N = \min\left(\left\lceil \frac{\lceil \frac{2L}{a} \rceil \lceil \frac{2W}{b} \rceil}{2} \right\rceil, \left\lceil \frac{\lceil \frac{2L}{b} \rceil \lceil \frac{2W}{a} \rceil}{2} \right\rceil \right).$$

B. Сортировка лыж

Сформулируем задачу математически: дана последовательность с пропусками из $2n$ чисел, где каждое число встречается не более двух раз. Для начала следует заметить, что *полная загруженность* робота для последовательности без пропусков равна сумме расстояний между первым и вторым вхождениями всех чисел. Поэтому, вместо того, чтобы считать среднюю загруженность по всем перестановкам чисел на пропусках, можно считать вклад каждого числа в ответ независимо.

Для решения первой подгруппы было достаточно просто посчитать ответ по определению, данному в условии. Для второй подгруппы можно было написать перебор, который заполняет все пропуски и считает среднюю загруженность. Эти решения могли помочь вам проверить решения для последующих подзадач.

Рассмотрим число x. Если оно уже встречается дважды, то к ответу следует сразу прибавить расстояние между вхождениями. Для всех чисел, которые не встречаются в списке, ответ одинаков и может быть посчитан один раз. Посчитаем для каждой свободной позиции i количество k_i и сумму индексов свободных позиций слева s_i. Вклад в общую сумму в том случае, когда эта позиция будет правым вхождением, равна $k_i i - s_i$. В конце эту сумму по всем свободным позициям следует поделить на $\frac{c_0(c_0-1)}{2}$, где c_0 — количество свободных позиций.

Оставшийся случай с одним вхождением разбирается аналогично. Если какое-то число встречается на позиции i, то его вклад в общий ответ

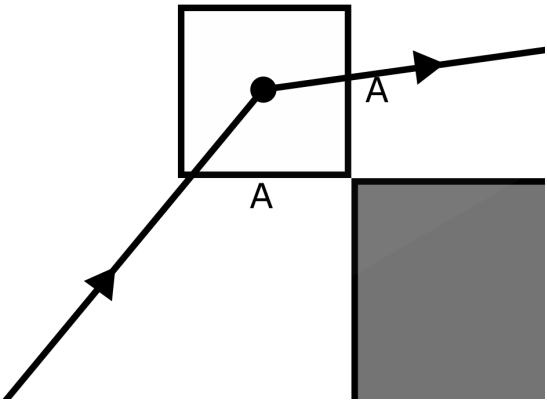
равен $\frac{1}{c_0}((k_{left} \cdot i - s_{left}) + (s_{right} - k_{right} \cdot i))$, где k_{left}, s_{left}, k_{right} и s_{right} — количества и суммы свободных позиций слева и справа от i соответственно. Описанное решение можно реализовать за $O(n)$.

C. Пожарная охрана

Для решения задачи будем использовать метод динамического программирования. Пусть A_i — это минимальные затраты на обеспечение пожарной безопасности населенных пунктов с 1 по i, при условии, что в пункте i находится пожарная станция. Очевидно, что $A_1 = C_1$. Если станция находится в пункте i, то нужно, чтобы предыдущая станция находилась на расстоянии не превышающем $2k + 1$. Поэтому $A_i = \min(A_{\max(i-2k-1,1)}, A_{\max(i-2k-1,1)+1}, \ldots, A_{i-1}) + C_i$. Ответом задачи будет $\min(A_{\max(n-k,1)}, A_{\max(n-k,1)+1}, \ldots, A_n)$.

D. Лабиринт

Вася сможет дойти оптимально до точки выхода, если он будет идти, касаясь углов многоугольника. В силу ограничений, заданных в задаче, Вася касается углов в определенных точках.



Можно предварительно для каждого угла вычислить точку касания,

затем проверить, сможет ли Вася находиться в этой точке. Такие углы назовем «правильными». Теперь нужно найти все прямые пути между точками касания правильных углов и вычислить их. Подобным образом нужно проверить прямой путь между точками касания и точками начала и конца лабиринта. Таким образом, мы можем построить граф, вершинами которого будут углы многоугольника, начало и конец лабиринта, а ребрами — длина прямого пути, если он существует. В этом графе с помощью алгоритма Дейкстры находим минимальный путь, отсюда — и минимальное время.

E. Либерлун

Пусть x – еще не доставленная доля сообщения первой станции. Представим H в виде $H = 10 \times k + r$, где k и r – целые положительные числа, причем $r < 10$. Сигнал отправляли в моменты времени $0,\ 10,\ 20, \ldots,\ 10 \times k$, то есть, было $k + 1$ отправлений, при этом, потери соответственно составляли $q,\ q^2, \ldots,\ q^{k+1}$.

Рассмотрим первую станцию. Если после времени последнего $k + 1$-ого отправления сигнала прошло менее 3 часов, то $x = q^{k+1} + q^k * p * a_1$, где $q^k * p * a_1$ означает долю, перехваченную ретрансляторами. Иначе, $x = q^{k+1}$ – часть сообщения уже дошла от ретрансляторов до адресата.

Тогда, $1 - x$ – доля сообщения, полученного адресатом. Описанным образом вычисляем для каждой станции связи долю полученного сообщения и проверяем, меньше ли S.

F. Опять ожерелье

Пусть имеются бусинки k цветов. Обозначим через $C(n)$ количество красивых цепочек из n бусинок. Очевидно, что $n \geq k$. Имеется следующая формула:

$$C(m + k + 1) = C(m + 1) + C(m + 2) + \ldots + C(m + k).$$

Если мы каким-то образом вычислим $C(k + 1)$, $C(k + 2)$, ..., $C(k + k)$, то, используя указанную формулу, можем вычислить все $C(n)$.

Разберем вычисление $C(k + 1)$. Рассмотрим k последних бусинок. Тут может быть два случая. В первом случае все бусинки разные. В этом

случае число различных цепочек, которые мы можем составить, равно $k \times k!$. Рассмотрим второй случай. В этом случае среди k последних бусинок имеются две бусинки одинакового цвета и, следовательно, какой-то цвет отсутствует. Используя это, получаем, что количество красивых цепочек равно $C_k^2 \times k \times (k-1) \times (k-2)!$. Обозначим число способов получения цепочек в первом и втором случаях соответственно через f_1 и r_1. Таким образом, $C(k+1) = f_1 + r_1$.

Рассмотрим, чему равен $C(k+2)$. Понятно, что увеличение количества способов составления цепочек зависит только от f_1. В этом случае $k+2$ бусинки можно выбрать любого цвета. В итоге получаем, что $C(k+2) = C(k+1) + (k-1) \times f_1$.

Рассмотрим вычисление $C(k+3)$. Снова разбиваем $C(k+2)$ на части, аналогичные f_1 и r_1. Обозначим их через f_2 и r_2. Нас интересует только f_2. Рассмотрим опять f_1. Разобьем его в свою очередь на части a и d. Причем $a = \frac{f_1}{k-1}$, $d = \frac{(k-2)f_1}{k-1}$. Тогда $f_2 = a + 2d$. Следовательно, $C(k+3) = C(k+2) + (k-1) \times f_2$. Обозначим через f_3 факториальную часть $C(k+3)$. $f_3 = a + 4d$. Получаем $C(k+4) = C(k+3) + (k-1) \times f_3$.

В общем случае $f_m = a + 2^{m-1}d$, $C(m+1) = C(m) + (k-1) \times f_m$. Этот процесс продолжается до тех пор, пока мы не вычислим $C(2k)$. Затем вычисления продолжаются по формуле $C(m+k+1) = C(m+1) + C(m+2) + ... + C(m+k)$. Чтобы упростить вычисления по этой формуле, представим, что $C(k+1) = a_1 + a_2 + ... + a_k$, где $a_i = C(i)$. Пусть некоторое $C(m) = b_1a_1 + b_2a_2 + ... + b_ka_k$. Тогда $C(m+1) = b_ka_1 + (b_k + b_1)a_2 + (b_k + b_2)a_2 + ... + (b_{k-1} + b_k)a_k$. Мы получили рекуррентную формулу, по которой можно вычислить любое $C(n)$.

G. Денежная система Страны Дураков

Вначале посчитаем следующую величину: f_i — минимальное количество монет, необходимое для того, чтобы набрать сумму i. Эта величина может быть посчитана с помощью динамического программирования за $O(MK)$ следующим образом: инициализируем массив $f_i = 0$ для $i = 0$ и $f_i = +\infty$ для $i > 0$. Затем, для всех монет a_k и для всех позиций i выполним обновление $f_i = \min(f_i, f_{i-a_k} + 1)$. Перебирая i в порядке возрастания, мы учитываем возможность взятия некоторой монеты несколько раз.

Теперь вернемся к решению задачи. Назовем сумму i *плохой*, если

$f_i > N$. Добавлением новой золотой монеты мы должны исправить все плохие позиции. Если стоимость золотого равна x, то плохая позиция i будет исправлена, если существует $j < i$, что $j - i$ делится на x и $f_i + \frac{i-j}{x} \leq N$. Используя это свойство, для каждой плохой позиции i можно составить список всех стоимостей x, что позиция i будет исправлена. Для этого переберем все хорошие позиции $j < i$, переберем все делители $i - j$, проверим, что значение f_i после обновления становится не больше N.

Довольно известный факт, что $\sum_{i=1}^{n} d(i) = O(n \log n)$, где $d(n)$ — число делителей числа n. Это значит, что для фиксированного i нам придется рассмотреть лишь $O(M \log M)$ чисел. После этого множества стоимостей для всех вершин можно пересечь и взять минимум и максимум. Время работы второй части и полного решения есть $O(M^2 \log M)$.

H. Магазины

Эту задачу можно разбить на две подзадачи. Первая подзадача должна находить минимальное количество денег, которое потребуется на транспортные расходы, чтобы объехать определенное множество магазинов. Вторая же подзадача должна выдавать минимальную сумму денег затрачиваемую для покупок товаров из определенного множества магазинов. Таким образом, для определенного множества магазинов можно вычислить минимальную сумму денег, которая требуется чтобы купить товары только в магазинах из этого множества. Это будет сумма денег затрачиваемых на дорогу и на товары. Перебрав всевозможные множества нужно найти минимальную сумму.

Для решения первой подзадачи рассчитаем матрицу A минимальных расстояний. Это можно сделать с помощью алгоритма Флойда-Уоршелла.

Далее приведем задачу к динамическому программированию. В элементе $D[mask][last]$, двумерного массива D, будем хранить минимальную сумму денег для состояния $[mask][last]$, которая потребуется, чтобы объехать все магазины из множества, которая определяет *mask*, и закончить путь в магазине с индексом *last*. Если i-ый бит индекса mask будет 1, то это значит, что i-ый магазин входит в наше множество. Для простоты будем говорить, что *mask* — множество. Все $D[mask][last]$, для которых

mask состоит только из одного элемента *last*, будут равны 0.

Чтобы найти значение элемента $D[mask][last]$ сначала определим *prevMask*. Так как мы должны закончить путь в магазине *last*, то во множестве, предыдущего состояния с которого мы пришли, не будет только элемента *last*. Значит $prevMask = mask \backslash last$. Пусть *prevLast* входит в *prevMask*, тогда если мы приходим из состояния $[prevMask][prevLast]$, то общая стоимость будет равна $D[prevMask][prevLast] + A[prevLast][last]$. Перебрав все *prevLast* из *prevMask* находим минимальную стоимость поездки. Таким образом для определенного множества *mask* можно вычислить минимальную стоимость поездки просто взяв минимум из всех $D[mask][last]$.

Рассмотрим частный случай для решения второй подзадачи. Пусть у нас будет только один продукт. В случае нескольких продуктов, стоимость для каждого вычисляется по отдельности и просто суммируется. Пусть мы побывали во всех магазинах из множества *mask*. Отсортируем магазины по возрастанию цен. Пробегаемся по отсортированному списку магазинов и покупаем до тех пор, пока не получим требуемое количество продуктов.

ИТОГИ / FINAL STANDINGS

	Имя /Name	Страна / Country	Первый день / First day					Второй день / Second day					ИТОГО TOTAL	НАГРАДА AWARD
			A	B	C	D	Σ	E	F	G	H	Σ		
1	СИТАРУ Богдан SITARU Bogdan	Румыния Romania	25	100	100	72	*297*	100	100	100	100	*400*	**697**	1DG
2	КОНСТАНТИН-БУЛИГА Стефан CONSTANTIN-BULIGA Ștefan	Румыния Romania	0	100	100	92	*292*	100	100	100	100	*400*	**692**	1DG
3	ПЕТРЕСКУ Александру PETRESCU Alexandru	Румыния Romania	0	100	100	44	*244*	100	100	100	100	*400*	**644**	2DG
4	МОРОЯНУ Теодор Пьер MOROIANU Theodor Pierre	Румыния Romania	25	100	100	24	*249*	100	60	100	100	*360*	**609**	2DG
5	ЛИ Джефри Чун Хин LEE Jeffrey Chun Hean	Сингапур Singapore	25	100	100	88	*313*	70	20	100	0	*190*	**503**	3DG, SA
6	АЙТКАЛИ Акылбек AYTKALI Akylbek	Казахстан Kazakhstan	0	100	100	0	*200*	94	20	100	88	*302*	**502**	3DG
7	МАГЕРЯНУ Ливия MĂGUREANU Livia	Румыния Romania	25	100	100	32	*257*	23	20	100	12	*155*	**412**	3DG
8	БУЛАТОВ Василий Алквиадович BULATOV Vasiliy	Россия Russia	100	20	25	12	*157*	76	20	0	0	*96*	**253**	HML
9	МАКСИМОВ Алексей Николаевич MAKSIMOV Aleksey	Россия Russia	0	100	25	0	*125*	94	20	0	0	*114*	**239**	HML
10	БОТАКАНОВ Торежан BOTAKANOV Torezhan	Казахстан Kazakhstan	25	40	0	0	*65*	52	20	16	0	*88*	**153**	
11	ПОПОВ Кирилл Евгеньевич POPOV Kirill	Россия Russia	0	40	0	8	*48*	94	0	0	0	*94*	**142**	

12	ГАВРИЛЬЕВ Валерий Валерьевич GAVRILIEV Valeriy	Россия Russia	0	0	25	0	*25*	100	0	0	0	*100*	**125**	
13	АРМЕНАКЯН Карен Арменакович ARMENAKYAN Karen	Россия Russia	100	20	0	0	*120*	0	0	0	0	*0*	**120**	
14	БОСОЕВА Эльвира Ивановна BOSOEVA Elvira	Россия Russia	0	0	100	0	*100*	0	0	0	0	*0*	**100**	
15	ЗАХАРОВ Алексей Викторович ZAKHAROV Alexey	Россия Russia	0	0	0	0	*0*	70	0	0	0	*70*	**70**	
16	ПАВЛОВ Степан Павлович PAVLOV Stepan	Россия Russia	0	0	0	28	*28*	18	0	0	0	*18*	**46**	
17	МАЛЬЦЕВА Юлия Игоревна MALTSEVA Yulia	Россия Russia	0	0	0	8	*8*	0	0	0	0	*0*	**8**	
18	ХАНИН Виктор Александрович KHANIN Viktor	Россия Russia	0	0	0	0	*0*	0	0	2	0	*2*	**2**	
19	СЕРИКОВА Альбина SERIKOVA Albina	Казахстан Kazakhstan	0	0	0	0	*0*	0	0	0	0	*0*	**0**	

18.07.2017

НАГРАДЫ / AWARDS: 1DG — диплом 1 степени / 1st degree diploma,
2DG — диплом 2 степени / 2nd degree diploma,
3DG — диплом 3 степени / 3rd degree diploma,
HML — почетная грамота / Honorable mention letter,
SA — спецприз за лучший результат 1 дня / special award for best results of 1st day

Содержание | Contents

Problems 1
A. Rhomboid tiles 1
B. Skis sorting 2
C. Fire stations 4
D. Labyrinth 5
E. Liberloun 6
F. Necklace, again 7
G. Monetary system of the Land of Fools 8
H. Shopping 9
Solutions 12
A. Rhomboid tiles 12
B. Skis sorting 13
C. Fire stations 14
D. Labyrinth 14
E. Liberloun 15
F. Necklace, again 15
G. Monetary system of the Land of Fools 16
H. Shopping 17
Задачи 18
A. Ромбовидная плитка 18
B. Сортировка лыж 19
C. Пожарная охрана 21
D. Лабиринт 22
E. Либерлун 24
F. Опять ожерелье 25
G. Денежная система Страны Дураков 26
H. Магазины 27

Решения 30
A. Ромбовидная плитка 30
B. Сортировка лыж 31
C. Пожарная охрана 32
D. Лабиринт 32
E. Либерлун 33
F. Опять ожерелье 33
G. Денежная система Страны Дураков 34
H. Магазины 35

Итоги олимпиады / Final Standings **37**

XXIV Международная олимпиада школьников по математике, физике, химии и информатике «Туймаада». ИНФОРМАТИКА. – Якутск, 2017.

Сборник содержит задачи XXIV Международной олимпиады «Туймаада» по информатике, а также возможные варианты решений. Олимпиада проводилась 17–18 июля 2017 года в г. Якутске. Олимпиада прошла в два дня, в каждый из которых участникам было предложено решить четыре задачи.

This booklet contains the problems and possible solutions of the 24rd *Tuymaada* International Olympiad in informatics. The olympiad took place on 17–18 July 2017 in Yakutsk, Russia. The participants were invited to solve four problems on each of the two competition days.

АВТОРЫ | *AUTHORS*

Александр Викторович ПАВЛОВ **A** *Alexander PAVLOV*
Северо-Восточный федеральный университет *Northeastern Federal University*

Артем Тарасович ВАСИЛЬЕВ **B** *Artem VASILYEV*
Университет ИТМО *ITMO Univeristy*

Кирилл Борисович ЧИХАЧЁВ **C** *Kirill TCHIKHATCHOV*
физико-математический лицей №239, Санкт-Петербург
Physics and Mathematics Lyceum No. 239, St Petersburg

Алексей Юрьевич ХОХОЛОВ **D** *Aleksey KHOKHOLOV*
СВФУ *NEFU*

Григорий Александрович СПИРОВ **E** *Grigory SPIROV*
СВФУ *NEFU*

Юрий Саввич АНТОНОВ **F** *Yuri ANTONOV*
СВФУ *NEFU*

Сергей Геннадьевич ВОЛЧЁНКОВ **G** *Sergey VOLCHENKOV*
Ярославский государственный университет им. П. Г.Демидова, Ярославль
P. G. Demidov Yaroslavl State University, Yaroslavl

Айтал Викторович ДЬЯКОНОВ **H** *Aytal DYAKONOV*
СВФУ *NEFU*