

Problem A. Amazing Chef Masters

Input file: **standard input**
Output file: **standard output**
Time limit: 1 second
Memory limit: 256 megabytes

The Amazing Chef Masters (ACM) have annual contests to determine the rankings of the members. Each chef has a skill level that could change over the years. Since each chef is a different individual, no two chefs will ever have the same skill level. The annual rankings are always based on the skill levels, so no ties will occur. More precisely, if chef A has a higher skill level than chef B , then chef A will have a better rank than chef B .

Usually the chefs would have a detailed record of the skill levels of the members each year, however, they were attacked by the Attacking Cactus Monsters (ACM)! These monstrous beings stole all of the records except the annual rankings.

Now the chefs want to regain some information about their skill levels. Given the rankings of two consecutive years, the chef masters would like to know the maximum possible number of chefs whose skill levels did not change.

Input

The first line of input contains a single integer n ($1 \leq n \leq 10^5$), the number of chefs in ACM.

The second line of input contains n distinct space-separated integers a_i ($1 \leq a_i \leq n$), describing the rank of the chefs in the first year. The a_i th chef has rank i in the first year.

The third line of input contains n distinct space-separated integers b_i ($1 \leq b_i \leq n$), describing the rank of the chefs in the second year. The b_i th chef has rank i in the second year.

Output

Print one integer that is the maximum possible number chefs whose skill levels did not change.

Example

standard input	standard output
4 2 3 1 4 3 2 1 4	3

Note

This is the explanation for the sample. In the first year, chef 2 is ranked first, chef 3 is ranked second, chef 1 is ranked third, and chef 4 is ranked fourth. In the second year, chef 3 is ranked first, chef 2 is ranked second, chef 1 is ranked third, and chef 4 is ranked fourth. We can get the answer 3 by letting chefs 1, 3, and 4 have skill levels that remained unchanged, while chef 2's skill decreases.

Problem B. Badminton

Input file: **standard input**
Output file: **standard output**
Time limit: 2 seconds
Memory limit: 256 megabytes

A little green Cactus is secretly in charge of a badminton club, and he wants to enter four members in the next tournament. There is a total of n members in the club. Each member has a playing style $t_i \in \{1, 2, 3\}$ and a strength a_i . To make the team of four well rounded, Cactus wants at least one member of each playing style on the team. Cactus also wants his team to win the tournament, so he will only consider teams whose strength is at least k . The strength of a team is the average of the strengths of the four individuals.

Cactus wants to know the number of distinct teams of four whose strength is at least k and contains at least one member of each of the three playing styles. Two teams are considered different if there exists a member who is in exactly one of them.

Input

The first line contains two space-separated integers n and k ($2 \leq n \leq 3000$; $1 \leq k \leq 10^5$), the number of members in the club, and the minimum acceptable team strength respectively.

Each of the next n lines contains two space-separated integers t_i and a_i ($1 \leq t_i \leq 3$; $1 \leq a_i \leq 10^5$), describing the i th member. That is the i th member has playing style t_i and strength a_i .

Output

Print a single integer representing the number of distinct teams that satisfy the two conditions.

Example

standard input	standard output
6 3 1 2 1 2 2 2 3 1 3 3 3 7	5

Problem C. Cakes

Input file: **standard input**
Output file: **standard output**
Time limit: 1 second
Memory limit: 256 megabytes

Our friend, the little Cactus, has n stacks of cakes and an integer k . Initially, the n stacks of cakes have sizes $1, 2, \dots, n$, that is the i th stack has i cakes. For aesthetic reasons, Cactus wants to make a large stack of cakes. Cactus can perform up to 10^5 operations on the n stacks.

Let s_i denote the size of the i th stack. Each operation consists of 3 steps.

1. Cactus chooses an integer a where $2 \leq a \leq k$.
2. Cactus chooses two stacks A and B , where $s_A \geq (a - 1) \cdot s_B$.
3. Cactus moves $(a - 1) \cdot s_B$ cakes from stack A to stack B .

Now Cactus wants to know the size of the largest possible stack he could make. However, Cactus likes surprises, so he wants you to tell him the operations he should perform to make the largest possible stack.

Input

The first line contains two space separated integers n and k ($1 \leq n \leq 200$, $2 \leq k \leq 12$).

Output

In the first line, output a single integer x representing the number of operations Cactus should perform.

For each of the next x lines, output two space separated integers y_i , z_i , and t_i , representing an operation where Cactus should choose the integer t_i , and move cakes from stack y_i to stack z_i .

Your solution will be considered correct if $0 \leq x \leq 10^5$, each operation is valid, and the largest possible stack that could be made this way is one of the n stacks after performing all the operations.

If there are multiple solutions, output any of them.

Example

standard input	standard output
6 3	7 6 3 3 4 2 3 3 5 2 2 1 2 3 1 3 1 2 2 2 5 2

Problem D. Dessert First Strategy

Input file: **standard input**
Output file: **standard output**
Time limit: 6 seconds
Memory limit: 256 megabytes

Dessert First Strategy (DFS) refers to the strategy to monopolize the dessert before other people have a chance to get to it. This usually involves an elaborate plan with various positions that should be occupied by units.

More precisely, there are n positions that form a tree. Each position is one of k roles, labeled from 1 to k inclusive. You have an unlimited number of each of the two available types of units (tourist and cactus) to assign to the positions. However, you must assign the same unit within each role. That is, if you assign a tourist to a position of role 1, then you cannot assign a cactus to another position of role 1.

Each of the two units has a list of valid roles they can take (for example, cacti cannot take the role that is messenger pigeon). It is guaranteed that each role can be taken by at least one of the two units. Now, it is always possible to assign units to positions based on the aforementioned restrictions.

The n positions form a tree, and each edge has an effectiveness value c_i . This effectiveness can be gained if the two units assigned to the positions adjacent to the edge are the same type, in which case the effectiveness of the edge is c_i . If the two units are not the same type, then the effectiveness of the edge is 0. The effectiveness of the strategy is the sum of the effectivenesses of each edge.

Given the number of positions, the number of roles, and the shape of the tree, calculate the maximum possible effectiveness that could be obtained by assigning units to the roles optimally.

Input

The first line of input contains four space-separated integers n, k, p, q ($1 \leq n \leq 10^5; 1 \leq p, q \leq k \leq 200$), representing the number of positions, the number of communications, and the number of roles respectively.

The second line of input contains p distinct space-separated integers each between 1 and k inclusive, representing the roles that the tourists can take.

The third line of input contains q distinct space-separated integers each between 1 and k inclusive, representing the roles that the cacti can take.

The fourth line contains n space-separated integers. The i th integer r_i represents the role of the i th position.

Each of the next $n - 1$ lines contains three space-separated integers a_i, b_i , and c_i ($1 \leq a < b \leq n; 1 \leq c_i \leq 10^4$), describing the i th edge that is between positions a_i and b_i with a possible effectiveness gain of c_i .

Output

Output a single integer representing the maximum effectiveness that can be gained.

Example

standard input	standard output
5 3 2 2 1 2 2 3 1 1 2 1 3 1 2 2 2 3 3 3 4 1 4 5 42	6

Note

In the first sample, it is optimal to have the tourists take roles 1 and 2 while the cacti take role 3. This means we get the following effectiveness gains:

- 2 effectiveness from 1 and 2
- 3 effectiveness from 2 and 3
- 1 effectiveness from 3 and 4

Problem E. Easy

Input file: **standard input**
Output file: **standard output**
Time limit: 1 second
Memory limit: 256 megabytes

E for Easy.

Clearly, E for Easy sounds better than D for Deasy, F for Feasy, G for Geasy, or other such nonsense. Perhaps less clear is that E for Easy also sounds better than E for Empossible. It follows that the only logical title for this problem is Easy. Whether this problem is actually easy is up to you.

It is 23:16. The relentless rain pelting my window keeps me awake while I write this unintelligible mess. Although my fingers are moving across the k keys of the keyboard and pressing r keys that somehow translate to s complete albeit meaningless words, my mind is elsewhere. The rain outside just made me remember that nightmare I had w days ago. I was chasing my sanity down a network of p dank alleyways of some ghost town in the middle of a desert. My sanity was somewhere ahead of me, just outside of my reach. But this was only the prelude to the nightmare. As I chased my fleeing sanity into a tavern, I came face to face with a 40cm tall messenger pigeon. Frightened out of my wits, I awoke to the sound of birds chirping as if nothing had happened. Unfortunately, my sanity is still hiding inside the tavern.

The tavern has n rooms that are connected by m bidirectional halls. The n rooms are uniquely labeled with integers from 1 to n inclusive. The pigeon is in room 1, and my sanity is in room n . Each of the m halls require a certain amount of time to traverse. Even someone like me is able to follow these halls from 1 to n in the minimum amount of time possible. However, there are traps in each hall! Now, each hall only has an interval of time when it is safe to be in it. Should one set foot in a hall when it is unsafe, unthinkable things will happen.

I am unable to solve such a difficult problem, so it is up to you to determine the shortest amount of time it takes to go from the room where the pigeon is at to the room where my sanity is hiding. It is now time 0, let the search begin!

Input

The first line contains two space-separated integers n and m ($2 \leq n \leq 2 \cdot 10^5$; $1 \leq m \leq 2 \cdot 10^5$), representing the number of rooms and halls respectively.

Each of the next m lines contains five space-separated integers a_i , b_i , c_i , d_i , and t_i ($1 \leq a_i, b_i \leq n$; $0 \leq c_i < d_i \leq 10^9$; $1 \leq t_i \leq 10^9$), describing the i th hall. This means the i th hall is between rooms a_i and b_i , it is safe to be in the hall from time c_i to time d_i inclusive and it takes t_i minutes to traverse the hall (in either direction). Two rooms could be connected by multiple halls. It is guaranteed that $a_i \neq b_i$.

Output

Output a single integer representing the shortest time it takes to go from room 1 to room n if it is possible to safely reach room n , or output -1 otherwise.

Examples

standard input	standard output
3 3 1 2 4 10 3 2 3 19 42 5 1 3 1 7 7	24
3 2 1 2 14 17 1 2 3 12 15 1	-1

Note

In the first sample, the only way to reach the destination is by taking the halls from 1 to 2 and from 2 to 3. The journey can be as follows.

1. Time 0-4, wait 4 minutes in room 1
2. Time 4-7, take the hall from 1 to 2
3. Time 7-19, wait 12 minutes in room 2
4. Time 19-24, take the hall from 2 to 3

Problem F. Farmer Faul

Input file: **standard input**
Output file: **standard output**
Time limit: 5 seconds
Memory limit: 256 megabytes

Faul decided to become a farmer, and he grows bamboo!

Faul has n bamboo stalks numbered from 1 to n , and he can do three operations on them. He can cut and harvest all of the bamboo at a certain height, order a single bamboo stalk to grow, or use magic to make all bamboo stalks grow.

Faul knows exactly what his next q operations will be. Each operation will be one of the three types.

- **CUT h .** Faul cuts all the bamboo stalks at height h ($1 \leq h \leq 10^6$). More precisely, if a stalk is higher than height h , then the part higher than h will be cut off and harvested. For this query, you should output one integer, on a line by itself, representing the total length of bamboo harvested by this cut.
- **GROW i x .** Faul orders the i th stalk to grow by x ($1 \leq i \leq n; 1 \leq x \leq 10^6$). More precisely, if stalk i was at height k , then this query makes it grow to height $k + x$.
- **MAGIC y .** Faul uses magic to make all bamboo stalks grow by y ($1 \leq y \leq 10^6$).

Since Faul has more important things to tend to (such as badminton), he wants you to help him predict the amount of bamboo he will harvest after each cut.

Input

The first line of input contains two space-separated integers n and q ($1 \leq n \leq 10^6; 1 \leq q \leq 10^6$).

The second line of input contains n space-separated integers h_i ($1 \leq h_i \leq 10^6$), where h_i is the initial height of the i th bamboo stalk.

Each of the next q lines contains a query. Each query is one of the three aforementioned types.

Output

For each CUT query, output a single integer, on a line by itself, that is the total length of bamboo harvested by that cut.

Example

standard input	standard output
4 5	3
1 2 3 3	7
GROW 3 1	0
CUT 2	
MAGIC 2	
CUT 2	
CUT 7	

Note

In the first sample, we do the following.

- **GROW 3 1.** We grow the third bamboo by 1. Now the heights are 1 2 4 3.
- **CUT 2.** We cut everything at height 2, so we cut 2 and 1 units from the third and fourth bamboos respectively. We output 3 since $1 + 2 = 3$. Now the heights are 1 2 2 2.

- MAGIC 2. We grow all the bamboos by 2. Now the heights are 3 4 4 4.
- CUT 2. We cut everything at height 2. We cut 1 unit from bamboo 1, and we cut 2 units from bamboos 2, 3, and 4. We output 7 since $1 + 2 + 2 + 2 = 7$. Now the heights are 2 2 2 2.
- CUT 7. We cut everything at height 7. No bamboo is higher than 7 so nothing is cut. We output 0 and the heights remain unchanged.

Problem G. Game

Input file: **standard input**
Output file: **standard output**
Time limit: **1 second**
Memory limit: **256 megabytes**

Our little Cactus friend is back with an interesting game! This game features a wheel which consists of n compartments connected in a loop. Each compartment has a hole leading out of the loop. To play, the ball is randomly put into one of the compartments, and the player simply watches it roll around the loop. After some time, the ball will exit through one of the holes, and the game ends. Once the ball exists the wheel, the player is given a score based on the hole from which it exited, and the compartments that the it had visited (counting multiplicity).

More precisely, if the ball is at compartment i for $1 \leq i < n$, it may exit the wheel with probability p_i or it may move on to compartment $i + 1$ with probability $1 - p_i$. If the ball is at compartment n , it may exit the wheel with probability p_n or move on to compartment 1 with probability $1 - p_n$. The player begins with a score of 0. Each time the ball visits compartment i , the player gains a_i score. If the last compartment the ball was in before it exited the loop was compartment j , the player gains an additional b_j score.

All that is left is for Cactus to assign probabilities. To make things interesting, he wants to assign probabilities so that $0.01 \leq p_i \leq 0.99$. Since the score counting contraption cannot handle very large numbers, Cactus wants to minimize the expected score. Your task is to calculate the minimum possible expected score while the probabilities satisfy the given restriction. You may assume that the initial compartment is chosen uniformly at random.

Input

The first line of input contains a single integer n ($2 \leq n \leq 500$), the number of compartments in the wheel.

Each of the next n lines contains two space-separated integers a_i and b_i , ($1 \leq a_i, b_i \leq 10^6$), describing the i th compartment. a_i is the score gained each time the i th compartment is visited, and b_i is the score gained if the i th compartment is the last compartment visited.

Output

Print, on a single line, the minimum possible expected score. Your answer will be considered correct if it is within 10^{-6} absolute or relative error.

Examples

standard input	standard output
3 1 100 1 100 1 1	4.0227937347
2 420 3 1 1	214.6262626263

Problem H. Hiking

Input file: **standard input**
Output file: **standard output**
Time limit: 1 second
Memory limit: 256 megabytes

Cactus is hiking with his friends in the Great Bear Rainforest. As they were too busy practicing ACM problems the night before, they forgot to bring bug spray, emergency shelter, whistles, and most importantly, food! After several dozen hours, Cactus and his friends become tired and hungry. Suddenly, Cactus spots a berry on a bush. Just as he stretches out his hand to pick it, he hears a low grumbling noise behind him. It's a Great Bear, and it's hungry... for berries!

Thinking quickly, Cactus pulls out his M4 Carbine with a M203 grenade launcher attachment and blasts the Great Bear into the Spirit Realm. Unfortunately, he forgot to stock up on bullets and grenades, so he has no projectiles left. In his panic, Cactus asks the Geometry God to estimate the probability of their survival. Clearly this is an insult to Geometry's intelligence and omnipotence, as he immediately gives Cactus the exact probability of their survival as a function of the amount of time it takes them to leave the Great Bear Rainforest.

$$P(t) = \max \left\{ 0, 100 - \sum_{k=1}^t \frac{e^{-k} k^{2+\sqrt{k}} + e^{-k} k^{\sqrt{k}} \sin k}{1 - e^{-k}} \right\}.$$

In this equation, e is Euler's constant and t is the time it will take Cactus and his friends to escape from the Great Bear Rainforest. $P(t)$ gives the percent probability that the group manages to leave untraumatized.

Given this equation, Cactus is faced with the impossible task of finding t . Luckily, Cactus knows exactly how long it will take them to exit the Great Bear Rainforest from his primal instincts as a Great Bear warrior in his previous life.

Due to a lack of food and safety blankets, Cactus is too hungry to think straight. Help Cactus and his friends find the probability that they will survive, and not be bearied by barely beary barbearian berry bears bearfacedly bearing down on them!

Input

The only line of input contains one integer t ($1 \leq t \leq 10^6$).

Output

Output $P(t)$ for the given time. Your answer will be considered correct if the absolute error is at most 10^{-3} .

Example

standard input	standard output
1	98.9283067805

Problem I. LASER SHIELDS

Input file: standard input
Output file: standard output
Time limit: 1 second
Memory limit: 256 megabytes

Our green Cactus friend is in danger again! A hostile troop of cacti is trying to attack this lonely cactus. Luckily, Cactus happens to have N LASER SHIELDS. However, Cactus can only activate up to k LASER SHIELDS, and he wants you to help him determine the best way to use k LASER SHIELDS.

Each LASER SHIELD has an activation time t_i at which it can be activated. If it is not activated at that time, it loses its power and becomes useless forever. Each LASER SHIELD has an initial power p_i and a corrosion rate r_i . When a LASER SHIELD is activated, it will have the initial power p_i and lose power at the rate of r_i power units per second. When the power of a LASER SHIELD drops below 0, it becomes inactive and cannot be used again.

Define the defense power at a specific time to be the maximum power over all activated shields. The defense power is 0 if there are no active shields. Define the protection of Cactus to be the minimum defense power at any time during the attack.

Cactus used w3m to look into the future, seeing that the hostile cacti begin their attack at time t_s and end their attack at time t_e . Your task is to determine the maximum possible protection of Cactus.

Input

The first line contains a two space-separated integers n and k ($0 \leq n, k \leq 10^5$), the total number of LASER SHIELDS and the number of LASER SHIELDS that Cactus can activate respectively.

Each of the next n lines contains three space-separated integers t_i, p_i , and r_i ($0 \leq t_i \leq 10^9; 1 \leq p_i, r_i \leq 10^9$), describing the i th LASER SHIELD.

The last line of input contains two space-separated integers t_s and t_e ($0 \leq t_s < t_e \leq 10^9$), the start and end times of the attack respectively.

Output

Print, on a single line, the maximum possible protection that can be achieved.

Example

standard input	standard output
4 3 2 10 4 3 6 1 7 3 1 8 12 1 4 9	2

Note

In the first sample, it is optimal to activate the second, third, and fourth shields.

Problem J. Journey

Input file: **standard input**
Output file: **standard output**
Time limit: 5 seconds
Memory limit: 256 megabytes

The little green Cactus is on his daily run through the cactus desert. As usual, he is coding problem J as he runs. However, he forgot his charger today, and his computer is running out of battery. Now Cactus must run from point A (his current location) to point B (the location of his charger).

There are n oddly rectangular cacti in the desert, and Cactus cannot run through these cacti. No two cacti overlap, but they may touch at a point or on a line segment. Cactus cannot run through the points where two cacti touch. Find the minimum distance that Cactus must run to get from A to B .

Input

The first line of input contains a single integer n ($0 \leq n \leq 100$), the number of cacti.

Each of the next n lines contains four space-separated integers a_i , b_i , x_i , and y_i ($-10^6 \leq a_i < x_i \leq 10^6$; $-10^6 \leq b_i < y_i \leq 10^6$), describing the bottom left and the top right corners of the cactus (on the Cartesian plane).

The last line of input contains four space-separated integers A_x , A_y , B_x , B_y ($-10^6 \leq A_x, A_y, B_x, B_y \leq 10^6$), describing point A and point B . It is guaranteed that A and B do not lie in the interior or on the boundary of a cactus.

Output

Print the minimum distance Cactus must run, or -1 if he cannot run from A to B . The answer will be considered correct if the relative or absolute error is at most 10^{-6} .

Example

standard input	standard output
3 0 0 1 1 0 1 1 2 1 -1 5 0 0 -1 2 1	5.4142135624