**Code vita**

## 1. Bride Hunting

**Problem Description**

Sam is an eligible bachelor. He decides to settle down in life and start a family. He goes bride hunting.

He wants to marry a girl who has at least one of the 8 qualities mentioned below:-

1) The girl should be rich.

2) The girl should be an Engineer/Doctor.

3) The girl should be beautiful.

4) The girl should be of height 5.3".

5) The girl should be working in an MNC.

6) The girl should be an extrovert.

7) The girl should not have spectacles.

8) The girl should be kind and honest.

He is in search of a bride who has some or all of the 8 qualities mentioned above. On bride hunting, he may find more than one contenders to be his wife.

In that case, he wants to choose a girl whose house is closest to his house. Find a bride for Sam who has maximum qualities. If in case, there are more than one contenders who are at equal distance from Sam’'s house; then

print "“Polygamy not allowed”".

In case there is no suitable girl who fits the criteria then print “**"No suitable girl found"**”

Given a Matrix N\*M, Sam's house is at (1, 1). It is denoted by 1. In the same matrix, the location of a marriageable Girl is also denoted by 1. Hence 1 at location (1, 1) should not be considered as the location of a marriageable Girl’s location.

The qualities of that girl, as per Sam’'s criteria, have to be decoded from the number of non-zero neighbors (max 8-way) she has. Similar to the condition above, 1 at location (1, 1) should not be considered as the quality of a Girl. See Example section to get a better understanding.

Find Sam, a suitable Bride and print the row and column of the bride, and find out the number of qualities that the Bride possesses.

NOTE: - Distance is calculated in number of hops in any direction i.e. (Left, Right, Up, Down and Diagonal)

**Constraints**

2 <= N,M <= 10^2

**Input Format**

First Line contains the row (N) and column (M) of the houses.

Next N lines contain the data about girls and their qualities.

**Output**

It will contain the row and column of the bride, and the number of qualities that Bride possess separated by a colon (i.e. :).

**Explanation**

Example 1

**Input**:

2 9

1 0 1 1 0 1 1 1 1

0 0 0 1 0 1 0 0 1

**Output**:

1:7:3

**Explanation**:

The girl and qualities are present at (1,3),(1,4),(1,6),(1,7),(1,8),(1,9),(2,4),(2,6),(2,9).

The girl present at (1,3) has 2 qualities (i.e. (1,4)and (2,4)).

The girl present at (1,4) has 2 qualities.

The Bride present at (1,6) has 2 qualities.

The Bride present at (1,7) has 3 qualities.

The Bride present at (1,8) has 3 qualities.

The Bride present at (1,9) has 2 qualities.

The Bride present at (2,4) has 2 qualities.

The Bride present at (2,6) has 2 qualities.

The Bride present at (2,9) has 2 qualities.

As we see, there are two contenders who have maximum qualities, one is at (1,7) and another at (1,8).

The girl who is closest to Sam's house is at (1,7). Hence, she is the bride.

Hence, the output will be 1:7:3.

Example 2

**Input**:

6 6

1 0 0 0 0 0

0 0 0 0 0 0

0 0 1 1 1 0

0 0 1 1 1 0

0 0 1 1 1 0

0 0 0 0 0 0

**Output**:

4:4:8

**Explanation**:

The bride and qualities are present at (3,3),(3,4),(3,5),(4,3),(4,4),(4,5),(5,3),(5,4),(5,5)

The Bride present at (3,3) has 3 qualities (i.e. (3,4),(4,3) and (4,4)).

The Bride present at (3,4) has 5 qualities.

The Bride present at (3,5) has 3 qualities.

The Bride present at (4,3) has 5 qualities.

The Bride present at (4,4) has 8 qualities.

The Bride present at (4,5) has 5 qualities.

The Bride present at (5,3) has 3 qualities.

The Bride present at (5,4) has 5 qualities.

The Bride present at (5,5) has 3 qualities.

As we see, the girl present in (4,4) has maximum number of Qualities. Hence, she is the bride.

Hence, the output will be 4:4:8.

2. Skateboard

Problem Description

The amusement park at Patagonia has introduced a new skateboard competition. The skating surface is a grid of N x N squares. Most squares are so constructed with slopes that it is possible to direct the skateboard in any of up to three directions of the possible four (North ,East, South or West, represented by the letters N, E, S and W respectively). Some squares however have a deep drop from the adjacent square from which it is impossible to go to any adjacent square. These are represented by D (for Drop) in that square. The objective is to maneuver the skateboard to reach the South East corner of the grid, marked F.

Each contestant is given a map of the grid, which shows where the Drop squares are (marked D), where the Final destination is (marked F), and, for each other square, the directions it is possible to maneuver the skateboard in that square.

The contestant draws lots to determine which of the squares on the boundaries of the grid on the North or the West of the grid (the top or the left in the diagram) he or she should start in. Then, using a map of the grid, he or she needs to try to reach the South East corner destination by maneuvering the skateboard.

.

In some cases, it is impossible to reach the destination. For example, in the diagram above, if one starts at the North East corner (top right in the diagram), the only way is to go is South, until the Drop square is reached (three squares South), and the contestant is stuck there.

A contestant asks you to figure out the number of squares at the North or West boundary (top or left boundary in the map) from which it is feasible to reach the destination.

Constraints

5<=N<=50

Input Format

The first line of the input is a positive integer N, which is the number of squares in each side of the grid.

The next N lines have a N strings of characters representing the contents of the map for that corresponding row. Each string may be F, representing the Final destination, D, representing a drop square, or a set of up to three of the possible four directions (N,E,S,W) in some random order. These represent the directions in which the contestant can maneuver the skateboard when in that square.

Output

The output is one line with the number of North or West border squares from which there is a safe way to maneuver the skateboard to the final destination.

Explanation

Example 1

Input

6

ES,ES,SE,ES,ES,S

SE,ES,SE,ES,ES,S

ES,ES,SE,ES,SE,S

ES,SE,ES,SE,E,D

SE,ES,D,WSE,NES,NS

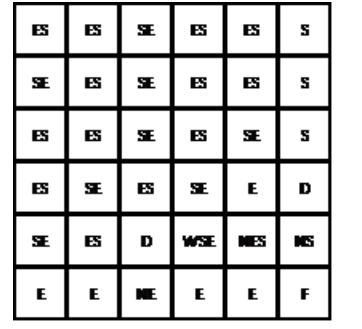
E,E,NE,E,E,F

Output

9

Explanation

N =6, and the size of the grid is 6x6. The map of the grid is as below.



There are many ways to maneuver the skateboard. For example, if the contestant starts from the North West corner (top left in the map) and goes East all the way until he reaches the top right corner in the map, and then goes South, he will reach a Drop square. But if he goes South all the way from the same square until he reaches the bottom left square on the map, and keeps going East from there, the Final destination will be reached. Hence the top left square (1,1) is safe.

Similarly, from the square (1,5), all the paths lead to a drop square., The other 9 North and West border squares have ways skateboard to get to the final destination. The output is 9

Example 2

Input

5

ES,SE,ES,SE,S

S,EWS,SE,E,S

E,D,SEW,NSE,S

NE,N,SEW,D,W

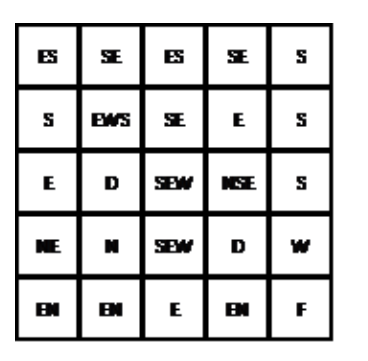
EN,EN,E,EN,F

Output

4

Explanation

N=5, and the grid is 5 x 5 squares. The map of the grid looks like this.



It can be seen that from squares (1,4) and (1,5), there is no way to maneuver the skateboard to the Final destination, and hence are not safe starting points.. Similarly, squares (2,1),(3,1), and (4,1) are not safe starting points. The only safe starting points on the North and West borders are (1,1),(1,2),(1,3),(5,1). Hence the output is 4

ESESSEESESS

SEESSEESESS

ESESSEESSES

ESSEESSEED

SEESDWSENESNS

EENEEEF

ESSEESSES

SEWSSEES

EDSEWNSES

NENSEWDW

ENENEENF

## 3. Cross Words

**Problem Description**

A crossword puzzle is a square grid with black and blank squares, containing clue numbers (according to a set of rules) on some of the squares. The puzzle is solved by obtaining the solutions to a set of clues corresponding to the clue numbers.

The solved puzzle has one letter in each of the blank square, which represent a sequence of letters (consisting of one or more words in English or occasionally other languages) running along the rows (called “Across”, or “A”) or along the columns (called “Down” or “D”). Each numbered square is the beginning of an Across solution or a Down solution. Some of the across and down solutions will intersect at a blank square, and if the solutions are consistent, both of them will have the same letter at the intersecting square.

In this problem, you will be given the specifications of the grid, and the solutions in some random order. The problem is to number the grid appropriately, and associate the answers consistently with the clue numbers on the grid, both as Across solutions and as Down solutions, so that the intersecting blank squares have the same letter in both solutions.

**Rules for Clue Numbering**

The clue numbers are given sequentially going row wise (Row 1 first, and then row2 and so on)

Only blank squares are given a clue number

A blank square is given a clue number if either of the following conditions exist (only one number is given even if both the conditions are satisfied)

It has a blank square to its right, and it has no blank square to its left (it has a black square to its left, or it is in the first column). This is the beginning of an Across solution with that number

It has a blank square below it, and no blank square above it (it has a black square above it or it is in the first row). This is the beginning of a Down solution with that number

**Constraints**

5<=N<=15

5<=M<=50

**Input Format**

The input consists of two parts, the grid part and the solution part

The first line of the grid part consists of a number, N, the size of the grid (the overall grid is N x N) squares. The next N lines correspond to the N rows of the grid. Each line is comma separated, and has number of pairs of numbers, the first giving the position (column) of the beginning of a black square block, and the next giving the length of the block. If there are no black squares in a row, the pair “0,0” will be specified. For example, if a line contains “2,3,7,1,14,2”, columns 2,3,4 (a block of 3 starting with 2), 7 (a block of 1 starting with 7) and 14,15 (a block of 2 starting with 14) are black in the corresponding row.

The solution part of the input appears after the grid part. The first line of the solution part contains M, the number of solutions. The M subsequent lines consist of a sequence of letters corresponding to a solution for one of the Across and Down clues. All solutions will be in upper case (Capital letters)

**Output**

The output is a set of M comma separated lines. Each line corresponds to a solution, and consists of three parts, the clue number, the letter A or D (corresponding to Across or Down) and the solution in to that clue (in upper case)

The output must be in increasing clue number order. Ifa clue number has both an Across and a Down solution, they must come in separate lines, with the Across solution coming before the Down solution.

**Explanation**

Example 1

Input

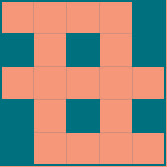
|  |
| --- |
| 5 |
| 5,1 |
| 1,1,3,1,5,1 |
| 0,0 |
| 1,1,3,1,5,1 |
| 1,1 |
| 5 |
| EVEN |
| ACNE |
| CALVE |
| PLEAS |
| EVADE |

Output

|  |
| --- |
| 1,A,ACNE |
| 2,D,CALVE |
| 3,D,EVADE |
| 4,A,PLEAS |
| 5,A,EVEN |

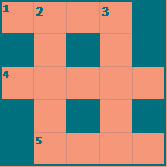
Explanation

N is 5, and the disposition of the black squares are given in the next 5 (N) lines. The grid looks like this

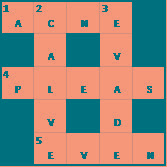
[](https://4.bp.blogspot.com/-N8vfksLEvLU/W2Rr39jIUvI/AAAAAAAAftk/-pw1dLwMGwYf0Igvf6eAIR_Qs0NjfZXxQCEwYBhgL/s1600/Cross%2BWordsimage1.png)

M=5, and there are 5 (M) solutions.

If the grid is numbered according to the rules, the numbered grid loos like this. Note that row 3 has no blanks, and the input line says “0,0”

[](https://2.bp.blogspot.com/-GT5VCxJSM2Y/W2RsJtQHKXI/AAAAAAAAfts/GK3TBXAxJKw7hZBukMx8YfBjnX_n3q9UwCLcBGAs/s1600/Cross%2BWordsimage2.png)

The solutions are fitted to the grid so that they are consistent, and the result is shown below. Note that this is consistent, because the letter at each intersecting blank square in the Across solution and the Down solution.

[](https://2.bp.blogspot.com/-2IKtffukBsM/W2RsUlxiqUI/AAAAAAAAftw/iNYMfUYlevsAIYNsfIQu2M92KFcnfY_-gCLcBGAs/s1600/Cross%2BWordsimage3.png)

Based on this the output is given in clue number order. 1 Across is ACNE, and hence the first line of the output is 1,A,ACNE. The same logic gives all the remaining solutions.

Example 2

Input

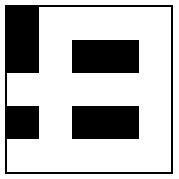
|  |
| --- |
| 5 |
| 1,1 |
| 1,1,3,2 |
| 0,0 |
| 1,1,3,2 |
| 0,0 |
| 5 |
| ASIAN |
| RISEN |
| FEAR |
| CLAWS |
| FALLS |

Output

|  |
| --- |
| 1,A,FEAR |
| 1,D,FALLS |
| 2,D,RISEN |
| 3,A,CLAWS |
| 4,A,ASIAN |

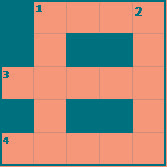
Explanation

N=5, and the grid looks like this

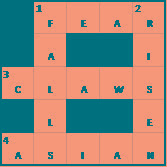
[](https://3.bp.blogspot.com/-Zl7edzWZ_rM/W2RseFiqT0I/AAAAAAAAft4/1qqeff-c_dsNK5tqAVyREM5B0tfKOagMwCLcBGAs/s1600/Cross%2BWordsimage4.jpeg)

M=5, and the 5 solutions are given

The numbered grid looks like this

[](https://1.bp.blogspot.com/-Gj33dTPGzq8/W2RskYjSNpI/AAAAAAAAft8/TemBgivgWgk3h1AfC2s8aNnB5Q_WMVXDQCLcBGAs/s1600/Cross%2BWordsimage5.png)

The consistently populated grid (with the solutions) look like this

[](https://1.bp.blogspot.com/-zQ8x0Xytt6s/W2RsrqGrOVI/AAAAAAAAfuA/UaE0TB5E8doRlNfLcpeB25i3Clsk54o6gCLcBGAs/s1600/Cross%2BWordsimage6.png)

The output can be easily given from this. Note that clue number 1 has both an Across solution (FEAR) and a DOWN solution (FALLS). The Across solution must precede the Down solution in the output.

## 4. Bank Compare

**Problem Description**

There are two banks; Bank A and Bank B. Their interest rates vary. You have received offers from both bank in terms of annual rate of interest, tenure and variations of rate of interest over the entire tenure.

You have to choose the offer which costs you least interest and reject the other.

Do the computation and make a wise choice.

The loan repayment happens at a monthly frequency and Equated Monthly Installment (EMI) is calculated using the formula given below :

EMI = loanAmount \* monthlyInterestRate /

( 1 - 1 / (1 + monthlyInterestRate)^(numberOfYears \* 12))

**Constraints**

1 <= P <= 1000000

1 <=T <= 50

1<= N1 <= 30

1<= N2 <= 30

**Input Format**

First line : P – principal (Loan Amount)

Second line : T – Total Tenure (in years).

Third Line : N1 is number of slabs of interest rates for a given period by Bank A. First slab starts from first year and second slab starts from end of first slab and so on.

Next N1 line will contain the interest rate and their period.

After N1 lines we will receive N2 viz. the number of slabs offered by second bank.

Next N2 lines are number of slabs of interest rates for a given period by Bank B. First slab starts from first year and second slab starts from end of first slab and so on.

The period and rate will be delimited by single white space.

**Output**

Your decision – either Bank A or Bank B.

**Explanation**

**Example 1**

Input

10000

20

3

5 9.5

10 9.6

5 8.5

3

10 6.9

5 8.5

5 7.9

Output

Bank B

**Example 2**

Input

500000

26

3

13 9.5

3 6.9

10 5.6

3

14 8.5

6 7.4

6 9.6

Output

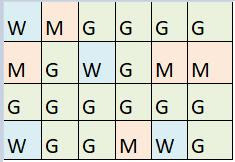
Bank B

## 5. Jurrasic Park

**Problem Description**

Smilodon is a ferocious animal which used to live during the Pleistocene epoch (2.5 mya–10,000 years ago). Scientists successfully created few smilodons in an experimental DNA research. A park is established and those smilodons are kept in a cage for visitors.

This park consists of Grasslands(G), Mountains(M) and Waterbodies(W) and it has three gates (situated in grasslands only). Below is a sample layout.

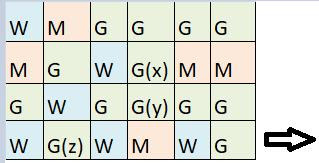
[](https://4.bp.blogspot.com/-rkomOcA2Q2g/W2RqCfnhxwI/AAAAAAAAftQ/SfgwTDrzsXgsJ6GTM9u6w_RrixkGG46egCLcBGAs/s1600/Jurrasic%2BParkimage1.png)

Before opening the park, club authority decides to calculate Safety index of the park. The procedure of the calculation is described below. Please help them to calculate.

Safety Index calculation

Assume a person stands on grassland(x) and a Smilodon escapes from the cage situated on grassland(y). If the person can escape from any of those three gates before the Smilodon able to catch him, then the grassland(x) is called safe else it is unsafe. A person and a Smilodon both take 1 second to move from one area to another adjacent area(top, bottom, left or right) but a person can move only over grasslands though Smilodon can move over grasslands and mountains.

If any grassland is unreachable for Smilodon(maybe it is unreachable for any person also), to increase safe index value Club Authority use to mark those grasslands as safe land. Explained below

[](https://4.bp.blogspot.com/-lVsQ5CxU8eA/W2RqPwPnEHI/AAAAAAAAftU/6T8pvRH-FzkckS3NJsna-n1aAUb1tfMFACLcBGAs/s1600/Jurrasic%2BParkimage2.png)

For the above layout, there is only one gate at (4,6)

Y is the position of Smilodon’s cage

X is not safe area

Z is a safe area as is it not possible for smilodon to reach z

Safety index=(total grassland areas which are safe\*100)/total grassland area

**Constraints**

3<= R,C <= 10^3

Gates are situated on grasslands only and at the edge of the park

The cage is also situated in grassland only

The position of the cage and the position of three gates are different

**Input Format**

The first line of the input contains two space-separated integers R and C, denoting the size of the park (R\*C)

The second line contains eight space-separated integers where

First two integers represent the position of the first gate

3rd and 4th integers represent the position of second gate

5th and 6th integers represent the position of third gate respectively

The last two integers represent the position of the cage

Next R lines, each contains space separated C number of characters. These R lines represent the park layout.

**Output**

Safety Index accurate up to two decimal places using Half-up Rounding method

**Explanation**

Example 1

Input

4 4

1 1 2 1 3 1 1 3

G G G G

G W W M

G G W W

M G M M

Output

75.00

Explanation

|  |  |  |  |
| --- | --- | --- | --- |
| G | G | G | G |
| G | W | W | M |
| G | G | W | W |
| M | G | M | M |
|  |  |  |  |
|  | Mountains | 4 |  |
|  | Gates- Safe Areas | 3 |  |
|  | Other Safe Areas | 3 |  |
|  | Waters | 4 |  |
|  | Cage Pos.-unsafe | 1 |  |
|  | Other unsafe areas | 1 |  |

Safety Index= (6\*100)/8

Example 2

Input

4 6

1 6 3 6 4 6 3 4

W M G G G G

M G W G M M

G W G G G G

W G W M W G

Output

69.23

## 6. String Rotation

**Problem Description**

Rotate a given String in the specified direction by specified magnitude.

After each rotation make a note of the first character of the rotated String, After all rotation are performed the accumulated first character as noted previously will form another string, say **FIRSTCHARSTRING**.

Check If **FIRSTCHARSTRING** is an Anagram of any substring of the Original string.

If yes print "YES" otherwise "NO"**.**Input

The first line contains the original string s. The second line contains a single integer q. The ith of the next q lines contains character d[i] denoting direction and integer r[i] denoting the magnitude.

**Constraints**

1 <= Length of original string <= 30

1<= q <= 10

**Output**

YES or NO

**Explanation**

Example 1

Input

carrace

3

L 2

R 2

L 3

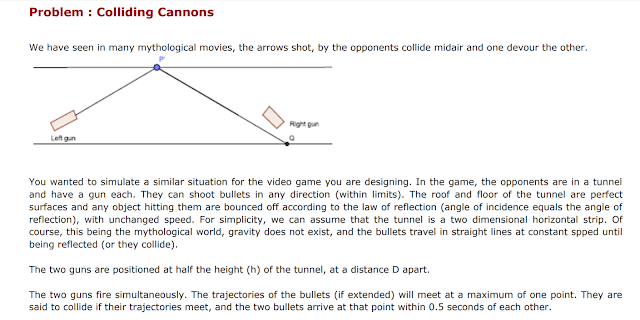
Output

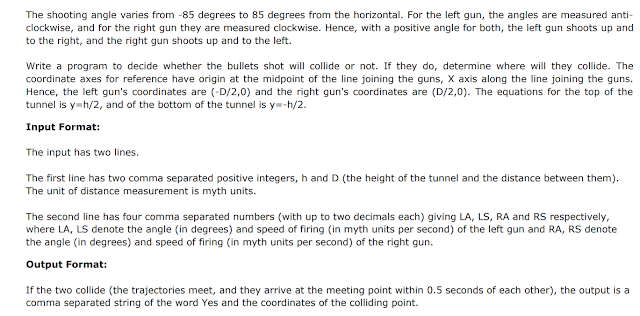
NO

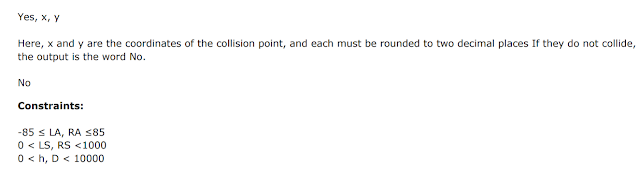
Explanation

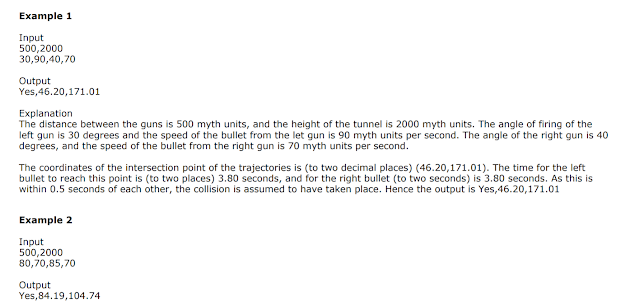
After applying all the rotations the **FIRSTCHARSTRING** string will be "rcr" which is not anagram of any sub string of original string "carrace".

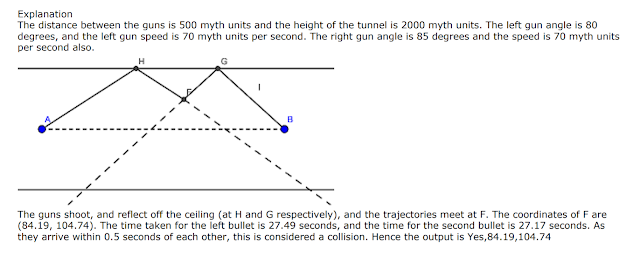
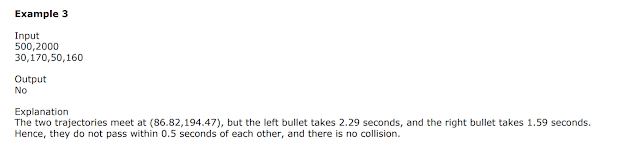
7. **Problem: Colliding Canons**

[](https://2.bp.blogspot.com/-Y_OiebP1eX8/WzHr_mPPKsI/AAAAAAAAfYo/S2C_Hm_t-HkCEw_TIBAF-f312cUoXfJOQCLcBGAs/s1600/colliding-canons-1.png)

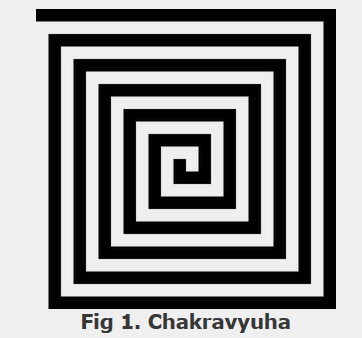
[](https://2.bp.blogspot.com/-WihE7kv9ers/WzHr_pcJOHI/AAAAAAAAfYk/OeEap2PFp2EJrlKa9VoFhdc8pNlbFmNdgCLcBGAs/s1600/colliding-canons-2.png)

[](https://2.bp.blogspot.com/-IpA6mPHfS-4/WzHr_FzMMxI/AAAAAAAAfYg/6lLwliUVyqoOqH0492rcVl7O8MzAXrtSACLcBGAs/s1600/colliding-canons-3.png)

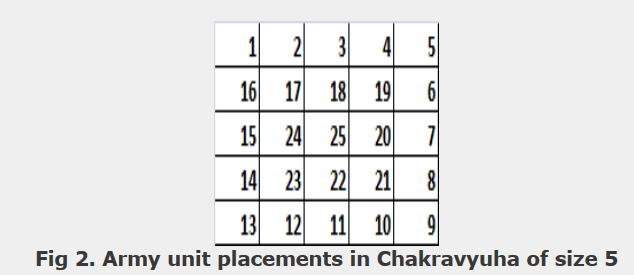
[](https://3.bp.blogspot.com/-6xGggFjp8IA/WzHsAAKUZEI/AAAAAAAAfYs/4e0SOmKQIz0sf-FG_TTCPhdaasooBUhzgCLcBGAs/s1600/colliding-canons-4.png)

[](https://4.bp.blogspot.com/-sFXxCGrN59I/WzHsAwhDDmI/AAAAAAAAfYw/xXATTgAf0nE4xfUDuWhRv7bTm786WYWfQCLcBGAs/s1600/colliding-canons-5.png)[](https://1.bp.blogspot.com/-hyRkDqlnakw/WzHsA4KRMGI/AAAAAAAAfY0/QEIZQnZSd4IhC_7DiH-hLrirmWa9Fc9MQCLcBGAs/s1600/colliding-canons-6.png)

8. During the battle of Mahabharat, when Arjuna was far away in the battlefield, Guru Drona made a Chakravyuha formation of the Kaurava army to capture Yudhisthir Maharaj. Abhimanyu, young son of Arjuna was the only one amongst the remaining Pandava army who knew how to crack the Chakravyuha. He took it upon himself to take the battle to the enemies.   
  
Abhimanyu knew how to get power points when cracking the Chakravyuha. So great was his prowess that rest of the Pandava army could not keep pace with his advances. Worried at the rest of the army falling behind, Yudhisthir Maharaj needs your help to track of Abhimanyu's advances. Write a program that tracks how many power points Abhimanyu has collected and also uncover his trail   
  
A Chakravyuha is a wheel-like formation. Pictorially it is depicted as below 

[](https://4.bp.blogspot.com/-s2i3h6j5FqA/WySYchPAdSI/AAAAAAAAfCo/XFhkkO4bYdsBYARIq16-MOVfSPe5l90vQCPcBGAYYCw/s1600/chakravyua.png)

A Chakravyuha has a very well-defined co-ordinate system. Each point on the co-ordinate system is manned by a certain unit of the army. The Commander-In-Chief is always located at the center of the army to better co-ordinate his forces. The only way to crack the Chakravyuha is to defeat the units in sequential order.  
  
A Sequential order of units differs structurally based on the radius of the Chakra. The radius can be thought of as length or breadth of the matrix depicted above. The structure i.e. placement of units in *sequential order* is as shown below

[](https://3.bp.blogspot.com/-sLRCfd2K2rI/WySY8WUb7rI/AAAAAAAAfCw/PvcsyXCT17s38Vf1Ne7Qv4-slh3TthWNQCLcBGAs/s1600/chakravyuh.png)

The entry point of the Chakravyuha is always at the (0,0) co-ordinate of the matrix above. This is where the 1st army unit guards. From (0,0) i.e. 1st unit Abhimanyu has to march towards the center at (2,2) where the 25th i.e. the last of the enemy army unit guards. Remember that he has to proceed by destroying the units in sequential fashion. After destroying the first unit, Abhimanyu gets a power point. Thereafter, he gets one after destroying army units which are multiples of 11. You should also be a in a position to tell Yudhisthir Maharaj the location at which Abhimanyu collected his power points.   
  
**Input Format:**First line of input will be length as well as breadth of the army units, say N  
  
**Output Format:**

* Print NxN matrix depicting the placement of army units, with unit numbers delimited by (\t) Tab character
* Print Total power points collected
* Print coordinates of power points collected in sequential fashion (one per line)

**Constraints:**0 < N <=100

**Sample Input and Output**

|  |  |  |
| --- | --- | --- |
| **SNo.** | **Input** | **Output** |
| 1 | 2 | 1 2 4 3 Total Power points : 1 (0,0) |
| 2 | 5 | 1 2 3 4 5 16 17 18 19 6 15 24 25 20 7 14 23 22 21 8 13 12 11 10 9 Total Power points : 3 (0,0) (4,2) (3,2) |

9. **Problem : Exam Efficiency**

In an examination with multiple choice questions, the following is the exam question pattern.

* X1 number of One mark questions, having negative score of -1 for answering wrong
* X2 number of Two mark questions, having negative score of -1 and -2 for one or both options wrong
* X3 number of Three mark questions, having negative score of -1, -2 and -3 for one, two or all three options wrong
* Score Required to Pass the exam : Y
* For 1,2 and 3 mark questions, 1,2 and 3 options must be selected. Simply put, once has to attempt to answer all questions against all options.

Identify the minimum accuracy rate required for **each type of question** to crack the exam.  
  
Calculations must be done up to 11 precision and printing up to 2 digit precision with ceil value

**Input Format:**  
  
First line contains number of one mark questions denoted by X1,  
Second line contains number of two mark questions denoted by X2  
Third line contains number of three mark questions denoted by X3  
Fourth line contains number of marks required to pass the exam denoted by Y.

**Output Format:**  
  
Minimum Accuracy rate required for one mark question is 80%  
Minimum Accuracy rate required for Two mark question is 83.33%  
Minimum Accuracy rate required for Three mark question is 90%  
  
  
Note: - If the mark required to pass the exam can be achieved by attempting without attempting any particular type of question then show message similar to, One mark question need not be attempted, so no minimum accuracy rate applicable  
  
See Example Test cases for better understanding.

**Sample Input and Output**

|  |  |  |  |
| --- | --- | --- | --- |
| **SNo.** | **Input** | **Output** | **Explaination** |
| 1 | 20 30 30 120 | One mark questions need not be attempted, so no minimum accuracy rate applicable. Minimum Accuracy rate required for Two mark question is 58.33% Minimum Accuracy rate required for Three mark question is 72.23% | If one got full marks in two marks question and three marks question then total accuracy can be 0 in one mark question  In same way it will be done for two marks and three marks question |
| 2 | 20 30 30 170 | Minimum Accuracy rate required for one mark question is 100% Minimum Accuracy rate required for Two mark question is 100% Minimum Accuracy rate required for Three mark question is 100% | If one got full marks in two marks question and three marks question then total accuracy should be 100% in one mark question to pass the exam.  In same way it will be done for two marks and three marks question |

10. **Problem : Calculate Salary And PF**

**Statement :**  
  
Calculate the **Final Salary** & **Final Accumulated PF** of an Employee working in ABC Company Pvt. Ltd. The Company gives two Increments (i.e. ***Financial Year Increment & Anniversary Increment***) to an Employee in a Particular Year.  
  
The Employee must have Completed 1 Year to be Eligible for the Financial Year Increment. The Employee who are joining in the month of Financial Year Change (i.e. April) are considered as the Luckiest Employee's, because after completion of 1 Year, they get Two Increments  
(Financial Year Increment & Anniversary Increment).  
  
Rate of Interest for the Financial Year Increment = 11%.  
Rate of Interest for the Anniversary Increment = 12%.  
  
From 4th Year, the Financial Year Increment will be revised to 9%.  
From 8th Year, the Financial Year Increment will be revised to 6%.  
The Company is giving special Increment for the Employee who have completed 4 years & 8 years respectively.  
  
So, the Anniversary Increment of the Employee for the 4th Year will be 20% and the Anniversary Increment of the Employee for the 8th year will be 15%.  
  
Calculate the Final Salary after N number of Years as well as Calculate the Accumulated PF of the Employee after N number of Years.  
  
Please Note that, the Rate of Interest for calculating PF for a Particular Month is 12%. Moreover, take the upper Limit of the amount if it is in decimal (For e.g. - If any Amount turns out to be 1250.02, take 1251 for the Calculation.)

**Input Format:**

1. Joining Date in dd/mm/yy format
2. Current CTC.
3. Number of Years for PF & Salary Calculation.

**Output Format:**

1. Salary after the Specified Number of Years (i.e. CTC after N number of Years) in the following format  
   Final Salary =
2. Accumulated PF of the Employee after N number of Years in the following format  
   Final Accumulated PF =

**Constraints:**

1. **Calculation should be done upto 11 digit precision and output should be printed with ceil value**

**Sample Input and Output**

|  |  |  |
| --- | --- | --- |
| **SNo.** | **Input** | **Output** |
| 1 | 5 01/01/2016 10000 2 | Final Salary = 13924 Final Accumulated PF = 2655 |
| 2 | 19/01/2016 6500 4 | Final Salary = 14718 Final Accumulated PF = 4343 |

11. **Problem : House Numbers**

There are a number of buildings of different sizes (width) and with different number of floors. Each may have zero, one or more flats marked with a different character, say $. Identify these flats and print the address in form of building number, floor and flat number.  
  
Buildings are separated from each other by a space character. Floors start from 1 i.e. the lowest floor is the first floor. Also flat numbers are assigned from left to right. The $ character is the flat of interest for which we have to find the address.  
  
For example,  
  
  
The above is an example of a colony with 4 buildings. The first building has 5 floors with 4 flats on each floor. The flat marked by $ in the first building can be identified as 134 (building 1, floor 3 and flat 4). Similarly, the flat marked by $ in the second building can be identified as 223 (building 2, floor 2 and flat 3).  
  
A file containing a random colony will be given as an input to your program. You have to identify all flats marked by $ in the colony. Note that there is no limit to the number of buildings, floors or flats.

**Input Format:**  
  
Text file path

**Output Format:**  
  
Print Building number(B), floor number(F), Flat numbers(R) of all flats marked by $ in the following format:

**Constraints:**

**Print order for $-marked flats should be from top floors to bottom floors and left building to right building**

**Sample Input and Output**

|  |  |  |  |
| --- | --- | --- | --- |
| **SNo.** | **Input** | **Contents Of Input File** | **Output** |
| 1 | /tmp/T1.txt |  | 232 |
| 2 | /tmp/T2.txt | > | 152 142 341 221 223 111 213 |

12. **Problem : ISL Schedule**

The Indian Soccer League (ISL) is an annual football tournament.  
The group stage of ISL features **N** teams playing against each other with following set of rules:

1. **N** teams play against each other twice - once at Home and once Away
2. A team can play only one match per day
3. A team cannot play matches on consecutive days
4. A team cannot play more than two back to back Home or Away matches
5. Number of matches in a day has following constraints  
   * The match pattern that needs to be followed is -  
     + Day 1 has two matches and Day 2 has one match,
     + Day 3 has two matches and Day 4 has one match and so on
   * There can never be 3 or more matches in a day
6. Gap between two successive matches of a team cannot exceed floor(N/2) days where floor is the mathematical function floor()
7. Derby Matches (any one)  
   * At least half of the derby matches should be on weekend
   * At least half of the weekend matches should be derby matches

Your task is to generate a schedule abiding to above rules. 

**Input Format:**  
  
First line contains number of teams (**N**).  
Next line contains state ID of teams, delimited by space

**Output Format:**  
  
Match format: **Ta-vs-Tb**  
where **Ta** is the home team with id **a** and **Tb** is the away team with id **b**.  
For each day print the match(es) in following format:-

* Two matches:- "**#D Ta-vs-Tb Tm-vs-Tn**"
* One match:- "**#D Tx-vs-Ty**"

where D is the day id and [a, b, m, n, x, y] are team ids. 

**Constraints:**

**1. 8 <= N <= 100**

**Note :** 

* Team ids are unique and have value between 1 to N
* Day id starts with 1
* Every 6th and 7th day are weekends
* Derby is a football match between two teams from the same state

**Sample Input and Output**

|  |  |  |
| --- | --- | --- |
| **SNo.** | **Input** | **Output** |
| 1 | 8 1 2 5 4 3 1 6 6 | #1 T1-vs-T6 T3-vs-T5 #2 T7-vs-T4 #3... and so on |

Note:- There can be multiple correct answers for the same test cases. For better understanding of test case refer this [PDF](https://www.tcscodevita.com/CodevitaV5/images/PTC.PDF). This PDF contains one of the correct answer for a test case. 

**Explanation:**  
  
There are 8 teams with following information:-

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Team ID | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| State ID | 1 | 2 | 5 | 4 | 3 | 1 | 6 | 6 |

13. **Problem : Longest Possible Route**

Given an MxN matrix, with a few hurdles arbitrarily placed, calculate the cost of longest possible route from point A to point B within the matrix.

**Input Format:**

1. First line contains 2 numbers delimited by whitespace where, first number M is number of rows and second number N is number of columns
2. Second line contains number of hurdles H followed by H lines, each line will contain one hurdle point in the matrix.
3. Next line will contain point A, starting point in the matrix.
4. Next line will contain point B, stop point in the matrix.

**Output Format:**  
  
Output should display the length of the longest route from point A to point B in the matrix.

**Constraints:**

1. **The cost from one position to another will be 1 unit.**
2. **A location once visited in a particular path cannot be visited again.**
3. **A route will only consider adjacent hops. The route cannot consist of diagonal hops.**
4. **The position with a hurdle cannot be visited.**
5. **The values MxN signifies that the matrix consists of rows ranging from 0 to M-1 and columns ranging from 0 to N-1.**
6. **If the destination is not reachable or source/ destination overlap with hurdles, print cost as -1.**

**Sample Input and Output**

|  |  |  |  |
| --- | --- | --- | --- |
| **SNo.** | **Input** | **Output** | **Explaination** |
| 1 | 3 10 3 1 2 1 5 1 8 0 0 1 7 | 24 | Here matrix will be of size 3x10 matrix with a hurdle at (1,2),(1,5 ) and (1,8) with starting point A(0,0) and stop point B(1,7)  3 10 3 -- (no. of hurdles ) 1 2 1 5 1 8 0 0 -- (position of A) 1 7 -- (position of B)  So if you examine matrix below shown in Fig 1, total hops  ( ->) count is 24. So final answer will be 24. No other route longer than this one is possible in this matrix. |
| 2 | 2 2 1 0 0 1 1 0 0 | -1 | No path is possible in this 2\*2 matrix so answer is -1 |

### 14. **Min Product Array**

The task is to find the minimum sum of Products of two arrays of the same size, given that k modifications are allowed on the first array. In each modification, one array element of the first array can either be increased or decreased by 2.  
  
Note- the product sum is Summation (A[i]\*B[i]) for all i from 1 to n where n is the size of both arrays

**Input Format:**

1. First line of the input contains n and k delimited by whitespace
2. Second line contains the Array A (modifiable array) with its values delimited by spaces
3. Third line contains the Array B (non-modifiable array) with its values delimited by spaces

**Output Format:**  
  
Output the minimum sum of products of the two arrays

**Constraints:**

1. **1 ≤ N ≤ 10^5**
2. **0 ≤ |A[i]|, |B[i]| ≤ 10^5**
3. **0 ≤ K ≤ 10^9**

**Sample Input and Output**

|  |  |  |
| --- | --- | --- |
| **SNo.** | **Input** | **Output** |
| 1 | 3 5 1 2 -3 -2 3 -5 | -31 |
| 2 | 5 3 2 3 4 5 4 3 4 2 3 2 | 25 |

**Explanation for sample 1:**  
  
Here total numbers are 3 and total modifications allowed are 5. So we modified A[2], which is -3 and increased it by 10 (as 5 modifications are allowed). Now final sum will be  
(1 \* -2) + (2 \* 3) + (7 \* -5)  
-2 + 6 - 35  
-31  
  
-31 is our final answer.  
  
**Explanation for sample 2:**  
  
Here total numbers are 5 and total modifications allowed are 3. So we modified A[1], which is 3 and decreased it by 6 (as 3 modifications are allowed).  
Now final sum will be  
(2 \* 3) + (-3 \* 4) + (4 \* 2) + (5 \* 3) + (4 \* 2)  
6 - 12 + 8 + 15 + 8  
25  
  
25 is our final answer.   
  
**Simplified Pseudo Code:**  
1. Initialize maxDiff = 0, minimumSum = 0  
2. For i to n  
    i. product = A[i] \* B[i]  
    ii. if ( product < 0 && B[i] < 0 ) then  
            temp = (A[i] +  2  \* k ) \* B[i]  
        else if( product < 0 && A[i] < 0) then  
            temp = (A[i] - 2 \* k) \* B[i]  
        else if( product > 0 && A[i] < 0) then  
               temp = (A[i] + 2 \* k) \* B[i]  
        else if (product > 0 && A[i] > 0)  
               temp = (A[i] - 2 \* k)  \* B[i]  
    iii. diff =  abs(product - temp)  
    iV. if( diff > maxDiff )  
         maxDiff = diff  
     V. minimumSum = minimumSum + product  
  
3. minimumSum = minimumSum - maxDiff

15. **Problem : Consecutive Prime Sum**

Some prime numbers can be expressed as Sum of other consecutive prime numbers.  
For example  
  
5 = 2 + 3  
17 = 2 + 3 + 5 + 7  
41 = 2 + 3 + 5 + 7 + 11 + 13  
  
Your task is to find out how many prime numbers which satisfy this property are present in the range 3 to N subject to a constraint that summation should always start with number 2.  
  
Write code to find out number of prime numbers that satisfy the above mentioned property in a given range.

**Input Format:**  
  
First line contains a number N

**Output Format:**  
  
Print the total number of all such prime numbers which are less than or equal to N.

**Constraints:**

**1. 2**

**Sample Input and Output**

|  |  |  |  |
| --- | --- | --- | --- |
| **SNo.** | **Input** | **Output** | **Comment** |
| 1 | 20 | 2 | (Below 20, there are 2 such numbers: 5 and 17). 5=2+3 17=2+3+5+7 |
| 2 | 15 | 1 |  |

**Pseudo Code:**  
1. Find all the prime numbers till N  
    i.  A  is an array length N + 1 initialized with numbers from 0 to N                 (0,1,2, ..., N)  
    ii. initialize p = 2  
        a.  for p = 2 to p \* p <= N ; p++  
               if A[p] != 0 then  
                  for  i = p \* 2; i <= N; i += p  
                         A[i] = 0 // mark non prime as 0  
2.  sum = 5, count = 0  
3. for j = 5 to j <= N;  j = j+2  
    i.  if  ( (A[j]  != 0 && A[j] = sum) || A[j] = -1)  
        count = count + 1  
    ii. if (A[j] != 0 || A[j] == -1)  
        sum = sum + j  
        if ( A[sum] != 0) // if A[sum] is prime  
            A[sum] = -1 // mark A[sum] as sum of consecutive  
4. print count  
  
16. **Problem : Logic Pyramid**

Identify the logic behind the series  
  
6 28 66 120 190 276....  
  
The numbers in the series should be used to create a Pyramid. The base of the Pyramid will be the widest and will start converging towards the top where there will only be one element. Each successive layer will have one number less than that on the layer below it. The width of the Pyramid is specified by an input parameter N. In other words there will be N numbers on the bottom layer of the pyramid.  
  
The Pyramid construction rules are as follows

1.    First number in the series should be at the top of the Pyramid

2.    Last N number of the series should be on the bottom-most layer of the Pyramid, with Nth number being the right-most number of this layer.

3.    Numbers less than 5-digits must be padded with zeroes to maintain the sanctity of a Pyramid when printed. Have a look at the examples below to get a pictorial understanding of what this rule actually means.

Example  
  
**If input is 2, output will be**  
  
 00006  
00028 00066  
  
**If input is 3, output will be**  
  
  00006  
 00028 00066  
00120 00190 00276  
  
Formal input and output specifications are stated below

**Input Format:**   
  
First line of input will contain number N that corresponds to the width of the bottom-most layer of the Pyramid

**Output Format:**   
  
The Pyramid constructed out of numbers in the series as per stated construction rules

**Constraints:**

**1.    0 < N <= 14**

**Sample Input and Output**

|  |  |  |
| --- | --- | --- |
| **SNo.** | **Input** | **Output** |
| 1 | 2 |  |
| 2 | 3 |  |

**Pseudo Code:**   
The series is subset of **Hexagonal Number**generated using formula: n \* ( 2 \* n -1 )  
  
If n is only even number then you will get the series as 6, 28, 66, 120, 190, 276, 378,  496, 630, 780, 946, 1128, 1326, 1540 and so on.  
  
1. for i = 0 to i  < N; i++  
    a. print blank space for (N- i - 1) times  
    for j = 0 to j <= i ; j++   
            a. n = ( i + j +1 ) \* 2   
        b. num =  n  \* ( 2 \* n  - 1 )    
        c. if number of digit in num < 5  
               print 0 for (number of digit in num - 5) times  
        d. print num  
        e. if (  j < i )  
               print space