1.Codekart

Problem Description

Codu wants to create a shopping application. The application would sell only SHIRT and SHOE and have a cost that can be modified based on market needs. This application should allow users in two roles, viz. store manager(SM) and shopper(S).

Codu wants to test the app. He wants the application to execute a few commands and print the output.

Following is the list of allowed **commands**:
CMD SM ADD [ITEM NAME] [ITEM QTY] - adds the given quantity of the item to the inventory and prints(returns) the quantity added, otherwise prints -1 when there is any error or invalid input. Item qty can only be whole numbers > 0.
CMD SM REMOVE [ITEM NAME] - removes the item from the inventory, returns and prints -1 when there is an error, otherwise prints(returns) 1.
CMD SM GET_QTY [ITEM NAME] - returns and prints the currently available quantity for the item in the inventory, otherwise prints(returns) 0 in case the item is not found.
CMD SM INCR [ITEM NAME] [ITEM QTY] - adds the given quantity of the item to the inventory and prints(returns) the quantity added, otherwise prints(returns) -1 when there is any error or invalid input. item qty can only be whole numbers > 0.
CMD SM DCR [ITEM NAME] [ITEM QTY] - removes the given quantity of the item from the inventory and prints(returns) the quantity added, otherwise prints(returns) -1 when there is any error or invalid input. item qty can only be whole numbers > 0.
CMD SM SET_COST [ITEM NAME] [COST] - sets the cost of the item, returns the value, otherwise prints -1 in case of any errors or invalid input. Cost must be decimal.

CMD S ADD [ITEM NAME] [ITEM QTY] - adds the given quantity of the item to the shopping cart and prints(returns) the quantity added, otherwise prints -1 when there is any error or invalid input. Item qty can only be whole numbers > 0.
CMD S REMOVE [ITEM NAME] - removes the item from the shopping cart, returns and prints -1 when there is an error, otherwise prints(returns) 1.
CMD S INCR [ITEM NAME] [ITEM QTY] - adds the given quantity of the item to the shopping cart and prints(returns) the quantity added, otherwise prints(returns) -1 when there is any error or invalid input. item qty can only be whole numbers > 0.
CMD S DCR [ITEM NAME] [ITEM QTY] - removes the given quantity of the item from the shopping cart and prints(returns) the quantity added, otherwise prints(returns) -1 when there is any error or invalid input. item qty can only be whole numbers > 0.
CMD S GET_ORDER_AMOUNT - gets the total price of the items in the cart, returns the value, otherwise prints -1 in case of any error or invalid input. The total amount should be rounded and printed up to two decimal places.
NOTE- Increment and Decrement operations are only possible when the item is already in the inventory or cart. If increment or decrement is attempted on items that do not exist in the cart, then the command should return and print -1.

If an attempt is made to add an item that is already in the inventory or cart, such operations should result in an error and must return and print -1.

If an item which is present in the cart of inventory is removed using *remove* command and an increment or decrement operation is performed on it, such operations should result in an error and must return and print -1.

If any item quantity after decrement becomes zero, the same is removed from the corresponding inventory or cart. Performing increment or decrement operation after such a previous decrement operation, should result in an error and return -1.

You need to think of other similar error conditions while implementing the solution.

Please note at the beginning of a test case or command set, both the inventory as well as the cart is empty.

Here,

SM= STORE MANAGER
S= SHOPPER

You are required to create the application for Codu to manage the shopping kiosk.

The first line of input **T**, gives the number of test cases.

Each test set is a set of commands, which ends with "END" string.

Each command in a test case is on a new line

Constraints

1<=T<=10

Input

First line contains an integer T, which denotes the number of test cases

Second line onwards, there will be commands until we receive END command. Any command after the END command belongs to next test case.

For command format refer to Example section.

Output

Print output of every command. (Print double value for command SET_COST(rounding to one decimal places) and GET_ORDER_AMOUNT(rounding to two decimal places))

Time Limit

1

Examples

Example 1

Input

1

CMD SM SET_COST SHOE 5

CMD SM SET_COST SHIRT 10

CMD SM ADD SHOE 5

CMD SM ADD SHIRT 10

CMD SM DCR SHIRT 5

CMD SM INCR SHOE 5

CMD SM GET_QTY SHIRT

CMD SM GET_QTY SHOE

CMD SM REMOVE SHIRT

CMD SM GET_QTY SHIRT

CMD S ADD SHOE 2

CMD S INCR SHOE 2

CMD S DCR SHOE 1

CMD S GET_ORDER_AMOUNT

END

Output

5.0

10.0

5

10

5

5

5

10

1

0

2

2

1

15.00

Explanation :

From commands "**CMD SM SET_COST SHOE 5**" and "**CMD SM SET_COST SHIRT 10**"

We are successfully setting the cost as 5.0 and 10.0 respectively.

From next commands "**CMD SM ADD SHOE 5**" and "**CMD SM ADD SHIRT 10"**

Quantity of 5 shoes and 10 shirts has been successfully added to the inventory.

From next commands "**CMD SM DCR SHIRT 5**"" and "**CMD SM INCR SHOE 5"**

Shirt quantity is decremented by 5 and shoe quantity is incremented by 5. This leaves us with 5 shirts and 10 shoes in the inventory.

From next commands "**CMD SM GET_QTY SHIRT**" and "**CMD SM GET_QTY SHOE"**

We are getting the quantity of shirt and shoe, which is 5 and 10 respectively.

From next command "**CMD SM REMOVE SHIRT"**

Shirt is removed from the inventory and hence 1 is printed.

From next command "**CMD SM GET_QTY SHIRT"**

We are querying the quantity of shirt, which is 0 as it was removed in the previous command.

From next command "**CMD S ADD SHOE 2"**

Shopper adds two shoes to the cart hence 2 is printed.

From next commands "**CMD S INCR SHOE 2**" and "**CMD S DCR SHOE 1**"

The user increments these shoes by 2 and then decrements by 1 hence 2 and 1 are printed. SO current shoes in cart= 2+2-1=3

From next commands "**CMD S GET_ORDER_AMOUNT**"

The next command asks to print order amount or cart value, which is the cost of shoes * the number of shoes=5*3=15 hence 15.00 is printed by rounding to two decimal places.

2.Signal Connection

Problem Description

You have been given longitude and latitude of locations from where the channels are going to be broadcasted. You also get the height of tower from which these channels are broadcasted. Moreover, you have been given the location of your friend Jason. You have to calculate how many connections he can make to the towers. Take Radius of earth= 6371 KM.

All the computation has to be accurate up to 6 digits after the decimal point.

Constraints

1 <= N < 10^5

Input

First line contains integer N denoting the number of locations from where the channel is going to be broadcasted

Second line contains N space-separated decimal values denoting latitudes

Third line contains N space-separated decimal values denoting longitudes

Fourth line contains N space-separated integer values denoting the height of tower from which channels are broadcasted

Fifth Line contains two space-separated decimal values denoting latitude, longitude of Jason's location

Output

Print the number of channels Jason can connect with

Time Limit

1

Examples

Example 1

Input

2

19.076090 17.387140

72.877426 78.491684

2 1

18.516726 73.856255

Output

1

Explanation

First latitude and longitude is Mumbai and second is for Hyderabad from where the channel signals are broadcasted. Jason is in Pune. According to signal strength Jason will only be able to connect Mumbai tower.

Example 2

Input

2

28.644800 22.572645

77.216721 88.363892

5 7

48.864716 2.349014

Output

0

Explanation

First latitude and longitude is Delhi and second is for Kolkata from where the channel signals are broadcasted. Jason is in Paris. According to signal strength Jason will not be able to connect any tower.

3.Best Sequence

Problem Description

Some of the keys of Ajith's Laptop's Keyboard are damaged and he is not able to type those keys. He has to complete his assignment and submit it the next day and since it is midnight he will not be able to give his laptop for repair. So he decides to make a character sequence of **all** the damaged keys in a sequence that he can copy and paste and make a word out of them.
Ajith needs to type a paragraph with all the characters in lower case. Help Ajith to find out the best permutation of the sequence of the characters (corresponding to the damaged keys) per word, that can be used while typing the paragraph, i.e. the sequence that will require least insertion and deletion while typing a word. Consider paste operation to be of one keystroke. Ignore the copy operation.

Recursively apply the same procedure for all the words in the paragraph. This way you will get the best combination that should be selected for that word. Finally, how many different words exist per character sequence combination. The combination that is the best for maximum words should be printed as output. If there are more than one candidates for best character sequence print the lexicographically smallest character sequence.

Refer the example section for better understanding.

Constraints

0 < Number of words in paragraph < 50

0 < Number of damaged keys <= 6

Input

First line contains the paragraph P that is to be written Second line contains the characters that represent the damaged keys

Output

One Line containing the best string which can be used to copy paste for the words.

Time Limit

1

Examples

Example 1

Input

supreme court is the highest judicial court

s u

Output

su

Explanation

There are two possible combinations of the damaged keys i.e. either *su* or *us*.
For word **supreme**, *su* is suitable as it requires only paste operation
For **court**, *us* is suitable as it requires less keystrokes for deletion
Similarly, for **is**, *su* is suitable and so on
Finally, we get *su* suitable for words supreme, is and highest and *us* suitable for words court (twice) and judicial. We get *su* and *us* suitable for 3 words each.
Since *su* is lexicographically smaller than us. So, the output will be *su*.

Example 2

Input

ginnestinggin gniinginging

n i g

Output

gin

Explanation

There are six possible combinations of the damaged keys i.e. {*nig, ngi, ing, ign, gni, gin*}.
For the first word, *gin* sequence requires least keystrokes for insertion and deletion. Similarly, for the second word *ing* sequence requires least keystrokes. Since both are eligible for best sequence, we will need to output the lexicographically smaller string. So, the output will be *gin*.

4.Engagement Ring

Problem Description

Sejal was on a month-long vacation to Europe and has a return trip to India from Lisbon, a city in south west part of Europe. However, a day before the return flight she realizes that she lost her engagement ring. After much contemplation, she decides to go to all the cities she visited to find her ring.

She maps all the cities she visited on a graph with Lisbon being at point (0,0). She then makes a route plan to visit all the cities and return to Lisbon by taking the shortest possible distance. She does not remember having her ring even in the first city she visited so there are high chances that she may have lost her ring in the initial part of her trip also. In case there are more than one routes which have the shortest possible distance, she picks that route in which the first city she visited, comes first. For example, if she visited cities 2,5,7,1,8,3 in that order and routes 0,1,8,3,2,5,7,0 and 0,8,3,1,5,2,7,0 (0 being Lisbon) have the same shortest possible distance then she will choose route 0,1,8,3,2,5,7,0 because she visited city 1 before city 8.

Her travel guide, Harry also offers to help Sejal. Sejal asks him to travel separately on the same route, but in reverse direction such that each city is visited only once. They plan to travel 20 Kms in each city on taxi to search the ring. Inter-city travel is done on trains only.

A secret service officer knows the coordinates of the city that Sejal visited during her the trip in that order. He also knows the city in which the ring is lost but will inform Sejal or Harry only when one of the two is in that city.

He knows the path that Sejal has drawn to visit all the cities and return to Lisbon. With Sejal and Harry following that path and either one of them reaching the city where the ring is lost, the secret service officer will inform the person in that city, that the ring will be 10 km away from their current location. They will travel back 10 km in the same city, to catch the train back to Lisbon. Calculate the total distance traveled by Sejal in her search and her return to Lisbon from that city.

If the ring is found by Sejal, she goes back to Lisbon from that city. If the ring is found by Harry, he informs Sejal on call at that point. If Sejal is in a city (searching in taxi) she returns to Lisbon via train from that city (without searching any further in that city). If Sejal is on train, she will need to complete the journey and then return from that other city. If the call comes at the exact point she is taking a train, she can return from that city itself.

Each unit in the graph is equal to 1 Km. Assume the speed of all trains and taxis is same. Do not consider the decimal values while calculating the distance between two cities, ie. distance will be the floor of the calculated distance.

Constraints

Floor of the value is to be used while calculating distance between cities

Each city is connected to every other city via trains

Total number of cities <10

Input

First Line will provide the coordinates (x|y) of cities separated by semicolon (;)

Second Line will provide the number of city where the ring is found

Output

One integer representing the number total distance traveled by Sejal

Time Limit

1

Examples

Example 1

Input

0|90;90|90;90|0

2

Output

347

Explanation

Since routes 1,2,3 and 3,2,1 will give the shortest distance, she will select route 1,2,3 as she visited city 1 before city 3. However, Harry will follow the opposite path - 3,2,1. They both will be in city 2 when they will find it. Total distance Sejal covers will be Lisbon to city 1 + 20Kms, City 1 to City 2 + 10 km +10 km and then City 2 to Lisbon. i.e. 90 + 20 + 90 + 20 +127 = 347 km

Example 2

Input

10|70;30|30;80|20;120|75;90|120

3

Output

334

Explanation

Sejal will choose the route 1, 5, 4, 3, 2 with minimum distance of 378 km. However, Harry will follow the opposite path - and will find the ring in City 3. The moment Harry finds the ring, Sejal will be travelling from City 1 to City 5. So she will complete the journey till City 5 and return from there to Lisbon. So the total distance covered by Sejal will be 70+20+94+150 = 334 km

5.Lift

Problem Description

In a building there are some lifts. Optimize the allocation of lifts.

Say there are N requests and M lifts. Minimize the maximum request waiting time.

Rules of lift allocation

1) One needs to assign indexes to lifts. i.e. decide lift #1, lift #2, lift #3, etc apriori.

2) Since all N requests are known apriori decide the initial (at time t = 0) location of lift such that it will help in minimizing waiting time.

3) Even if all the N requests time are known apriori, other than initial time, i.e. t = 0, the waiting lifts cannot be moved towards target floor until the request is actually issued.

4) After a request is issued for optimality calculation, even a lift is in transit it can be considered for serving that request.

5) If at any moment, more than one lift can serve the request with same waiting time, give preference to the lift with lower index. i.e. If Lift #2 and Lift #4 can serve a particular request in 2 seconds, then prefer Lift #2 over Lift #4.

6) Neglect the time required to get in and get out of a lift.

7) Once a lift starts serving a request it would not stop in between. If would finally stop at destination of that request.

8) The speed of lift is 1 floor/second.

Constraints

0 <= T <= 86400.

0 <= S, D <= 100.

1 <= N, M <= 1000.

Input

First line contains 2 integers, N and M, representing the number of requests and number of lifts in the building respectively.

Next N lines contain 3 integers, T, S, and D, representing the timestamp of request, source floor, destination floor respectively.

Output

Print single integer representing the waiting time

Time Limit

1

Examples

Example 1

Input

3 2

0 2 3

4 2 5

6 7 3

Output

3

Explanation

There are 3 requests and 2 lifts.

Initially at time t = 0, both the lifts, lift #1 and lift #2 will be at floor 2 (Initial allocation).

Request #1 and Request #2 can be responded instantly, i.e. waiting time = 0.

Lift #1 will get unallocated at floor 3 at time t = 1.

Lift #1 can move only after the next request is actually received at time t = 4, if required. Relate this with Rule #3 mentioned in problem description.

Lift #2 will get unallocated at floor 5 at time t = (4 + 3) = 7.

Now, if Lift #1 is used to respond request #3, waiting time would be 4 seconds since Lift#1 is at floor #3 and will need 4 seconds to reach floor #7.

In this case, waiting time of all requests - {0,0,4} and the maximum waiting time is 4.

Instead, if Lift #2 is used to respond request #3, waiting time would be calculated as follows

Lift #2 will get unallocated at time t = 7, so we will have to wait 7-6 = 1 seconds for lift #2 to complete it's previous request.

Time needed for Lift #2 to travel from floor #5 to floor #7 = 7-5 = 2 seconds. Therefore, total waiting time = 1 + 2 = 3 seconds.

In this case, waiting time of all requests - {0, 0, 3} and the maximum waiting time is 3.

As we have to minimize the maximum waiting time, since 2nd option is yielding lesser waiting time than 1st option, 2nd option is the answer.

Therefore, the output is 3.

Example 2

Input

5 3

0 2 3

4 2 5

6 7 3

3 5 6

2 5 7

Output

1

Explanation

There are 5 requests and 3 lifts.

Initially, at t = 0, lift #1 will be at floor #2 whereas lift #2 and #3 will be at floor#5 (Initial allocation).

Request #1, #4, #5 can be responded instantly, i.e. waiting time = 0.

Lift #1 will get unallocated at floor 3 at time t = 1.

Lift #2 will get unallocated at floor 7 at time t = 2 + 2 = 4.

Lift #3 will get unallocated at floor 6 at time t = 3 + 1 = 4.

Request #2 can be served by lift #1. Waiting time would be 2 - 1 = 1.

Now, lift #1 will get unallocated at floor #5 at time t = 4 + 1 + 3 = 8.

Request #3 can be served by lift #3 as it will be floor #6 at t = 4. Waiting time = 0.

Waiting time of all requests - {0, 1, 0, 0, 0}.

Therefore, the output is 1.

6.Hedger

Problem Description

A hedger is an investor who takes steps to reduce the risks of investment by doing appropriate research and analysis of stocks. Assume that you are a hedger.

Now you have been given some parameters based on which you have to analyze and pick the correct stocks. You have been assigned a fund that you have to manage in such a way that it should give maximum returns.

According to your research you have a list of stocks for which you know the corresponding profit percentages that you can earn within a certain horizon. Being a hedger you don't want to put all your eggs in one basket. That's why you have decided the upper limit in terms of quantity that you will buy.

Given number of stocks, the upper limit on quantity, the amount to be invested, the list of stock prices and list of profit percentages, calculate the maximum profit you can make.

Note: All computation should be up to two digits after the decimal point.

Constraints

0 <= A <= 10^6

1 <= K <= 100

1 <= N <= 10^4

Input

First line contains three space separated integers N K A where N is number of stocks in Market, K is maximum quantity of any particular stock you can buy and A is capital amount you have.

Second line contains N space separated decimals denoting the prices of stocks.

Third line contains N space separated decimals denoting the profit percentages corresponding to the index of stock prices.

Output

Print the maximum profit that can be earned from the given amount, rounded to the nearest integer.

Time Limit

1

Examples

Example 1

Input

4 2 100

20 10 30 40

5 10 30 20

Output

26

Explanation

Here, we can select only two stocks of any stocks that we choose to buy. We choose to buy stocks which are priced at 30 and 40 respectively. Before we exhaust our capital maximum profit that can be earned

Profit=2*((30*30)/100) +1*((40*20)/100)=26

Example 2

Input

5 3 200

90 25.5 15.5 30.8 18.8

5 10 20 5.5 2.5

Output

20

Explanation

Here, we can select only three stocks of any stocks that we choose to buy. We choose to buy stocks which are priced at 25.5, 15.5 and 30.8 respectively. Before we exhaust our capital maximum profit that can be earned

Profit=3*((25.5*10)/100) +3*((15.5*20)/100)+2*((30.8*5.5)/100)=20.34 Hence, output is 20.