**TCS CODE-VITA PROBLEMS**

# Program-1

## **WebPages Marks: 30**

Problem Description

Consider a set of web pages, numbered from 1 to N. Each web page has links to one or more web pages. Clicking on a link in a page, takes one to the other web page. You are provided numbers of two web pages viz, starting web page and end web page. Your task is to find the minimum number of clicks required to reach the end page from the start page. If end page cannot be reached from start page, print -1 as the output. For better understanding refer *Examples* section.

Constraints

0 < N <= 100

0 < L < 10

Input

First line contains an integer N denoting number of web pages.

Next N lines contain L space separated integers depicting linked webpage number(s) from that webpage

Output

Print the minimum number of clicks required to open the end page from start page. If not possible, print -1 as output.

Time Limit (secs)

1

Examples

Input

5

2 4

1

1 5

2 3

5

2 3

Output

3

Explanation:

First line conveys that there is total 5 pages.

Second line conveys that there are links from page 1 to pages 2 and 4.

Third line conveys that there is a link from page 2 to page 1.

Fourth line conveys that there are links from page 3 to pages 1 and 5.

Fifth line conveys that there are links from page 4 to pages 2 and 3.

Sixth line conveys that there is a links from page 5 to page 5 itself.

Seventh line conveys that starting page is 2 and ending page is 3

From page 2, we can open only page 1. From page 1, we can open page 4. From page 4, we can open page 3. So, minimum 3 clicks are required, and this is the output.

Example 2

Input

3

2

1

1

2 3

Output

-1

Explanation:

First line conveys that there is total 3 pages.

Second line conveys that there are links from page 1 to page 2.

Third line conveys that there is a link from page 2 to page 1.

Fourth line conveys that there are links from page 3 to page 1.

Since there is no way to reach from page 2 to page 3, print -1 as output.

# Program-2

## **Mario Marks: 30**

### **Problem Description**

Imagine you are a video games developer. You are developing a game which requires the player to collect coins and cross hurdles. Let's call the character in your video game to be Mario. As Mario moves to collect coins and cross hurdles, the game keeps a count of relevant metrics. Write code to implement this flow.

Mario will run from left to right and jump from ground in the air to collect coins or cross hurdles. The Game Screen will be provided as input in form of a matrix comprising of three characters viz {0, C and H}, where

0 - denotes empty space

C - denotes coins to be collected

H - denotes hurdles to be crossed

All coins are of the same type, whereas there are two types of hurdles - simple hurdle and ring hurdle. Simple hurdle is referred to as Hurdle hereafter.

A Hurdle always begins from the ground and a series of letter H stacked vertically make up the height of the hurdle.

A Ring Hurdle on the other hand, has a hole in it i.e., between H characters there will be exactly one hole denoted by 0 character. This hole is big enough for Mario to jump through it to cross that hurdle.

Now, let us understand how this information is provided in the input

* The screen will be depicted in the input as a M \* N matrix. The index of row and columns of this matrix begin from zero.
* The left bottom cell of this matrix is (0, 0). As we move right and up, the row and column indices increase
* Row zero is considered as Ground and anything above row zero is considered as Air
* Coins will always be in air, whereas hurdles will always manifest from the ground
* Hurdle will never be so tall that Mario cannot cross it
* Once Mario crosses all the columns, the game is over
* To collect coins Mario will jump vertically in the column where the coin is. Mario always jumps to the highest point where a coin is, on the screen. On his way down from that point, he grabs all coins lower in height in that column. Thus, one jump in one column is enough to fetch all coins in that column
* Jumping consumes energy. Jumping one row consumes 2 calories. Similarly, if Mario jumps R rows in a column, his calorific expenditure is 2 \* R
* Mario never jumps unless he must collect coins or cross a hurdle
* When crossing a ring hurdle, the calories consumed in clearing it is 2 \* height of the hole in the ring hurdle. Refer Examples section for better clarity
* Walking i.e., moving from one column to another consumes no energy

Your task is to keep a track of how many coins Mario collects and how many calories are expended in collecting them.

Consider a screen(grid) of size 5 \* 10:

0000000000

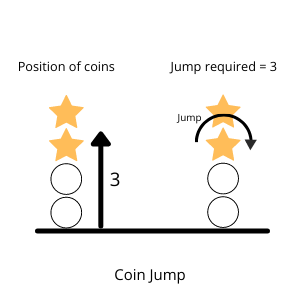
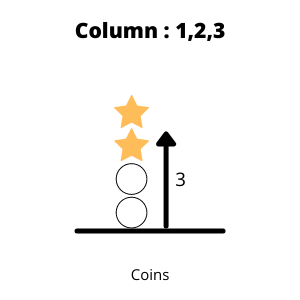
0CCC00000H

0CCC0H0000

00000H0H0H

00000H0H0H

We can see that we have coins on the screens at column 1, 2 and 3.

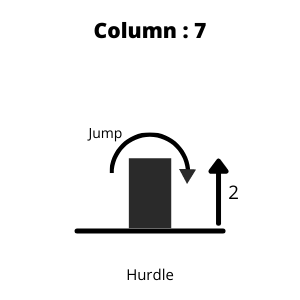
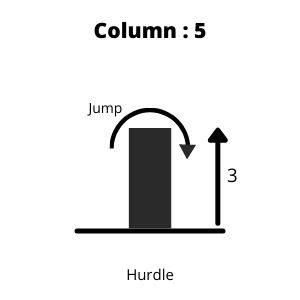


The above two images describe collection of coins and energy spent in collecting them.

Column 1 has two coins at a height of 2 and 3 respectively. So, Mario will jump 3 units high and collect the highest coin. On his way down he will collect the coin at height 2. Total calories expended in collecting both coins in Column 1 is 3 \* 2 = 6 calories.

Columns 2 and 3 are identical to Column 1. Hence Mario will have collected 6 coins and spent 18 calories in traversing the grid up to column 3.

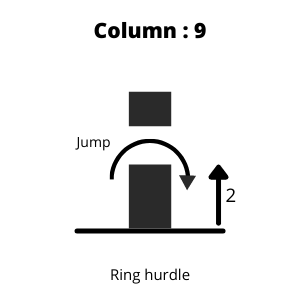
Column 4 is empty. So, no energy is expended traversing it. Next, there are hurdles at Column 5 and 7.



For clearing these hurdles, he must jump over two hurdles, and by doing this he will consume 3 \* 2 + 2 \* 2 calories.

Total calories burned till now: 18 + 10 = 28

There is also a ring hurdle at column 9.



For clearing these hurdles, he must jump over two hurdles, and by doing this he will consume 2 \* 2 calories.

Total calories burned till now: 28 + 4 = 32

Total Coins grabbed: 6

Total Calories burned: 32

Note: Assume Mario can jump any hurdle of any height and collect coins at any height

### **Constraints**

0 < M <= 10

10 <= N < 100

### **Input**

First line contains two space separated integers M and N which denote the size of grid (screen)

Next M lines will contain string of size N characters. The string will be comprised of {0, C and H} characters

### **Output**

Print two space delimited integers. First integer denotes the number of coins grabbed and second integer denotes the calories expended in crossing the screen

### **Time Limit (secs)**

1

### **Examples**

Example 1

Input

5 10

0000000000

0CCC00000H

0CCC0H0000

00000H0H0H

00000H0H0H

Output

6 32

Explanation:

Explained in problem description section.

Example 2

Input

5 10

0000000000

000000000H

00H00H0000

00H00H0H0H

00H00H0H0H

Output

0 20

Explanation:

As we can see there are two hurdles of height 3 at column 2 and column 5 (3 \* 2 \* 2), one hurdle of height of 2 at column 7(2 \* 2) and a ring hurdle of height 2(2 \* 2) at column 9.

Thus, total coins grabbed in 0 and total calories expended is 20.

# Problem-3

## **Primevilla Marks: 30**

### **Problem Description**

It rains only on certain days in Primevilla. It is a rainy day when the number of days since a certain date is prime as well as the month is prime (i.e., month is one of Feb, Mar, May, Jul, Nov). Primevilla follows same calendar that we use.

Given the date d, identify if it would ever rain on a given weekday w within the next given n days. Also calculate the number of days r (where r > 0) after which it would rain within the next n days.

### **Constraints**

2 <= n <= 10 ^ 10

### **Input**

Input consists of three space delimited parts viz. <Date, DOW, n> where

* Date is of the format yyyymmdd
* DOW is Date of the Week in Ddd format
* n is a natural number (where n>=2)

### **Output**

Output would be one of the three formats

* If it would rain on Ddd, r days after d within the next n days, print 'Yes r'
* Else if p is the least prime > n such that it would rain on Ddd after p days, print 'No p'
* Else print 'No 0', if it would never rain on Ddd.

### **Time Limit (secs)**

1

### **Examples**

Example 1

Input

20211201 Sun 100

Output

Yes 67

Explanation

Starting from 20211201 we start checking for prime number days whether it would rain. The process will be as depicted below

20211201+2 is 20211203 Fri, month 12 is not prime ---> It would not rain

20211201+3 20211204 Sat, month 12 is not prime ---> It would not rain

..

20211201+31 is 20220101 Sat, month 1 is not prime ---> It would not rain

..

20211201+67 is 20220206 Sun, month 2 is prime ---> It would rain

Hence it could rain on a Sunday 67 days after the given date and within the given 100 days. Hence, output is "Yes 67"

Example 2

Input

20211201 Wed 100

Output

No 0

Explanation

The given date 20211201 is itself a Wednesday.

20211201+7 is 20211208 Wed, month 12 is not prime ---> It would not rain

Also, any future Wednesday would be a multiple of 7 and hence non-prime days later than 20211201. So, it would never rain on future Wednesdays. Hence, output is "No 0"

Example 3

Input

19470815 Sat 150

Output

No 197

Explanation

If similar computation as depicted in prior examples is carried out one can figure out that it will rain on Sat, 28 Feb 1948 which is 197 days from 15 Aug 1947. However, since 197 is greater than 150 output is "No 197".

# Problem-4

## **HelpMLA Marks: 100**

### **Problem Description**

Imagine you are an MLA of a district and there are N number of villages in your constituency.

Your job is to vaccinate all the people in your constituency in minimum amount of time. There are two centres where vaccination is going on. First centre takes m1 minutes as average time for vaccinating one person and second centre takes m2 minutes as average time.

Population of every village is known to you prior to the vaccination drive. Schedule all the villagers to any centre such that overall time for vaccinating all the people of all the villages will be minimum.

Assume that there is no wait time in between vaccinating two people. Also, people belonging to the same village will need to be vaccinated in the same centre.

For example:

First centre takes 2 min as average time

Second centre takes 4 min as average time

Population data of 3 villages is known: 10 30 20

Number of people in 3 villages is given:

v1 = 10, v2 = 30, v3 = 20

Consider if schedule is drawn by distributing equal number of people to both centres, then

First centre: 10 20 total time = (10 + 20) \* 2 = 60 min

Second centre: 30 total time = (30) \* 4 = 120 min

Hence, minimum time required to vaccinate all the people will be = 120 min. i.e., Maximum of time taken in first centre or second centre.

But if it is scheduled like this:

### **Constraints**

0 < m1, m2 <= 20

0 < N < 10 ^ 3

0 < Population of village <= 100

### **Input**

First line contains an integer m1 which is average time in minutes taken for vaccination by the first centre

Second line contains an integer m2 which is average time in minutes taken for vaccination by the second centre

Third line contains an integer N which is number of villages in the constituency

Fourth line contains N space delimited integers denoting the population of villages

### **Output**

Single integer value denoting the maximum time taken in minutes to vaccinate all villagers from all villages in your constituency

### **Time Limit (secs)**

1

### **Examples**

Example 1

Input

2

3

5

10 50 20 30 40

Output

180

Explanation:

Given the data of centre1 and centre2:

First centre takes 2 min as average time. Second centre takes 3 min as average time. Your constituency has 5 villages.

Number of people in each of the 5 villages is given: 50 10 20 30 40

v1 = 50, v2 = 10, v3 = 20, v4 = 30, v5 = 40

If schedule looks like this:

First centre: 10 50 total time = (10 + 50) \* 2 = 120 min

Second centre: 30 40 20 total time = (20 + 40 + 20) \* 3 = 240 min

Minimum time required to vaccinate all the people will be = 240 min

But if the schedule is drawn like this:

First centre: 10 30 50 total time = (10 + 30 + 50) \* 2 = 180 min

Second centre: 40 20 total time = (40 + 20) \* 3 = 180 min

Minimum time required to vaccinate all the people will be = 180 min

Example 2

Input

1

2

3

100 90 70

Output

180

Explanation:

Given the data of centre1 and centre2:

First centre takes 1 min as average time. Second centre takes 2 min as average time. There are 3 villages in your constituency.

Number of people in each of the 3 village is given: 100 90 70

v1 = 100, v2 = 90, v3 = 70

If schedule looks like this:

First centre: 100 90 total time = (100 + 90) \* 1 = 190 min

Second centre: 70 total time = (70) \* 2 = 140 min

Minimum time required to vaccinate all the people will be = 190 min

But if schedule is drawn like this:

First centre: 100 70 total time = (100 + 70) \* 1 = 170 min

Second centre: 90 total time = (90) \* 2 = 180 min

Minimum time required to vaccinate all the people will be = 180 min. Hence the output is 180.

# Problem-5

## **SnakeAndPriest Marks: 100**

### **Problem Description**

Given a Matrix of N\*N.  
There are M snakes present in the matrix. They will move in a specific range the movement is given below.  
Snake movement-  
  
1)If it is vertically oriented in the matrix. It will move up or down. Based on the start and end block provided in the input.  
2)If it is horizontally oriented in the matrix. It will move left or right. Based on the start and end block provided in the input.  
3)Snake will move one block ahead at 1 unit time.  
4)If it reaches the boundary(ith row or jth column) of the matrix it will come from another end of the matrix in the same ith row or jth column.

5) start\_block is the head and end\_block is the tail.

Note: start\_block and end\_block represent the snake occupying these cells and the cells between them.

For example, if start\_block is 1,5 and end\_block is 1,8 then snake occupies (1, 5), (1, 6), (1, 7), (1, 8)  
  
For Example-  
Snake Initial Position

|  |  |  |
| --- | --- | --- |
| Snake\_name | Start\_block | End\_block |
| Snake1 | 1,8 | 1,5 |

In this case, snake1 will move towards west all the time because the start block is towards the east and the end block is towards the west, when it reaches the end of the matrix it will again start from the start of row 1 towards east.  
  
To achieve Nirvana the priest has to cross the matrix. The priest moves one step at a time. He can start from any side of the matrix and reach the other side. If he starts from the north, he needs to go to the south but cannot go in east and west direction, if he starts from the east, he needs to go to the west but not in south and south directions and vice versa.  
Check whether he will be able to get Nirvana, if not print which snake killed him at which location.

**Priest Input format**-

The input format of priest is as follows:

DirectionNumber where Direction represents directions (E for East, w for west, N for North, S for South) and Number represents column or row.

For north and south directions, number always represents the column. The row of north and south are first and last row respectively.

For east and west directions, number always represents the row. The column of east and west are last and first column respectively.

For example  
Ex- W5 -> This means it will start from the west and move towards the east. Initially, it will be the 1st column and 5th row.  
N2 -> This means it will start from the north and move towards the south. Initially, it will be the 1st row and 2nd column.  
Note- The snakes can overlap each other.

### **Constraints**

10<=N<50  
1<=M<20

### **Input**

First-line contains an integer N denoting the number of rows and columns in a matrix.  
Second-line contains an integer M denoting the number of snakes present.  
Next M lines contain space-separated data about each snake's initial position. In the below-given Format  
Snake\_name start\_block end\_block  
The next line contains string denoting the block and direction from which the priest will start to move.

### **Output**

Print "NIRVANA" in case of priest reaches the other side of the matrix.  
OR  
Print the which snake killed the priest and at which position.

### **Time Limit (secs)**

1

### **Examples**

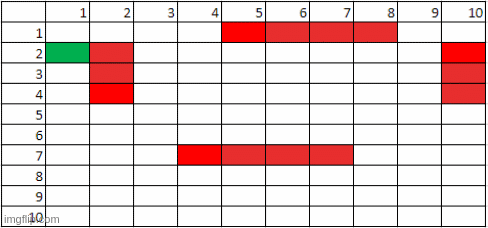
Input

10  
4  
Snake1 1,5 1,8  
Snake2 7,4 7,7  
Snake3 2,10 4,10  
Snake4 4,2 2,2  
W2

Output

NIRVANA

Explanation-



Example 2

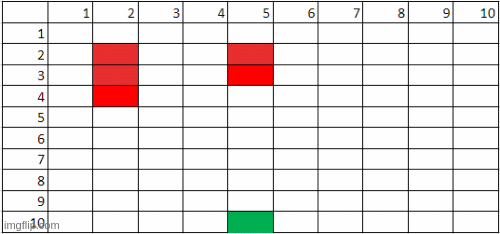
Input

10  
2  
Snake1 3,5 2,5  
Snake2 4,2 2,2  
S5

Output

Snake1 6,5

Explanation-



# Problem-6

## **IPLMINUSTWO Marks: 200**

### **Problem Description**

The Indian Premier League (IPL) is a professional Twenty20 [cricket](https://en.wikipedia.org/wiki/Laws_of_Cricket) league, contested by eight teams based out of eight Indian cities. IPL is going to start, and, in this season, you have been given the responsibility to maintain the score board of the league. You have been provided the result of each match. Total 8 teams compete (MI, CSK, RCB, RR, SRH, DC, PKBS, KKR) for winning the title. **Each team will play two matches with every other team before playoffs.** So, the total number of matches before playoffs will be 56 league matches. However, you will be provided details of only 54 matches.

The format in which the scores will be given is - the runs scored by team batting first followed by the fall of wickets followed by the number of balls played. The same data will be provided for team batting second.

For example-

Abstract Format - FirstTeamName Runs/Wickets BallsPlayed SecondTeamName Runs/Wickets BallsPlayed

Actual Record - MI 191/10 110 CSK 190/4 120

Here MI scored 191 runs in 110 balls at the loss of 10 wickets and CSK scored 190 runs at the loss of 4 wickets in 120 balls.

The points table is to be constructed based on the given inputs. The points table contains <TeamName, TotalPointsScored by team, and NRR (Net Run Rate)>. The order in which teams will be present is based on the ranking. Ranking of teams is first based on number of points and then on NRR, in case the points are tied.

The points table formation rules are given below-

* The team winning the match will get 2 points.
* NET RUN RATE is calculated as follows:
  + Case 1: When both teams make same score. Each team is rewarded with 1 point each and run rate is zero for both the teams.
  + Case 2: Team batting first wins the match In this scenario, the margin of runs by which team wins are multiplied by 2 and we get run rate of the match.
    - Run rate (RR) = (Runs Difference) \* 2
    - The winning teams gets positive RR and the losing team gets negative RR.
    - For example - team1 = 200 in 120 balls (20.0 overs) and team2 scores 160 in 96 balls (16.0 overs). Here, team2 scored 160 runs in 96 balls. So, team1 RR = (200-159) \* 2 = 82 and team2 RR is -82.
  + Case 3: Team Batting second wins the match.
    - In this scenario total runs which would have been scored by the winning team if it had batted 20 overs (120 balls) are calculated and the difference with the first team is taken.
    - Run Rate= (Projected score of winning team - score of losing team) \* 2
    - For example - team1 = 159 and team2 scores 160 in 96 balls (16.0 overs). Here, team2 scored 160 runs in 96 balls so runs that can be scored in 120 balls is 200. So, the projected score for team2 is 200. So, team2 RR = (200-159) \* 2 = 82 and team1 RR is -82.

Net Run rate = (sum of Run Rate of individual matches) / (total number of matches).

Information about 54 matches is provided. Ranking of the teams after all 56 matches is provided. Basis these two pieces of information, determine the teams that played the other two matches along with the outcome of those matches i.e., who won and who lost.

Assume that the result of remaining two matches will be win or loss, no draw / tie. Further assume that even in 54 match records there are no tied matches.

Note - There are no test cases where the teams have same points and same NRR by the end of league matches.

For reference about Cricket and IPL

Cricket T20 - <https://en.wikipedia.org/wiki/Twenty20>

IPL- <https://en.wikipedia.org/wiki/Indian_Premier_League>

### **Constraints**

N = 54 where N is the outcome of 54 league matches

### **Input**

First 54 lines contains a string which represents the result for every match in the format described in problem description section

Next 8 lines contain a string denoting ranking of teams in ascending order of ranks (i.e., first sorted on points and then by net run rate (NRR)).

### **Output**

Print the possible outcomes of the remaining two matches in lexicographical order of cases.

WIN:TEAM1|LOSE:TEAM2||WIN:TEAM3:LOSE:TEAM4

WIN:TEAM2|LOSE:TEAM1||WIN:TEAM3:LOSE:TEAM4

.. all possible cases.

Where || is the match separator and | is the team separator

And winner is always placed on the left hand side of the team separator and loser is always placed on the right hand side.

### **Time Limit (secs)**

1

### **Examples**

Example 1

Input

MI 111/4 120 PBKS 115/8 84

RR 211/9 120 SRH 150/4 120

KKR 179/3 120 RCB 129/7 120

MI 217/7 120 PBKS 174/10 120

SRH 177/2 120 DC 149/4 120

RR 190/4 120 PBKS 137/10 120

KKR 202/9 120 RCB 202/8 120

MI 174/8 120 RCB 164/9 120

KKR 162/9 120 DC 114/4 120

CSK 197/3 120 RCB 120/1 120

MI 136/2 120 KKR 116/6 120

RR 140/4 120 RCB 144/4 115

CSK 160/6 120 DC 164/6 79

CSK 124/2 120 SRH 128/7 76

MI 216/3 120 CSK 196/5 120

KKR 118/8 120 PBKS 122/10 113

MI 205/2 120 DC 184/8 120

CSK 150/6 120 RR 154/10 77

PBKS 110/10 114 RCB 114/9 105

MI 178/1 120 RR 146/2 120

CSK 198/9 120 KKR 148/9 120

RR 120/6 120 PBKS 124/9 71

SRH 144/2 120 RCB 148/9 113

CSK 127/4 120 PBKS 119/10 120

DC 186/6 120 RCB 190/7 91

CSK 129/9 120 DC 133/8 91

MI 148/2 120 RR 152/9 119

MI 213/7 120 SRH 142/4 120

MI 208/6 120 CSK 198/4 120

RR 171/4 120 DC 123/6 120

SRH 169/6 120 DC 147/6 120

KKR 116/2 120 RR 120/7 91

CSK 168/10 117 KKR 172/5 67

SRH 179/7 120 RCB 183/6 92

CSK 210/6 120 PBKS 186/4 120

KKR 136/8 120 SRH 110/5 120

MI 110/5 120 RCB 114/5 103

CSK 193/6 120 RR 147/8 120

SRH 178/6 120 PBKS 162/9 120

KKR 193/10 103 DC 156/9 120

MI 210/3 120 KKR 188/6 120

KKR 117/2 120 PBKS 111/4 120

MI 218/8 120 SRH 118/1 120

RR 163/3 120 RCB 167/4 100

DC 130/4 120 RCB 134/3 109

CSK 189/1 120 RCB 173/4 120

SRH 151/6 120 PBKS 125/6 120

RR 112/6 120 SRH 116/2 91

MI 162/2 120 DC 166/9 99

PBKS 214/10 88 DC 185/6 120

PBKS 116/7 120 RCB 120/6 61

PBKS 163/4 120 DC 167/2 92

KKR 114/3 120 RR 118/10 109

CSK 131/9 120 SRH 117/8 120

MI

RCB

RR

SRH

CSK

KKR

DC

PBKS

Output

WIN:DC|LOSE:RR||WIN:SRH|LOSE:KKR

WIN:RR|LOSE:DC||WIN:SRH|LOSE:KKR

Explanation-

From the given 54 matches and the rankings as input, four teams have one match remaining. Specifically, the match DC vs RR and SRH vs KKR.

Case1- DC wins against RR and SRH wins against KKR. In that case the rankings will look as follows MI, RCB, RR, SRH, CSK, KKR, DC, PBKS.

Case2- RR wins against DC and SRH wins against KKR. In that case the rankings will look as follows MI, RCB, RR, SRH, CSK, KKR, DC, PBKS.

Then print the cases lexicographically in provided format.