Problem A
Art Installation
Time limit: 1 second

Jolie is setting up an art installation for her cat, Millie. The art installation will be made up of multiple LEDs.

Jolie has decided that she needs a specific number of red LEDs, green LEDs, and blue LEDs. After rummaging through her desk, she has found some of each. She can buy two special types of LEDs, one which can be either red or green, and another which can be either green or blue.

How many of the special LEDs will Jolie need to buy to finish her installation?

Input

The first line of input contains three integers r, g and b ($0 \leq r, g, b \leq 1{,}000$), which are the numbers of red LEDs, green LEDs, and blue LEDs Jolie needs.

The second line contains three integers c_r, c_g and c_b ($0 \leq c_r, c_g, c_b \leq 1{,}000$), which are the numbers of red LEDs, green LEDs, and blue LEDs Jolie already owns.

The third line contains two integers c_{rg} and c_{gb} ($0 \leq c_{rg}, c_{gb} \leq 2{,}000$), which are the numbers of special LEDs that can be either red or green, and the number of special LEDs that can be either green or blue, that are available for Jolie to buy.

Output

Output a single integer, which is the total number of LEDs Jolie needs to buy to make her installation. Output -1 if there aren't enough LEDs for her to complete her installation.

Sample Input 1	Sample Output 1
583 290 710 396 65 796 309 800	412

This page is intentionally left blank.

Problem B
Backup Towers
Time limit: 3 seconds

The nation of Gridlandia can be, naturally, represented as a grid of houses.

Gridlandia is planning to install a number cell towers at some of the points of the grid. When a resident connects to cellular data, their phone will try to connect first to their closest tower, and if that fails, to their second closest tower. The government of Gridlandia is worried that some towers will become much more overloaded than others. Help them by computing for each house both the closest and second closest cell tower, as measured by the Manhattan Distance, which is the difference in rows plus the difference in columns.

Input

The first line of input contains three integers r, c ($1 \leq r, c \leq 500, r \cdot c \geq 2$) and n ($2 \leq n \leq 2 \cdot 10^5$) where Gridlandia is a grid with r rows and c columns, and n is the number of cell towers. The towers are numbered from 1 to n.

Each of the next n lines contains two integers i and j ($1 \leq i \leq r$, $1 \leq j \leq c$) which are the positions of the cell towers, in order. Every (i, j) pair will be unique.

Output

The first r lines of output should each contain c space separated integers. The integer on the i^{th} row and j^{th} column should be the number of the *closest* cell tower to position (i, j), as measured by the Manhattan distance.

The next r lines of output should also each contain c space separated integers. The integer on the i^{th} row and j^{th} column should be the number of the *second closest* cell tower to position (i, j), as measured by the Manhattan distance.

If there are multiple closest cell towers to a given position, output the one with the lowest number. Likewise, if there are multiple second-closest cell towers to a given position, output the one with the lowest number.

Sample Input 1	Sample Output 1
4 8 3 1 1 4 1 2 6	1 1 1 1 3 3 3 3 1 1 1 3 3 3 3 3 2 2 2 3 3 3 3 3 2 2 2 2 3 3 3 3 2 2 3 3 1 1 1 1 2 2 3 1 1 1 1 1 1 1 1 2 2 2 2 2 1 1 1 3 2 2 2 2

Problem C
Call for Problems, Round 2
Time limit: 1 second

The Call for Problems for the ICPC North America Qualifier (NAQ) has finished, and a number of problems were proposed. The judges voted on the difficulty of each problem. The NAQ this year will feature some number of problems. The NAQ wants to feature problems with as many unique difficulties of possible. Compute the maximum number of unique difficulties attainable.

Input

The first line of input contains two integers n and k ($1 \le k \le n \le 50$). NAQ will use exactly k problems out of the n proposed.

Each of the next n lines contains a single integer d ($1 \le d \le 50$). These are the difficulties of the n problems proposed.

Output

Output a single integer, which is the maximum number of unique difficulties that the NAQ can feature.

Sample Input 1	Sample Output 1
20 19 43 4 19 27 34 7 12 34 44 36 38 38 39 34 30 35 44 47 39 5	15

Problem D
Carl's Maze-Solving Algorithm
Time limit: 1 second

Carl the ant is back! After traveling around some pyramids, Carl has decided to study some algorithms and has invented a novel algorithm for solving grid mazes. It works as follows:

- Carl starts somewhere in the maze facing to the right and wants to get to a destination square.
- While Carl is not yet in the destination square.
  - If Carl can turn left by 90 degrees and face an empty square, he will turn left 90 degrees and then move forward by one square.
  - Otherwise, if Carl can move forward by one square, he will do so.
  - Otherwise, he will turn right 90 degrees.

Carl wants to know if this algorithm works. Help him check!

Input

The first line of input contains two integers, r and c $(1 \leq r, c \leq 50)$, indicating the size (rows, columns) of the maze. The cell at $(1, 1)$ is the top left corner of the maze.

The next line of input contains two integers, i_{start} and j_{start} $(1 \leq i_{start} \leq r, 1 \leq j_{start} \leq c)$, the starting location for Carl in row i_{start}, column j_{start}.

The next line of input contains two integers, i_{end} and j_{end} $(1 \leq i_{end} \leq r, 1 \leq j_{end} \leq c)$, the desired ending location for Carl in row i_{end}, column j_{end}. It is guaranteed the starting location and desired ending location for Carl are different.

Each of the next r lines contains a string of c characters, consisting only of `0` or `1`. If the character is `1`, then that square has an obstacle in it and cannot be traversed, otherwise it is empty. It is guaranteed that Carl's starting location and desired ending location are empty.

Output

Output a single integer, which is 1 if it is possible for Carl to get from the starting location to the ending location, and 0 otherwise.

Sample Input 1	Sample Output 1

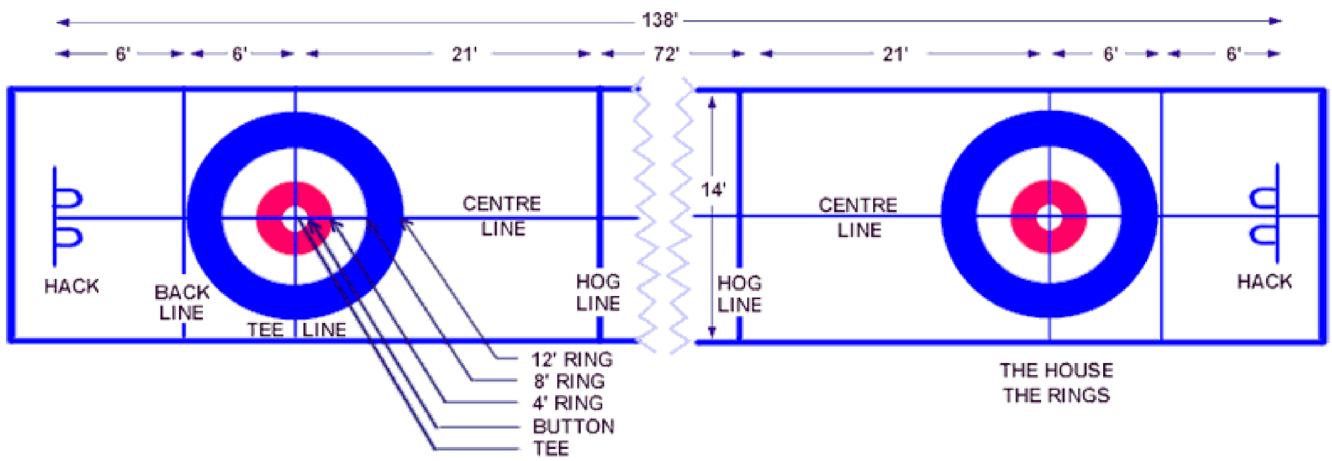
<pre>4 5
1 1
4 5
00111
10100
10111
10000</pre>
	<pre>1</pre>

Sample Input 2	Sample Output 2

<pre>3 3
1 1
3 3
001
001
110</pre>
	<pre>0</pre>

Problem E
Curling
Time limit: 1 second

Curling is a winter sport. Players on two teams alternate sliding eight "stones" each of their color, red or yellow, down an ice sheet, trying to get their stones as close to the "button" as possible. The ice sheet is 138 feet long by 14 feet wide. In inches, that's $1{,}656 \times 168$. The button is 12 feet from the end, in the center, 7 feet from either side. That's at location $(144, 84)$ in inches.



Each game has ten frames. In each frame, the teams alternate sliding their stones down the ice sheet, trying to get their stones closest to the button (or knocking their opponent's stones away from the button). Whichever team has the closest stone to the button gets all the points in a frame. They get one point for every stone of theirs that is closer to the button than any of their opponent's stones. It is possible for stones to completely exit the ice sheet.

Given the positions of the stones, in inches, after each of the ten frames, and assuming that the button is at $(144, 84)$, compute the final score of the game.

Input

The input consists of exactly 20 lines, in 10 pairs. Each pair consists of the positions of the red stones followed by the positions of the yellow stones for one frame.

Each line begins with an integer n ($0 \le n \le 8$), which is the number of stones that remain on the ice sheet, followed by n ordered pairs of integers x ($0 \le x \le 1{,}656$) and y ($0 \le y \le 168$), which are the positions of each stone that remains on the sheet, in inches. It is guaranteed that no two stones in a frame will be at the same distance from the button.

Output

Output two integers on a single line, which are the red team's score followed by the yellow team's score.

Sample Input 1

```
7 1627 63 1506 128 338 123 1035 64 1350 11 275 74 560 53
8 1500 32 1015 96 524 160 1578 160 28 112 1383 147 165 10 1334 167
8 385 75 807 141 903 91 1589 8 1091 68 398 24 1366 95 351 123
6 1003 68 560 149 147 117 779 12 595 166 1176 30
7 755 114 195 131 418 54 902 53 292 136 178 42 495 153
7 1226 109 1111 44 1282 91 624 30 1220 79 357 151 232 140
8 894 81 1519 65 1442 84 1157 3 952 150 1252 158 1005 67 1104 21
4 505 33 608 37 40 75 782 158
2 1002 167 826 9
1 301 138
3 81 85 1156 29 1261 125
6 1266 88 1570 157 1179 126 605 128 1375 14 972 57
8 840 85 327 118 1527 22 964 139 1347 110 1607 68 794 92 1577 6
8 253 129 1256 25 50 67 77 112 1195 84 1473 73 105 153 1410 167
4 358 2 1310 51 950 80 814 58
6 57 142 439 5 329 155 144 155 868 114 410 72
2 862 92 379 116
7 102 88 372 22 1236 53 1608 75 425 47 1574 154 1607 7
4 1150 83 1047 112 725 46 498 166
3 363 35 1014 138 911 53
```

Sample Output 1

```
3 18
```

Problem F
Figure Skating Judging
Time limit: 2 seconds

Anissa has just finished her figure skating routine and is anxiously awaiting her scores. The way scoring works at this specific event is a little peculiar - each figure skater is evaluated by several judges and the scores are averaged to give a final score.

Jolie is managing the software used to compile the scores and notices that a hacker has gotten in and inserted many fake evaluations. She knows exactly how many scores she should expect, so she will manually invalidate evaluations until the number of evaluations matches the number of judges she knows actually submitted evaluations.

The problem is, Jolie doesn't know which evaluation to invalidate. She decides to use the following metric for assessing the *badness* of a collection of scores after invalidating enough to get the number of evaluations to match the number of judges:

1. Compute the arithmetic mean of all the evaluations from the chosen collection.
2. Compute the squared deviation of each evaluation in the chosen collection to the given arithmetic mean. That is, if the mean is μ and a given evaluation is x, the squared deviation is $(\mu - x)^2$.
3. Compute the sum of all squared deviations from step 2. This sum is the *badness*.

Compute the minimum badness over all possible collections of scores that can be obtained.

Input

The first line of input contains two integers, n and k $(1 \leq k < n \leq 5 \cdot 10^5)$, where n is the total number of evaluations and k is the expected number of evaluations.

Each of the next n lines contains a single integer x $(1 \leq x \leq 10^6)$. These are the evaluations.

Output

Output a single number, which is the minimum badness over all possible collections of scores that can be obtained. Answer is correct if it is within absolute or relative error of 10^{-6}.

Sample Input 1

```
4 3
13
13
23
13
```

Sample Output 1

```
0
```

Sample Input 2

```
5 2
1
2
3
4
5
```

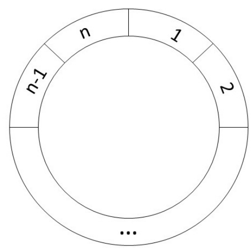
Sample Output 2

```
0.5
```

Problem G
Mingle
Time limit: 1 second

You and your friends are playing the popular childhood game, Mingle.

In the game of Mingle, n players start by standing on a spinning circular platform in the middle of a circular arena. Each player has a unique number ranging from 1 to n, and there are n rooms also with unique numbers from 1 to n arranged on the perimeter of the arena. The rooms are in numerical order, with room n also being adjacent to room 1.



Cheerful music plays for a few seconds, and then the music stops, the circular platform stops spinning, and everyone has to run into a room. Initially, each player tries to target the room with the same number as their number, but because of the spinning, everyone is disoriented. As a result, player i might enter a different room. Notably, the players have a disorientation factor of k, which is the same for all players, and player i might enter a room that is up to k rooms away from their target room. All $2k + 1$ candidate rooms are equally likely for each player and all players select their rooms independently. Every player who ends up alone in a room is a winner in that round of Mingle, even if the room's number is not the same as the player's number.

Compute the expected number of winners in a single round of Mingle.

Input

The first and only line of input contains two integers, n $(3 \leq n \leq 456)$, and k $(1 \leq k \leq \frac{n-1}{2})$, where n is the number of players playing, and k is the disorientation factor of the players.

Output

Let w be the expected number of winners in a single round of Mingle. It can be shown that w can be written as $\frac{a}{b}$ for relatively prime positive integers a and b. Output $ab^{-1} \pmod{998244353}$.

Sample Input 1	Sample Output 1
`3 1`	`332748119`

Problem H
Seesaw
Time limit: 2 seconds

A number of people are sitting on an infinitely long seesaw. The seesaw can be represented as a number line centered at 0. Each person has a weight, and sits at a location along the seesaw. They contribute a torque equal to their weight times their position. The seesaw is balanced if the sum of torques is 0. People are able to move any real-valued distance along the seesaw, so long as they do not go past the person immediately before them or after them. In other words, the relative ordering of the people along the seesaw must be preserved. It is ok for multiple people to occupy the same location, and for a person to move multiple times. What is the minimum sum of distances that people have to move to make the seesaw balanced?

Input

The first line of input contains a single integer n ($1 \leq n \leq 10^5$), which is the number of people.

Each of the next n lines contains two integers p ($-10^8 \leq p \leq 10^8$) and w ($1 \leq w \leq 10^5$), where p is that person's position on the seesaw, and w is that person's weight. The values of p are guaranteed to be unique and in ascending order.

Output

Output a single number, which is the minimum total amount of distance moved for all people in order to balance the seesaw. Answer is correct if it is within absolute or relative error of 10^{-6}.

Sample Input 1	Sample Output 1
3 -3 4 3 1 5 1	1.000000

Sample Input 2	Sample Output 2
3 -2 1 1 4 2 4	2.500000

This page is intentionally left blank.

Problem I
Star Guardians

Jenny and Ojas are part of an elite group of competitive programmers called the Star Guardians. Their next job - to do well in the NAQ!

The Star Guardians are working to assemble a team for the NAQ. They are yet to decide their team, but for each person in the group, it is known how many problems that person will solve.

The Star Guardians are also well-versed in teamwork, so they will solve an additional number of problems based solely on the size of their team.

The Star Guardians want to field a team that maximizes the average number of problems solved per team member. Compute the maximum average they can attain. You may assume the NAQ has infinitely many problems, so the Star Guardians will not run out of problems to solve.

Input

The first line of input contains a single integer, n $(1 \leq n \leq 10)$, the number of Star Guardians.

The next line contains n integers a $(0 \leq a \leq 10^9)$, with the i^{th} integer being the additional number of problems solved if the Star Guardians field a team of size i. It is guaranteed that these integers are monotonically increasing.

The next line contains n integers s $(0 \leq s \leq 10^8)$, which is the number of problems each Star Guardian can solve.

Output

Output a single number, which is the maximum average number of problems per team member attainable. Your output will be considered correct if it has absolute or relative error at most 10^{-6} from the correct answer.

Sample Input 1	Sample Output 1
2 283663485 732616075 0 93763082	413189578.5

Sample Input 2	Sample Output 2
1 17109549 100000000	117109549.0

This page is intentionally left blank.

Problem J
Ten Player Bingo
Time limit: 1 second

Bingo is a game of chance for multiple players. Each player receives a sheet with some numbers, and a game master then calls out these numbers in a random order. Players cross off the numbers that they have heard, and the first player to cross off all their numbers wins the game. This basic version of the game has a reputation for being, well, a bit sedate. No particular action is required of the players except for not falling asleep.

In this problem we will analyze a specific version of Bingo that requires exactly ten players. In our version, called Ten Player Bingo, each player is numbered from 1 to 10. Each player also receives a sheet that contains the integers from 1 to 100 that share the same units digit as the units digit of the player's number. The game master has also been given a sheet that contains every integer from 1 to 100 exactly once, but the integers have been shuffled. The game master will read the integers in the order presented on the sheet. When a player has heard all of their integers, they shout "BINGO" and leave the game. The game continues until everyone has shouted "BINGO".

Who is the last player to shout "BINGO"?

Input

The first and only line of input contains 100 integers, each ranging from 1 to 100. It is guaranteed each integer appears exactly once.

Output

Output a single integer, which is the number of the player who shouts "BINGO" last.

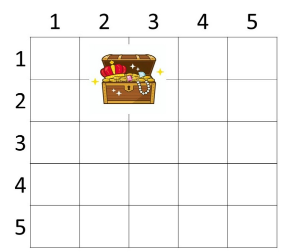
Note that Sample Input 1 as shown has been split into ten lines for display purposes. The actual data will be all on one line.

Sample Input 1	Sample Output 1

`12 91 49 4 52 95 45 51 50 40`
`11 5 88 87 39 38 42 89 76 85`
`30 69 35 81 23 67 32 3 62 8`
`79 58 1 75 37 27 24 46 18 16`
`97 61 41 59 13 74 78 54 98 66`
`14 33 86 55 83 93 63 72 6 19`
`92 17 56 64 100 53 28 71 7`
`96 36 57 84 43 73 60 15 77 80`
`31 20 99 21 90 70 22 26 10 25`
`2 94 9 82 34 65 68 48 29 44 47`
	`7`

Problem K
Treasure Hunt
Time limit: 1 second

There is a 2×2 box of treasure hidden somewhere in a 5×5 grid. The rows are numbered from 1 to 5, as are the columns, with cell $(1, 1)$ being the top left corner of the grid. Your goal is to find the treasure within 5 queries. You can query a grid cell and ask if there is any part of the treasure chest in that grid cell. After at most 5 queries, you must output the upper left coordinate of the treasure box.



Interaction

Your program can make up to 5 queries for the location of the treasure. To make a query, print a new line of the form

? i j

where i and j are integers between 1 and 5. You are asking is the cell at the i^{th} row and j^{th} column has a part of the treasure box. After each query, the interactor will respond with a 0 if the cell does not contain any part of the treasure, or a 1 if the cell does.

After determining the correct answer, output the answer on a new line in the form

! i j

specifying the upper left coordinate of the treasure box. After outputting the answer, your program must exit. If you attempt to read a response after outputting an answer, you will receive an arbitrary verdict.

Do not forget to flush the output after each query you output.

If the interactor receives any invalid or unexpected input, the interactor will output -1 and then immediately terminate. Your program must cleanly exit in order to receive a correct verdict, otherwise the verdict that it receives may be an arbitrary verdict indicating that your submission is incorrect.

A testing tool is provided in python. Use it as follows (You may have to replace `python3` by just `python`):

`python3 testing_tool.py` *`However-you-run-your-program`*

The tool is provided as-is, and you should feel free to make whatever alterations or augmentations you like to it. The tool attempts to detect and report common errors, but it is not guaranteed that a program that passes the testing tool will be accepted.

Sample Interaction 1

Read	Write
	`? 3 3`
`0`	
	`? 2 2`
`1`	
	`? 1 1`
`0`	
	`? 1 3`
`1`	
	`! 1 2`

Problem L
Triple Jump
Time limit: 1 second

You're watching your friend playing a platformer game. In this game, when a player jumps in a straight line, there are three different integer distances they can choose to travel. You don't know what the three are, but hope to find out by watching the gameplay.

You've seen your friend perform many "triple jumps" (i.e. three jumps in a row in the same direction) and recorded the total distance they traveled over all three jumps combined. In each triple jump, your friend can use any of the three integer differences, including repeating the same distance one or more times. After a long time of watching, you've recorded a list of unique distances, and are sure that all possible triple jump distances are in this list.

Given the distances you wrote down, can you recover the three distances?

Input

The first line of input contains a single integer n ($7 \leq n \leq 10$), which is the number of unique observations you made.

The next line contains n space separated integers d ($1 \leq d \leq 1{,}000$). These are your observations, in strictly ascending order.

Output

Output three space-separated integers on a single line. These are the three jump distances. Output them in sorted order, smallest to largest. The three jump distances are guaranteed to be unique.

Sample Input 1	Sample Output 1
10 6 9 10 12 13 14 15 16 17 18	2 5 6

This page is intentionally left blank.

Problem M
Utopia Relationships
Time limit: 1 second

In the Kingdom of Utopia, society has become highly digitized, even in relationships. The governor has built a massive database system of billions of GPUs to record relationships of its residents. Between any two residents, it is required that if they are acquaintances, they have to be registered in the kingdom's database. Note that a registered relationship is mutual: if A is registered to be an acquaintance of B, then B is also registered as an acquaintance of A.

In the last effort to fully digitize relationships, King Aurelius IV proposes "numerical affectionate points". Every resident of the Utopia Kingdom is given $10{,}000$ affectionate points. The residents are then required to distribute their affectionate points to the other residents that have registered their relationship in the database. For example, if A has registered to be an acquaintance to B and C, A can distribute $3{,}000$ affectionate points to B, and $7{,}000$ affectionate points to C. If A is not registered to D, A can not give D any affectionate point. Residents can distribute any integral quantity of affection points, from 0 to $10{,}000$ inclusive.

The King wants to make sure that points distribution is fair and equal: if A gives B x affectionate point, then B must also give A the same x affectionate points. Further, a resident must also distribute all of their affectionate points; the sum of all affectionate points that a resident distributes to their acquaintances must be $10{,}000$.

The King gives you the database of registered relationships, and he wants you to figure out if it is possible for the Kingdom to implement this protocol. You must determine if the king's scheme is possible, and if it is, you must give the King a valid affectionate point distribution.

Input

The first line of input contains two integers n ($2 \leq n \leq 1{,}000$) and m ($1 \leq m \leq 5{,}000$), where n is the number of citizens of Utopia, and m is the number of registered relationships. The citizens are numbered from 1 to n.

Each of the next m lines contains two integers a and b ($1 \leq a, b \leq n, a \neq b$), describing a registered relationship between citizens a and b. All relationships will be unique. If a b appears in the input, then a is an acquaintance of b and b is an acquaintance of a, so b a will *not* appear in the input.

Output

If it is possible to distribute Affection Points as the king requires, then output n lines, each containing n integers. The number at position (i, j) is the number of Affection Points between citizens i and j. Note that along the diagonal where $i = j$, the number of affection points is clearly 0.

If it is not possible to distribute Affection Points as the king requires, simply output -1.

Sample Input 1

```
5 6
1 2
1 3
2 3
3 4
3 5
5 4
```

Sample Output 1

```
0 7500 2500 0 0
7500 0 2500 0 0
2500 2500 0 2500 2500
0 0 2500 0 7500
0 0 2500 7500 0
```

Sample Input 2

```
6 5
1 2
3 1
3 2
4 5
6 5
```

Sample Output 2

```
-1
```