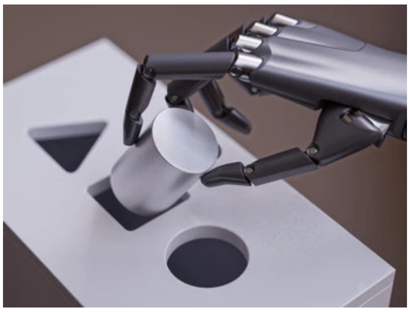
United Kingdom and Ireland Programming Contest 2021

Problems

Problems are not ordered by difficulty.
Do not open before the contest has started.

This page is intentionally left (almost) blank.

Problem A
Arm Coordination



All the cool kids in town want to become a member of the Bots and Androids Programmer Club (BAPC). To become a member of the club, applicants must show a feat of their skills with a home-made robot that is programmed to perform some tricks. Just like your older brother, you want to become a member of the BAPC, so it's time to lock yourself in the hobby basement and start building some robots!

Since your older brother has used up almost all of the parts for his own projects at the BAPC, you will have to get creative with whatever is still left. You find a robotic arm that has only a single purpose: fitting circle-shaped objects into square-shaped holes. Not exactly what you had in mind, but it will have to do. After all, you only have five hours left to apply for your BAPC membership.

The memory chip of the robotic arm seems to be wiped, but luckily you do know the programming interface of its ARM processor. Firstly, the robotic arm only supports integer coordinates. Secondly, when the arm picks up a circle-shaped object, you need to calculate the smallest possible square that it could fit the object in, after which it will autonomically find a suitable square-shaped hole.

Given the location of a circle-shaped object, calculate the smallest possible square which encloses the object.

Input

The input consists of:

- One line containing two integers x and y ($-10^9 \leq x, y \leq 10^9$), the coordinates of the center of the circle.
- One line containing one integer r ($1 \leq r \leq 10^9$), the radius of the circle.

Output

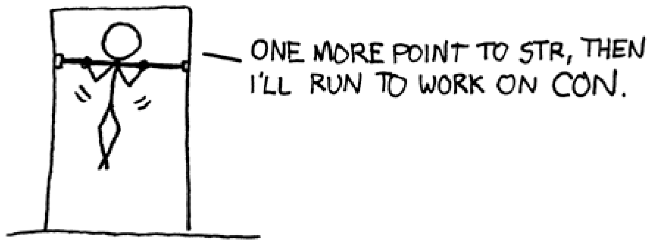
Output four lines, each line containing two integers, representing the x- and y-coordinates of one of the corners of the square. The coordinates should be printed in either clockwise or counter-clockwise order.

If there are multiple valid solutions, you may output any one of them.

Sample Input 1	Sample Output 1
-3 6 5	-10 7 -2 13 4 5 -4 -1

Sample Input 2	Sample Output 2
0 0 10	-14 -2 -2 14 14 2 2 -14

Problem B
BnPC



You are playing your favorite game, Basements and Pigeonlike Creatures, for the umpteenth time. You know the game pretty well, but you have never spent enough time on it to figure out the best strategy. That is, until now. The game consists of a certain sequence of events, such as battling a monster or saving a cat from a tree, and you need to complete all events to win. Attached to each event is an attribute, such as strength, and a threshold, some positive integer. If your attribute score matches or exceeds the threshold, you successfully complete the event! If not, it is unfortunately game over and your total score will be zero.

If you complete all the events successfully, your score depends on how well you did during these events. If your attribute score matches the threshold of an event exactly, you get 0 points, barely scraping by that event. If you exceed the threshold, you get points equal to your attribute score that was used for that event.

You are now at the final part of the game, but first you have some attribute points to spend to increase your attribute scores. You know what events will happen during the final part of the game, so all that is left is to figure out what attributes to increase.

Input

The input consists of:

- One line containing an integer n ($1 \leq n \leq 10^5$) and an integer k ($1 \leq k \leq 10^9$), the number of attributes and the number of attribute points you can still spend.
- n lines, each containing a distinct attribute name, and an integer s ($0 \leq s \leq 10^9$), the current score you have in that attribute.
- One line containing an integer l ($1 \leq l \leq 10^5$), the number of events.
- l lines each describing one event, containing the name of the attribute that is used, and an integer t ($0 \leq t \leq 10^9$), the threshold for this event.

Attribute names consist of upper case English letters (`A-Z`), and have a length between 1 and 20 characters inclusive.

Output

Output the maximum score you can get from the events.

Sample Input 1

```
2 3
STR 15
CON 12
2
STR 17
CON 14
```

Sample Output 1

```
0
```

Sample Input 2

```
3 7
JUMP 5
RUN 7
FLY 0
4
FLY 0
JUMP 6
RUN 10
RUN 8
```

Sample Output 2

```
31
```

Problem C
Cangaroo

Let us talk about the big elephant in the room[1]: you've had a kangaroo in your room for a while now and you need to hide it without raising suspicion, since you want to keep the animal. Hiding an animal of this size is difficult: if you use a lot of space, it is obvious that you are hiding something from your friends. Hence, you want to use as little space as possible to hide the kangaroo.

When the kangaroo was placed against the wall, you took a black and white picture of the animal. Looking around in the house, the only tools you found to hide the kangaroo with were empty tin cans. The dimensions of the tin cans correspond with 2×2 pixels in the picture and these cans cannot overlap. So, you can make a *cangaroo* and if someone asks why you have cans in the shape of a kangaroo, you simply say it is a bad joke of yours.

The position of each can has to exactly correspond to a block of 2×2 pixels in the picture, and they cannot be shifted or rotated to only partially cover some pixels. Furthermore, cans cannot float in the air, so every can has to be supported either by the floor, which is just below the bottom row of the picture, or by another can, for which at least one of the left and right half must directly rest on another can. The structure does not otherwise need to be balanced.

What is the minimum number of cans needed to hide the kangaroo?

Input

The input consists of:

- One line containing two integers n ($2 \leq n \leq 100$) and m ($2 \leq m \leq 10$), the height and width of your room. Both n and m are even.
- n lines, each containing m characters that are either '`.`' or '`#`', where '`#`' marks a position that needs to be hidden by a can.

Output

Output the minimal number of 2×2 cans that is required to hide the kangaroo in the room.

[1] You also wanted an elephant but this did not fit with the kangaroo in your room, sadly.

Sample Input 1

```
4 4
....
..#.
.##.
##..
```

Sample Output 1

```
3
```

Sample Input 2

```
4 4
#.#.
....
#...
....
```

Sample Output 2

```
4
```

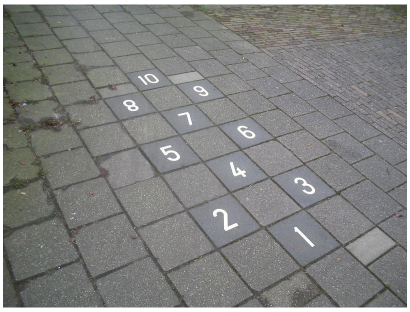
Sample Input 3

```
14 8
........
....##..
...###..
....##..
.....#..
...####.
..#####.
.######.
.#####..
.###....
.##.....
..##....
...#....
.###....
```

Sample Output 3

```
15
```

Problem D
Decelerating Jump



An athlete is participating in a new sport that is the perfect mix of hopscotch and triple jumping. For this jury sport, n squares are laid out on the ground in a line, with equal distances between them. The first phase is the approach, where an athlete sprints towards the first square, in which they start their first jump. Then, they may jump on any number of other squares, and must finally land in the last square.

The jury has given a predetermined number of points to jumping in each square, and the score of the athlete will be the sum of the scores of all the squares they jump in, including the very first and last squares.

Due to the nature of this sport, once the athlete starts jumping, they can not accelerate anymore, and the length of consecutive jumps can never increase. Naturally, it is also impossible to reverse direction.

Given the points the jury has allocated to each square, find the maximal possible score an athlete can get on this event.

Input

The input consists of:

- One line with an integer n ($2 \leq n \leq 3000$), the number of squares.
- One line with n integers $p_1, p_2, \ldots, p_n$ ($-10^9 \leq p_i \leq 10^9$), the number of points the jury awards for jumping on each of the squares.

Output

Output the maximum score that an athlete can get.

Sample Input 1

```
4
1 -1 1 1
```

Sample Output 1

```
3
```

Sample Input 2

```
4
1 1 -1 1
```

Sample Output 2

```
2
```

Sample Input 3

```
3
-1 1 -1
```

Sample Output 3

```
-1
```

Sample Input 4

```
3
-1 -1 -1
```

Sample Output 4

```
-2
```

Problem E
Eerie Shadows

You are walking along home across a bridge one night when you notice the pattern of shadows from the two large lamps at one end of the construction makes an interesting symmetrical pattern on the ground, as the lamps are equally far apart from the sides of the bridge in each direction.

The bridge has pillars running in pairs at irregular sizes and intervals, but always with an identical pillar on the opposite side. You may assume that the parts of the pillars above ground level are infinitesimally thin and that the bridge is infinitely long.

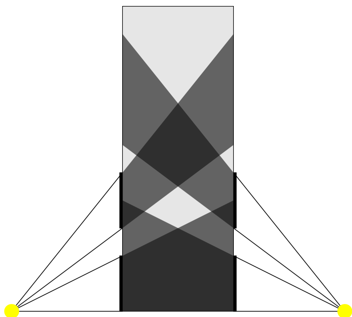


Figure E.1: A visualisation of the second sample input.

How much of the bridge ahead of you is in the shadow of at least one lamp?

Input

The input consists of:

- One line containing two integers w and l ($1 \leq w \leq 500$, $1 \leq l \leq 10000$), the distances in centimetres from the middle of the bridge to either side, and from the sides of the bridge to the lamps respectively.
- One line containing one integer n ($1 \leq n \leq 1000$), the number of pillars.
- n further lines, each containing the integers ab ($0 \leq a \leq b \leq 10000$) giving the distances in centimetres from the end of the bridge to the start and end of one pair of pillars. It holds that $b_i < a_{i+1}$ for all i).

Output

Output the total area in square centimetres that is in shadow from the pillars. Your output must be accurate to an absolute or relative error of at most 10^{-6}.

Sample Input 1	Sample Output 1
1 1 1 0 1	5

Sample Input 2	Sample Output 2
2 4 2 0 2 3 5	32.7

Problem F
Fair Play

Together with your coworker, Larry, you are organizing the exciting Billiards and Pool Competition for your coworkers in your small company. You and Larry are usually on the same page, and surely he will approve of your latest idea. You even bought a nice prize for your coworkers to win and you hope that they are as excited as you are. You want to maximise fun.

It would thus be nice to try to avoid complete walkovers: that is no fun for either player. After some thought, you think it is good to suggest to Larry to divide the players into groups of two. That way, you can compensate for a player's strength by pairing them with a weaker player. In fact, it would be perfect if every team had the exact same strength! Before you tell Larry your plans, you decide to first figure out whether this is possible.

According to your model, synergy plays a negligible role in determining team strength, and the strength of a team is simply determined by the strength of its individual members. Every coworker has a certain skill level in both billiards and pool, as indicated by two integers. When two coworkers are teamed up, their total skill is the sum of their individual skills. Can you divide everyone into teams of two such that every team has the exact same skill in both pool and billiards?

Input

The input consists of:

- A line with an integer n ($2 \leq n \leq 10^5$), the number of coworkers you have.
- Then follow n lines containing two integers b and p ($-10^6 \leq b, p \leq 10^6$), the skill in billiards and pool, respectively, of each coworker.

Output

Output "`possible`" if it is possible to divide all coworkers into teams of two with equal skill. Output "`impossible`" otherwise.

Sample Input 1

```
6
2 1
3 0
3 0
4 2
4 2
5 1
```

Sample Output 1

```
possible
```

Sample Input 2

```
4
1 0
0 1
-2 0
0 -2
```

Sample Output 2

```
impossible
```

Problem G
Getting Square

You have come into possession of an irregularly-shaped piece of a stained-glass mosaic. Although the overall piece is damaged from years of storage, and missing many parts around the border, you think it would be quite nice to keep a small part of this delightful object as a souvenir.

You will find a perfectly-square subregion of this mosaic and prise it out—but only by removing pieces along the existing lines between glass panes, never by cutting into the glass.

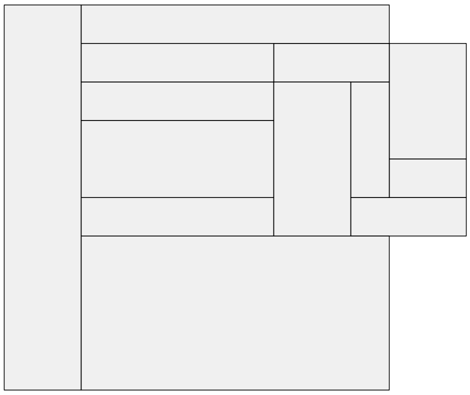


Figure G.1: A visualisation of the first sample input. You can take a 5×5cm piece out from the upper-right corner.

Glass is cumbersome and you'd rather not take more than necessary. What's the smallest square you can take out as one piece without damaging the glass?

Input

The input consists of:

- One line containing the integer n ($1 \leq n \leq 1'000'000$), the number of pieces of glass.
- n further lines, each containing the integers $x_0 y_0 x_1 y_1$ ($0 \leq x, y \leq 1'000'000$), the coordinates in centimetres of the lower-left and upper-right corner of one piece per line.

It is guaranteed that none of the given pieces of glass intersect.

Output

Output the area of the smallest square that you can remove along existing lines, in centimetres squared. If it is not possible to remove any square at all, output `impossible` instead.

Sample Input 1	Sample Output 1
13 20 16 22 19 20 15 22 16 19 14 22 15 10 10 12 20 12 10 20 14 12 14 17 15 19 15 20 18 17 18 20 19 12 19 20 20 12 15 17 17 12 17 17 18 12 18 17 19 17 14 19 18	25

Sample Input 2	Sample Output 2
4 0 0 1 9 1 0 9 1 1 8 9 9 8 1 9 8	impossible

Problem H
Hamiltooonian Hike

Alice loves hiking. She often travels through forests and over mountains for several days, bringing only a backpack. For next year's summer, she decided to travel to a beautiful area which contains a large number of cabins: places where hikers can lay down a sleeping bag and stay for the night. These cabins are connected by hiking trails, paths along the scenery in the area which lead to a next cabin.

Alice's plan is to perform a multi-day hike. Every day, she will walk along the trails to a new cabin to spend the night. She can walk up to three trails in one day—walking four trails is too exhausting. In order to experience as much of the cabins as possible, Alice has decided that she wants to sleep in every cabin at least once. However, the summer has a limited number of days: she does not have the time to visit a cabin multiple times.

Alice has noticed that this requires careful planning of her hike and wonders how to find such a route. Determine which cabin Alice should walk to for every day. Figure H.1 shows a possible route for the second sample case.

Input

The input consists of:

- One line containing two integers n ($2 \leq n \leq 2 \cdot 10^5$) and m ($1 \leq m \leq 2 \cdot 10^5$), the number of cabins and hiking trails.
- m lines each containing two integers x, y ($1 \leq x, y \leq n$, $x \neq y$), indicating that there is a hiking trail between cabins x and y.

It is guaranteed that every cabin is reachable from every other cabin. There is at most one hiking trail between any two cabins.

Output

Output the order in which Alice should visit the n cabins.

You do not need to minimize the total number of hiking trails.

If there are multiple valid solutions, you may output any one of them.

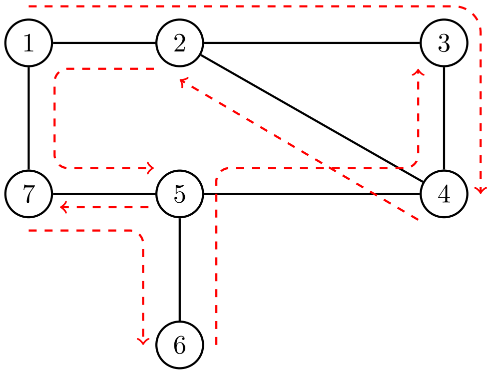
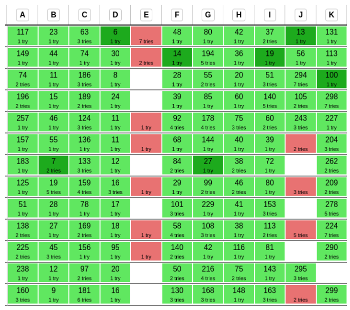


Figure H.1: The input and a possible route (dashed red arrows) for the second sample case.

Sample Input 1	Sample Output 1
4 3 1 2 1 3 1 4	2 1 3 4

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

Problem I
Irregularities



Impressed by the performance of the top teams at the recent BAPC preliminaries, you started to wonder whether teams were allowed to use one or multiple computers to implement their solutions.

Instead of unnecessarily bothering the organization with more questions, you will figure this out by yourself. Being a jury member, you already have estimates for the computer time required to solve each problem.

Using this information, and the time in the contest at which the top team solved each of their solved problems, compute the minimal number of computers used by the team.

The team may work on multiple problems before getting any one of them accepted. Furthermore, the contestants are great multitaskers and can work on a single problem using multiple computers at the same time, but each computer can only be used for one problem at a time.

Input

The input consists of:

- One line containing an integer n ($1 \leq n \leq 10^5$), the number of problems in the contest.
- One line containing n integers $t_1, t_2, \ldots, t_n$ ($1 \leq t_i \leq 10^4$), the computer time required to solve problem i.
- One line containing n integers $s_1, s_2, \ldots, s_n$ ($1 \leq s_i \leq 10^9$ or $s_i = -1$), the time at which problem i was solved, or -1 if it was not solved.

It is guaranteed that the team solved at least one problem.

Output

Output the minimum number of computers used by the team.

Sample Input 1	Sample Output 1
11 50 8 10 6 300 5 6 3 18 5 12 117 23 63 6 -1 48 80 42 37 13 131	1

Sample Input 2	Sample Output 2
1 10 3	4

Sample Input 3	Sample Output 3
2 2 4 3 3	2

Sample Input 4	Sample Output 4
2 4 6 10 10	1

Problem J
Joyride

A group of teenagers has stolen a fast sports car for a Saturday night joyride. The local police department has only one car available to catch the teenagers red handed and put them in a youth detention center.

The city consists of a set of junctions and bidirectional roads, each of a certain length. The teenagers stay at a certain junction until just before the police car arrives at this junction. At that moment, the teenagers want to get to a junction as far as possible from their current location, without using the road the police car is on. They quickly look at a map to determine all junctions within the city which are reachable without using the road with the police car. Then the teenagers determine the distance to each of these junctions using their satnav system and randomly pick one of the furthest located junctions. Note that the satnav system does not know about the location of the police car, and will not take it into account when computing the distance. The sports car then drives instantly to that junction using any route which does not pass by the police car, while the police is left behind dumbfounded. The youngsters will wait there until the police car makes a new approach. The only way for the police to catch the teenagers is by approaching them while they are in a dead end (a junction with only one incoming road). Figure J.1 shows how the police can capture the teenagers in the first sample case.

Since time is precious for the police, they need you to find out if it is possible to catch the joyriders with absolute certainty. And if so, what is the minimal distance they need to drive to be guaranteed to catch the youngsters, assuming the police uses an optimal strategy?

Input

The input consists of:

- One line containing four integers: n ($2 \leq n \leq 300$), the number of junctions, m ($1 \leq m \leq \frac{n(n-1)}{2}$), the number of roads, p ($1 \leq p \leq n$) the initial position of the police car, and t ($1 \leq t \leq n$, $t \neq p$) the initial position of the group of teenagers.
- Then follow m lines, each containing three integers a, b and ℓ ($1 \leq a, b \leq n$, $a \neq b$, and $1 \leq \ell \leq 10^9$), indicating a road between junctions a and b with a length of ℓ.

There is at most one road between every pair of junctions and you can reach any junction from any other junction by making use of the roads.

Output

If it is possible to catch the teenagers with absolute certainty, output the minimal distance that the police car needs to cover to achieve this. Otherwise, output "`impossible`".

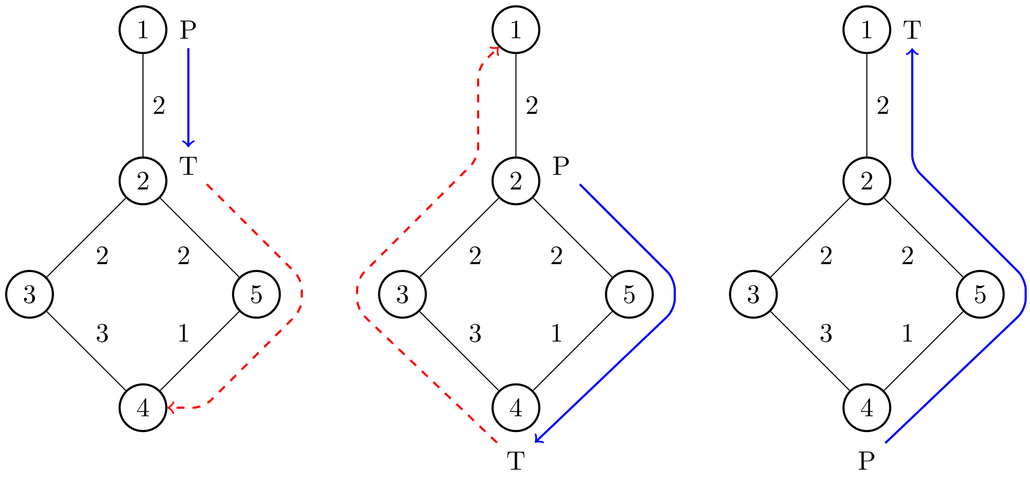
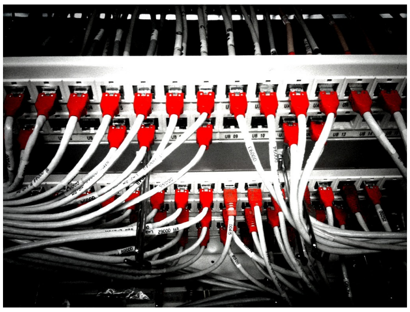


Figure J.1: Possible movements of the police (P) and teenagers (T) in the first sample case. The movement of the police (solid blue arrows) takes time according to the length of the edges, while the movement of the teenagers (dashed red arrows) is instant.

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

Sample Input 2	Sample Output 2
5 5 1 3 1 2 2 2 3 2 3 4 3 4 5 1 2 5 2	impossible

Problem K
Kinking Cables



You need to lay a cable to connect two computers with each other. This cable however has a very specific length and you need to use exactly the full length of the cable. Moreover, the cable cannot intersect itself and one part of the cable cannot be too close to another part. Can you connect the two computers with each other using the full length of the cable?

The two computers are standing in an $n \times m$ rectangular two-dimensional room. Computer 1 is always positioned at $(0, 0)$ (the upper left corner) and Computer 2 at (n, m) (the lower right corner). The cable is specified by a sequence of marked points $p_1, p_2, \ldots, p_s$. The path of the cable is then obtained by connecting the consecutive points of this sequence with (straight) line segments. The cable path should satisfy the following constraints:

- None of the line segments within the cable path should intersect.
- The marked points of the path should not be too close to each other: given a point p_i there should be no other marked points strictly within a radius of 1 of p_i, except possibly p_{i-1} and p_{i+1} (the two consecutive points).
- The path should always start at $(0, 0)$ and end at (n, m).
- All points should lie somewhere in the $n \times m$ room.

Input

The input consists of:

- One line with two integers n and m ($2 \leq n, m \leq 100$), the width and height of the room.
- One line with a floating-point number ℓ ($\sqrt{n^2 + m^2} \leq \ell \leq n \cdot m$), the length that the cable should have.

Output

Output the number of points k ($2 \leq k \leq 500$) that the cable path contains, followed by the k points of the path, in their respective order. Each point consists of two floating-point numbers x and y ($0 \leq x, y \leq 100$), the x- and y-coordinates of this point in the path.

The total length of the path should be exactly ℓ, up to a relative or absolute error of 10^{-6}.

If there are multiple valid solutions, you may output any one of them.

Sample Input 1

```
3 4
5.0
```

Sample Output 1

```
2
0 0
3 4
```

Sample Input 2

```
3 4
7.0
```

Sample Output 2

```
3
0 0
3 0
3 4
```

Sample Input 3

```
5 5
11.5
```

Sample Output 3

```
7
0 0
2 0
2 1.75
4 1.75
4 1
5 1
5 5
```

Problem L
Lopsided Lineup

Together with your coworker, Sergey, you are organizing the exciting Billiards and Pool Competition for your coworkers in your small company. However, communication has not been great between you two. You are not sure you and Sergey think alike, but as far as you are concerned, this would be a great opportunity to do some team building. The actual prizes are meaningless, but there is possibly a lot to be gained from this in team bonding. You want to maximise result.

You start reading some pseudo-scientific books on team management, and after some research, you conclude that there are two good ways of team bonding: people feel more connected after either a triumphant victory or a crushing defeat. This gives you a great idea: if you divide your coworkers into two groups that are as far apart in skill level as possible, both teams will experience improved bonding! You therefore think it is optimal to try to make the teams as unbalanced as possible. Make sure, however, that the teams are of equal size.

With a bit of work you come up with a nice model for the strength of a team. You think team strength is mainly determined by how well two players play together, whether they encourage one another and complement each other's weaknesses. Whenever two players i and j are in the same team, they increase the team score by an integer $c_{i,j}$. The total score of a team is thus equal to the sum of $c_{i,j}$, over all unordered pairs of players i and j in the team.

Input

The input consists of:

- One line with an even integer n ($2 \leq n \leq 1000$), the total number of players.
- n lines, the ith of which contains n integers $c_{i,1}, c_{i,2}, \ldots, c_{i,n}$ ($-10^6 \leq c_{i,j} \leq 10^6$).

  For any i and j, it is guaranteed that $c_{i,i} = 0$ and $c_{i,j} = c_{j,i}$.

Output

Output the maximum possible difference in strength between two teams of equal size.

Sample Input 1

```
4
0 1 2 2
1 0 8 -3
2 8 0 5
2 -3 5 0
```

Sample Output 1

```
6
```

Sample Input 2

```
6
0 4 -6 2 3 -3
4 0 2 -6 0 0
-6 2 0 0 2 2
2 -6 0 0 -1 5
3 0 2 -1 0 -4
-3 0 2 5 -4 0
```

Sample Output 2

```
0
```