New Zealand Programming Contest 2022

PROBLEM SET

PREAMBLE

Please note the following very important details relating to your solutions:

- Read all input from the keyboard, i.e. use `stdin`, `System.in`, `cin`, `Console.ReadLine`, `input` or equivalent. Input will be redirected from a file to form the input to your submission.
- Do NOT prompt for input as this will appear in your output and cause a submission to be judged as wrong.
- Write all output to the screen, i.e. use `stdout`, `System.out`, `cout`, `Console.Write`, `print` or equivalent. Do not write to `stderr`. Do NOT use, or even include, any module that allows direct manipulation of the screen, such as `conio`, `Crt` or anything similar.
- Output from your program is redirected to a file for later checking. Use of direct I/O means that such output is not redirected and hence cannot be checked. This could mean that a correct program is rejected! You have been warned.
- Unless otherwise stated, all *integers* will fit into a standard 32-bit computer word. If more than one integer appears on a line, they will be separated by white space, i.e. spaces or tabs.
- An *uppercase letter* is a character in the sequence 'A' to 'Z'. A *lower case letter* is a character in the sequence 'a' to 'z'. A *letter* is either a lower case letter or an upper case letter.
- Unless otherwise stated, a *word* or a *name* is a continuous sequence of letters.
- Unless otherwise stated, a *string* is a continuous sequence of visible characters.
- Unless otherwise stated, *words* and *names* will contain no more than 60 characters, and *strings* will contain no more than 250 characters.
- It is not necessary to check that input data meets the criteria specified – it will always do so. Unless the problem specifies that you need to check the validity of the data, there is no need to do so.
- If it is stated that ‘a line contains no more than *n* characters’, this does not include the character(s) specifying the end of line.
- Input files are often terminated by a ‘sentinel’ line, followed by an end of file marker. This line should not be processed.
- Make sure all your code is in a single source code file.
- Make sure you submit the source code file to DOMjudge not a solution or project file.
- DOMjudge may work out the language being used based on the source code file extension, but if not make you choose the correct language.

PROBLEM A — JELLY BEANS — 3 POINTS

A popular event at fairs and fundraisers is "How many jelly beans are in the jar?" Participants guess the number and the one who comes closest wins the jar. If two people are equally close, the first one who guessed wins the prize.

Input

The first line of input is the number of jelly beans in the jar. This will be between 50 and 1,000 inclusive.

The second line is the number of people who had a guess, a number between 1 and 250 inclusive.

The guesses then follow, two lines per guess.

- The first line of each guess is the name of the person who guessed. Each name is a single word as defined in the preamble. Names are not repeated.
- The second line is the person's guess, a positive integer.

Output

Output consists of a single line containing just the name of the winner.

Sample Input

```
256
8
Alan
240
Betty
232
Cheng
253
Dimitri
284
Evan
300
Fathima
258
Gurtrude
206
Harry
238
```

Output for Sample Input

```
Fathima
```

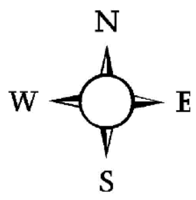
Explanation

Fathima's guess was just 2 away from the correct answer, beating Cheng who was 3 away.

PROBLEM B **BRIDGE** **3 POINTS**

Bridge is a popular card game played all round the world. A game uses a standard pack of 52 playing cards. Two pairs of players compete to score the most points.

North (N) and South (S) play East (E) and West (W). One player deals the cards one at a time so that each player holds 13 cards. The first card is dealt to the player to the left of the dealer, the last to the dealer. During play there are 13 rounds (or tricks) – the best card in each trick winning the trick.



Before playing the cards, players have to bid how many tricks they will win. There are many bidding systems that may be used; several of these evaluate a hand by assigning points to the best cards. In the system we are going to use in this problem, each ace (A) scores 4 points, each king (K) scores 3 points, each queen (Q) scores 2 points and each jack (J) scores 1 point. Your job is to assign the correct cards to each player in a deal, and evaluate their hands using the specified points system.

Input

Input consists of two lines.

On line 1 is a single letter (N, S, E or W) showing which player is dealing.

On line 2 are 52 characters showing the order of the cards in the pack. Apart from A, K, Q and J, t represents a 10, and numbers 2 to 9 represent cards with those numbers. There are 4 of each valued card in the pack. Suits are ignored.

Output

Output consists of a single line showing the points value of the cards held by the 4 players, separated by spaces. The dealer will be first with their points, the other players following in clockwise order. As a check, note that the total points must be 40.

Sample Input

```
N
7534K66K72QK953769t859QK8A4J54Q7AQt42A23At8JJ29t83J6
```

Output for Sample Input

```
N 11 E 12 S 10 W 7
```

Explanation

Cards are dealt as follows:

`E 7K796585A2AJ8`	2 aces (8), 1 king (3), 1 jack (1)	= 12
`S 562599A4QAt23`	2 aces (8), 1 queen (2)	= 10
`W 36Q3tQ4Qt289J`	3 queens (6), 1 jack (1)	= 7
`N 4KK78KJ743Jt6`	3 kings (9), 2 jacks (2)	= 11

PROBLEM C GRAFFITI 3 POINTS

In a small town in New Zealand, a lot of strange graffiti has been appearing. The local authorities have employed a graffiti expert who has determined that the graffiti follows 3 rules:

1. There are 3 distinct images which we will represent with the characters #, & and %.
2. A first image is selected and it is drawn once or twice.
3. One or more times, an image is chosen which is different from the last one used and, one or more times, drawn once to the left and once to the right of the previous image(s).

Lately, several copycat graffiti artists have been at work, but they are not following the established rules.

Your job is to analyse graffiti examples from around the town and tell the local authority if they are genuine or copycat.

Input

Input consists of a number of lines of graffiti. The first line is a positive integer, N (0< N<=15) which tells how many lines of graffiti follow.

The next N lines each contain at least 1 but no more than 20 characters (each being #, & or %), each line representing one piece of graffiti.

Output

For each line of input, output a one word verdict. If the graffiti example follows the rules, the output should be the word `genuine`. Otherwise the output should be the word `copycat`.

Sample Input	Output for Sample Input	Explanation
6		
#	copycat	No second image
&	copycat	No second image
#&	copycat	Only 1 of second image
%%	copycat	No second image
##%#&	copycat	Mismatch for second image
&%&	genuine	Minimum length

PROBLEM D **WINNING THE TOSS** **3 POINTS**

In a golf croquet tournament, a team will play several games to determine who wins the tournament. In most tournaments a team will expect to play 5 or 6 games in a single day, but major tournaments can involve more games over multiple days.

Often two games are played on a lawn at the same time, so to avoid confusion primary and secondary colours are used. With primary colours, blue and black play red and yellow with blue starting. With secondary colours, green and brown play pink and white with green starting.

To determine who starts, one player tosses a coin and a player on the other side calls. The winning team starts and takes the appropriate colours – blue and black if primary colours are being used, brown and green for secondary colours.

Input

Input consists of data for a single team in one tournament.

The first line of input is a positive integer, N (3< N<=15) which tells how many games are to be played.

The next N lines each contain data for one game, items being separated by single spaces. Each line starts with the number 1 or 2, 1 meaning primary colours are being used, 2 meaning secondary colours. Next follow 2 upper case letters, each being either H or T (for heads or tails). The first is what the team needs to win the toss[1], the second is how the coin landed.

Output

For each line of input, output the colours to be used by the team in that game.

Sample Input	Output for Sample Input
6	
2 H H	green and brown
1 H T	red and yellow
1 T H	red and yellow
2 H T	pink and white
2 T T	green and brown
1 H H	blue and black

[1] If the team called this would be their call. If the team tossed it would be what their opponent did **not** call!

PROBLEM E **WELL BEHAVED CHILDREN** **10 POINTS**

The Super Tots chain of pre-schools are introducing awards for good behaviour to encourage their children to be good. Each participating pre-school will use the same set of criteria and award prizes to their top 3 children each week. Teachers are required to make sure that there are no ties for the top 3 places!

Super Tots managers have come up with the following list of behaviours they wish to encourage:

Behaviour	Code	Merit points
Eating at least 1 piece of fruit for lunch	F	20
Tidying up without being told	T	25
Walking, not running, inside	W	15
Helping another child	H	30
Flushing the toilet after use	L	25
Sitting nicely at mat time	S	20

The weekly prizes are

1. Being star of the week. Child has a special cup and a special lunch place.
2. Being able to take a music CD home for the weekend
3. Being karakia leader for a day.

Input

The first line of input is a single integer, N, the number of children in the class. 5 <= N <= 20.

N lines then follow, each being the name of 1 child. Name is defined in the preamble.

The next line of input is a single integer, B, the number of good behaviour events which have been recorded. 3 <= B <= 50.

B lines then follow. Each line has the name of 1 child, followed by one of the behaviour codes listed above.

The merit points for each child have to be totalled.

Output

3 lines of output are required.

Line 1 is of the form “`Star of the week is <name>.`” where `<name>` is the name of the child with most merit points.

Line 2 is of the form “`Music CD for <name>.`” where `<name>` is the name of the child with the second most merit points.

Line 3 is of the form “`Karakia leader is <name>.`” where `<name>` is the name of the child with third most merit points.

Sample Input

```
10
Anna
Daniel
Charlotte
Alexis
Charles
Riley
Eli
Madison
Owen
Jack
15
Daniel F
Anna S
Riley H
Riley T
Anna W
Alexis H
Anna F
Charlotte L
Madison F
Riley H
Daniel H
Charlotte W
Anna W
Madison L
Charles T
```

Output for Sample Input

```
Star of the week is Riley.
Music CD for Anna.
Karakia leader is Daniel.
```

Explanation

Riley has H, T, H total 85 points.

Anna has S, W, F, W total 70 points.

Daniel has F, H total 50 points.

PROBLEM F **BATTLESHIPS** **10 POINTS**

Battleships is a game which originally used pencil and paper to simulate a naval battle. It is over 100 years old. It has been played by children all over the world, and may have been played by Russian naval officers as long ago as 1890.

Wikipedia tells us that "It is played on ruled grids (paper or board) on which each player's fleet of ships (including battleships) are marked. The locations of the fleets are concealed from the other player. Players alternate turns calling 'shots' at the other player's ships, and the objective of the game is to destroy the opposing player's fleet.".

This problem requires you to simulate one side of the game, shots of player A against player B's fleet. In our game, the fleet consists of the following vessels:

Vessel	**Size**
Battleship	5
Destroyer	4
Cruiser	3
Submarine	2

All vessels are properly formed:

• vessels must be arranged on consecutive squares vertically, horizontally or diagonally

• vessels must not overlap

• vessels may not bend

• vessels must be complete.

On the next page you will see a diagram of a legal fleet with a game in progress.

You are required to assess each shot against the fleet. Each square is defined by its column letter (A to J inclusive) and its row number (1 to 10 inclusive).

Input

The first 4 lines define the fleet, the rest define the shots against the fleet. The position of each vessel is defined by the squares of its 2 ends. The first given square will be the lower of the two column letters or, for vertically arranged vessels, the lower of its two row numbers. All square data on a line will be space separated.

The first line defines the position of the battleship.

The second line defines the position of the destroyer.

The third line defines the position of the cruiser.

The fourth line defines the position of the submarine.

Each remaining row contains data for a single square, where an enemy shot lands. Shots will not be repeated. The final entry will be `X0`, which should not be evaluated.

Continued

Output

For each shot against the fleet there must be a single line of output. It will be either

`Miss` or

`Hit <type of vessel> <layout of vessel>`

`<layout of vessel>` will be Vertical, Horizontal or Diagonal

Sample Input

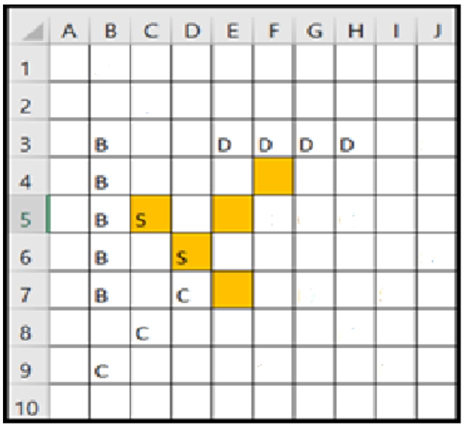
```
B3 B7
E3 H3
B9 D7
C5 D6
F4
D6
E7
E5
C5
X0
```

Output for Sample Input

```
Miss

Hit Submarine Diagonal

Miss

Miss

Hit Submarine Diagonal
```

Explanation

The layout of the fleet is as shown with shots shaded.



	A	B	C	D	E	F	G	H	I	J
1										
2										
3		B			D	D	D	D		
4		B								
5		B	S							
6		B		S						
7		B		C						
8			C							
9		C								
10										

PROBLEM G **FACTORISATION** **10 POINTS**

You all know from year 11 Algebra that you must be able to factorise quadratics; but just to jog your memory, here's some examples:

a) $x^2 - 5x + 6 = (x - 2)(x - 3)$
b) $x^2 + 10x + 24 = (x + 6)(x + 4)$
c) $x^2 - 6x - 27 = ???$

$$x^2 - 12x + 35$$

A = 1, B = −12, C = 35

$$x = \frac{-B \pm \sqrt{B^2 - 4AC}}{2A}$$

$$x = \frac{-(-12) \pm \sqrt{(-12)^2 - 4(1)(35)}}{2(1)}$$

$$= \frac{12 \pm \sqrt{144 - 140}}{2} = \frac{12 \pm \sqrt{4}}{2} = \frac{12 \pm 2}{2} = 6 \pm 1$$

$$x = 5 \quad \text{or} \quad x = 7$$

$$x^2 - 12x + 35 = (x - 5)(x - 7)$$

.

In Maths class you are told that you need to find pairs of factors of the last number. In example c, that's -27. Notice that the "subtract 27" is converted into the value -27.

Here are the possible pairs of factors of -27:

(1,-27) (3, -9) (-3, 9) (-1,27)

And now from these, find the pair of factors which add to give the coefficient of the x in the middle term. In example c, this is -6. The second pair (3,-9) add to give -6.

So we have found the numbers to put in the factors:

3 times -9 = -27 (the last number)

and

3 + -9 = -6 (the x coefficient)

Thus $x^2 - 6x - 27 = (x + 3)(x + -9)$

But we never write those last digits as negative. The negative factor becomes the positive value but the sign becomes "subtract". The correct factors are

$x^2 - 6x - 27 = (x - 9)(x + 3)$

Input

For this problem you have to find the factors of quadratic expressions which are of the form

```
x squared {+/-} ax {+/-} c
```

{+/-} indicates an operator present will be '+' or '-'

a is the coefficient for the x term and is guaranteed to be a positive integer. c is also a positive integer, no greater than 144.

In this problem it will always be possible to find a pair of factors.

Output

You should output the factors in the form

`(x + m)(x + n)`

The '+' may be '-'.

The + or – sign is separated from the other characters by spaces.

The brackets, (and), are not separated from the other characters, or each other, by spaces.

The first factor's number, m, is greater than or equal to the second factor's number, n. If m and n are equal, a + term comes before a – term where appropriate.

Sample Input 1

`x squared - 4x + 4`

Output for Sample Input 1

`(x - 2)(x - 2)`

Sample Input 2

`x squared - 6x - 27`

Output for Sample Input 2

`(x - 9)(x + 3)`

Sample Input 3

`x squared + 2x - 8`

Output for Sample Input 3

`(x + 4)(x - 2)`

PROBLEM H | **POINTLESS** | **10 POINTS**

Pointless is a UK TV game show in which 4 pairs of contestants try to find the most obscure answers to questions posed to 100 people before the show.

In each of rounds 1 and 2, one member of team 1 answers first followed by one member of each other team in order. Then the other member of the last team answers followed by the other member of the other teams in the opposite order.

If a contestant gives a wrong answer, they score 100 points, otherwise they score the number of the 100 people who gave their answer. A pointless answer is one which none of the 100 people knew, and adds £250 to the jackpot.

At the end of each round, the team with the highest combined score is eliminated.

There are 2 further rounds in the game, but these are not being covered in this problem. The jackpot is the prize for success in round 4.

Input

The first 4 lines each contain the names of two team members. Each pair will consist of no more than 32 characters. The names represent teams 1 to 4 in order. They line up right to left on the platform.

Line 5 contains a single integer it being the current jackpot. It will be an integer not lower than 1000.

Line 6 will contain 8 numbers in the range 0 to 100 inclusive, separated by spaces. These will be the scores for each answer in round 1. The pair with the highest score is eliminated.

Line 7 will contain 6 numbers in the range 0 to 100 inclusive, separated by spaces. These will be the scores for each answer in round 2. The pair with the highest score is eliminated.

Games have been selected such that there are no ties for highest score at the end of each round.

Output

At the end of rounds 1 to 2, the name of the eliminated team must be displayed on a line of its own.

The current jackpot must then be displayed on a new line.

Sample Input

```
Joe & Paul

Sally & Mike

Russel & Susan

Jane & Valerie

2000

75 27 100 10 54 12 0 32

23 31 6 7 12 79
```

Output for Sample Input

```
Round 1: Russel & Susan are eliminated.

Round 2: Joe & Paul are eliminated.

Jackpot is GBP2250.
```

Explanation

In round 1, Russel and Susan (team 3) score 100 + 12, 112 and are eliminated.

In round 2, Joe and Paul (team 1) score 23 + 79, 102 and are eliminated.

The jackpot had started at £2000, but £250 was added for the pointless answer in round 1.

Problem I **Farmer Party** **30 Points**

A group of farmers live in houses on a long country road. They decide to get together to watch the Rugby Test on TV. But there are too many of them to fit into a single house, so they decide to designate two of their group to each host a party at their house. All other farmers will choose to go to the house that is closest to them, picking arbitrarily if they are the same distance from both.

The farmers do not have EVs and are anxious to minimise their petrol costs. The distances of all houses from the end of their road are known. Compute the minimum total distance that the farmers must travel on their return trips if the two party house locations are chosen optimally.

Input

The first line of input contains a single integer n ($2 \leq n \leq 50$) which is the number of farmers.

Each of the next n lines contains a single integer x ($0 \leq x \leq 2\ 000$), which is the distance of the house from the end of the road. Each location will be distinct.

Output

Output consists of a single line containing just the minimum total distance travelled by the farmers to and from their selected parties, given that the two party house locations are chosen optimally.

Sample Input	**Output for Sample Input**
5 4 7 1 14 15	14

PROBLEM J **CONTEST SETTING** **30 POINTS**

One of the reasons the NZPC is so popular is that different problems have different point values, depending on their difficulty. There are 3-point problems, 10-point problems, 30-point problems and 100-point problems. But: why are there only 4 different point values? Why not set a contest in which every problem has a different number of points?

You have a set of n problems and want to use k of them in an upcoming contest. You evaluate the difficulty of every problem and assign it an integer point value. Often multiple problems have the same point value. How many distinct contests can you build from the set of n problems? A contest is valid only if all of its k problems have different point values. Two contests are distinct if and only if there exists some problem present in one contest but not present in the other.

Since there might be a large number of possible contests and you don't like big integers, print the result modulo 998,244,353.

Input

The first line of input contains two space-separated integers n and k ($1 \le k \le n \le 1000$).

The next line contains n space-separated integers representing the point values. Point values are between 1 and 10^9 (inclusive).

Output

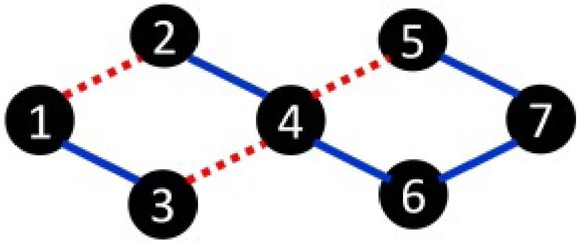
Print the number of distinct contests possible, modulo 998,244,353.

Sample Input 1	Output for Sample Input 1
5 2 1 2 3 4 5	10

Sample Input 2	Output for Sample Input 2
5 2 1 1 1 2 2	6

Sample Input 3	Output for Sample Input 3
12 5 3 1 4 1 5 9 2 6 5 3 5 8	316

PROBLEM K **COLOURING GAME** **30 POINTS**



Alice and Bob are playing a game on a simple connected graph with N nodes and M edges.

Alice colours each edge in the graph red or blue.

A path is a sequence of edges where each pair of consecutive edges have a node in common. If the first edge in the pair is of a different colour than the second edge, then that is a "colour change".

After Alice colours the graph, Bob chooses a path that begins at node 1 and ends at node N. He can choose any path on the graph, but he wants to minimize the number of colour changes in the path. Alice wants to choose an edge colouring to maximize the number of colour changes Bob must make. What is the maximum number of colour changes she can force Bob to make, regardless of which path he chooses?

Input

The first line contains two integer values N and M with $2 \le N \le 100{,}000$ and $1 \le M \le 100{,}000$ The next M lines contain two integers a_i and b_i indicating an undirected edge between nodes a_i and b_i ($1 \le a_i, b_i \le N$, $a_i \ne b_i$).

All edges in the graph are unique.

Output

Output the maximum number of colour changes Alice can force Bob to make on his route from node 1 to node N.

Sample Input 1	Output for Sample Input 1
3 3 1 3 1 2 2 3	0

Sample input/output data continues overleaf …

Sample Input 2	Output for Sample Input 2
7 8 1 2 1 3 2 4 3 4 4 5 4 6 5 7 6 7	3

PROBLEM L **PALINDROMIC ENCODING** **30 POINTS**

When transmitting signals over long distances we need to introduce some redundancy in order to detect errors in transmission.

We have created a new redundant system to detect transmission errors, using palindromic numbers. A palindromic number is one that, when written in base 10 with no leading zeros, reads the same in both directions. For example, 191 and 532235 are both palindromic.

To send a number in this palindromic encoding system you simply find a way to express it as a sum of palindromic numbers. You send each palindromic number as you would normally do and the receiver checks that each received number is palindromic. If not, an error has occurred.

To keep the communication efficient enough, we have added the constraint that a number can only be broken down into the sum of at most 10 palindromic numbers. You must find a way to break any number down into palindromic numbers[1].

Input

One line containing a single integer n ($1 \leq n < 10^{18}$), the number you want to write as a sum of palindromic numbers.

Output

First, output a line with a single number $1 \leq k \leq 10$, the number of palindromic numbers you need. Then follow k lines each containing a palindromic number. The palindromic numbers must sum to n. Where multiple solutions exist, any one will do.

Sample Input 1	Output for Sample Input 1
1100000	2 645546 454454

Sample Input 2	Output for Sample Input 2
1000	5 1 99 1 898 1

[1] A recent paper has shown that every positive integer is a sum of three palindromic numbers.

PROBLEM M **HEIGHT HARMONISATION** **100 POINTS**

Given a row of buildings of the same width but different heights, a developer with a weakness for straight lines would like to equalise the heights of the buildings as much as possible. The height of a building is defined by the number of storeys that it has. The developer would like to know the minimum cost of equalising the heights assuming the buildings may be divided into K groups, where a group consist of adjacent buildings, and all the buildings in one group must have the same height after the work is completed. The cost of modifying a building is equal to the number of storeys added/subtracted to reach the target height.

Input

The first line of input contains two integers, N and K. N is the number of buildings ($1 \leq N \leq 50$) and K is the number of groups ($1 \leq K < N$). The following N lines describe the heights of the buildings, given as integers between 1 and 100 inclusive, with one integer per line.

Output

Output a single integer, the minimum cost of equalising the heights when the buildings may be divided into K groups.

Sample Input

```
10 4
63
53
4
59
68
6
12
47
63
28
```

Output for Sample Input

```
96
```

Explanation

One optimal solution is to change the heights of the first five buildings to 59 (cost: 4 + 6 + 55 + 0 + 9), change the heights of the next two buildings to 9 (cost: 3 + 3), change the heights of the next two buildings to 55 (cost: 8 + 8), and leave the height of the last building as 28.

PROBLEM N **MAX EMPTY** **100 POINTS**

Deep in the heart of the ICPC rainforest, high on a hillside, there is an ancient brick water reservoir. The villagers who live in the valley below the reservoir rely on it to supply water for the dry summer season. Each year, at the end of autumn, the villagers are allowed to remove one brick from the outer wall of the reservoir and take the water that flows out (after the flow of water ends, a new brick is cemented into the hole, leaving the reservoir intact and ready to collect water for the next year). The amount of water taken depends on the choice of brick. Your task is to determine which brick to remove in order to maximize the amount of water extracted, and to report that amount of water.

Figure 1 shows a small and basic reservoir which is 8 bricks long, 8 bricks wide and 4 bricks high. When this reservoir is full, the volume of water stored is 6 * 6 * 3 = 108 units (where one unit is the volume of one brick).

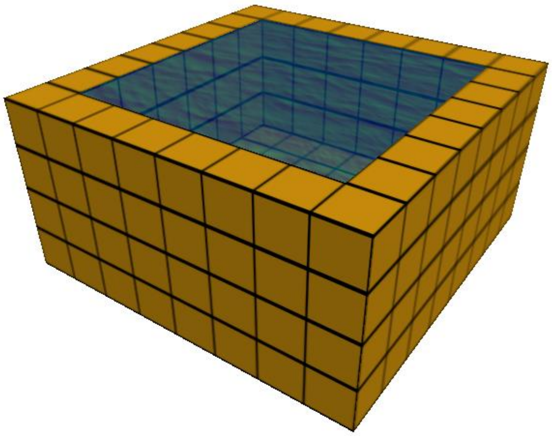


Figure 1: Basic 8×8×4 reservoir

For obscure reasons, over the years additional bricks have been cemented inside the reservoir, building up columns and walls that complicate the internal space. Each of the additional bricks is laid on another brick, in perfect alignment. This means that the internal structure can be described by listing the number of additional bricks stacked on each of the bricks that form the floor of the reservoir. Where bricks are adjacent, either directly or diagonally, the gap between them is fully sealed such that water cannot flow through it.

Figure 2 shows the structure of the reservoir corresponding to the sample input. The reservoir always starts completely full of water. Figure 3 shows the water that remains after the indicated brick is removed (the corner with the indicated brick corresponds to the bottom left corner of the sample input).

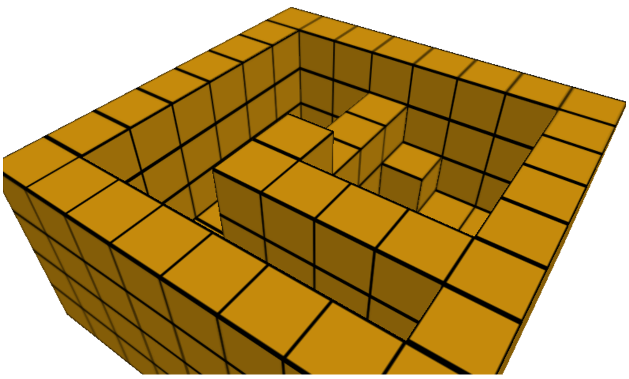


Figure 2: Reservoir corresponding to the sample input (water not shown)

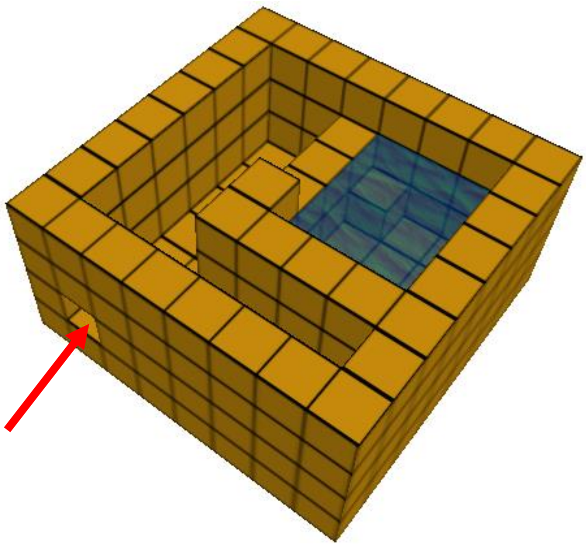


Figure 3: Remaining water after emptying through hole (arrow)

Continued

Input

The initial line of input has three integers: L, W, H; being the length, width and height of the reservoir ($3 \leq L, W, H \leq 500$). The initial line is followed by $W - 2$ more lines. Each of these lines holds $L - 2$ integers, each in the range 0 to $H - 1$ (inclusive), being the number of bricks stacked on the corresponding cell of the reservoir's floor. The outer wall implicitly has height H.

Output

Output a single integer, the maximum volume of water (in brick volumes) that can be extracted by removing a single of brick from the outer wall of the reservoir.

Sample Input

```
8 8 4
0 0 2 1 0 0
0 0 2 0 0 0
0 0 2 0 0 0
0 0 3 0 0 0
0 0 3 3 3 3
0 0 0 0 0 0
```

Output for Sample Input

```
63
```

PROBLEM O **MAIL RUN** **100 POINTS**

Around the main buildings of the ICPC headquarters lies a vast (deep and dark) forest. Fittingly the forest is crisscrossed by a maze of twisty paths connecting a myriad of small clearings. These paths constitute the only way into headquarters from the outside world. The difficulty of navigating this maze has unfortunately made it impossible to get a high-speed internet connection installed and so the organisation is still dependent on that old-fashioned system affectionately known as 'snail mail'. Mail is delivered to and collected from a letterbox at a clearing on the outer edge of the forest. Once each day a programmer in headquarters is chosen to do the 'mail run'. They are required to run from the central headquarters building to the mail box, collect the mail, and run back. It is a matter of honour to do this run as efficiently as possible, in the sense of running the least possible distance. Today, you have been chosen to be the runner and must decide on your route.

There is one final detail you need to know. The forest is inhabited by carnivorous little animals called 'bytes' – 'byte' by name and 'bite' by nature. Although they are small, they are ferocious and meeting one that is awake is invariably fatal for a runner. Fortunately, bytes spend most of their time asleep; but when a runner passes, they wake. It takes a moment for a waking byte to orient itself, so the runner that disturbed it can continue safely without fear of attack. However, the byte will remain awake for some time. It won't move, but it will attack anyone on the path or the clearing it occupies for the remainder of the day. It can be assumed that all bytes are asleep at the time the run starts and it can be assumed that a run is always completed within the day. Therefore, the only safe way to complete a run is to use a route that passes through different clearings and does not use any path twice.

Input

The initial line of input has four space separated integers: C, P, H and M. C is the number of clearings ($2 \leq C \leq 1000$) and P is the number of paths between clearings ($1 \leq P \leq 5000$). Clearings are identified by integers from 1 to C inclusive. H is the number of the headquarters' clearing and M is the number of the mailbox clearing ($H \neq M$). The initial line of input is followed by P more lines – one for each path between clearings. Each of these lines has three integers: u, v and w; denoting that there is a path connecting clearings u and v of length w ($u \neq v$, $1 \leq w \leq 1000$).

Output

Output a single integer, the length of a shortest route that starts from clearing H, passes through clearing M, and returns to clearing H, without traversing the same path more than once, and without visiting any clearing (other than H) more than once. It is possible that no such route exists, in which case output `-1`.

Turn over for sample input and output.

Sample Input

```
4 6 1 4
1 2 2
1 4 12
2 3 2
3 4 2
2 4 7
1 3 6
```

Output for Sample Input

```
17
```

PROBLEM P **SPEED** **100 POINTS**

Speed is a card game in which players place numbered cards in piles subject to certain rules. For the purpose of this problem, there is a single pile and the rule is that the value of each card must be either one higher or one lower than the value of the previous card in the pile.

We obtained the cards from the pile after a (presumed) game, but the order of the cards was lost in the process. We would like to know how many possibilities there are for the first and last cards played in the game.

Input

The first line of input will consist of a single integer N, the biggest number on any of the cards $(1 \leq N \leq 100{,}000)$.

The following line of input will consist of N space-separated integers describing the frequencies of the cards. The first integer is how many cards have the number "1", the second integer is how many cards have the number "2", and so on. Each frequency F_i will satisfy $0 \leq F_i \leq 1{,}000{,}000$, and the sum of F_i will be at least 1.

Output

Output a single integer, the number of distinct pairs (A, B) for which a pile can exist consisting of all the cards described in the input where the first card is numbered A, the last card is numbered B, and the value of each card in the pile is either one higher or one lower than the previous card.

It is possible that the cards do not correspond to a valid game and no valid pile can exist, in which case output 0.

Note: The output might not fit in a 32-bit integer.

Sample Input

```
3
1 2 1
```

Output for Sample Input

```
4
```

Explanation

There is 1 card with the number "1", there are 2 cards with the number "2", and there is 1 card with the number "3".

One of the possibilities for the pile is "2", "1", "2", "3". The corresponding *(first, last)* pair is ("2", "3").

In total, there are 4 possible *(first, last)* pairs: ("1", "2"), ("2", "1"), ("2", "3"), and ("3", "2").