max planck institut informatik

Competitive Programming

Saarland University — Summer Semester 2022

Markus Bläser, Karl Bringmann, Martin Bromberger, Christoph Weidenbach
Julian Baldus, Yasmine Briefs, Simon Schwarz, Dominic Zimmer

Assignments Week 3

Deadline: **May 5, 2022 at 16:00 sharp**

Please submit solutions to the problems in our judge system, available at
`https://compro.mpi-inf.mpg.de/`.
You can find your credentials on your personal status page in our CMS.

Problem	`begging`	`cashier`	`cake`		`warming`		
Points	3	3	2	1	2	1	0
Difficulty	🌶	🌶	🌶	🌶🌶	🌶	🌶	🌶🌶
Time Limit	1s	3s	6s	6s	1s	1s	1s
Memory Limit	2 GB	2 GB	2 GB	2 GB	2 GB	2 GB	2 GB

Please note:

- Your solution will be judged immediately after submitting. This may take some time, depending on the current server load.
- You can submit as many times as you want. However, don't abuse the server or try to extract the secret test cases.
- If your solution is **accepted**, you will receive the points specified in the table above.
- If you get **another verdict**, you will receive 0 points.

The Big Begging

Problem ID: begging
Time limit: 1 second

The wine party was a blast, but Dieter seriously underestimated his parents. Unsurprisingly, they found out that Dieter 'borrowed' some of their wine. Quite possibly because he returned different bottles than he took.

They are furious at him and want him to have fixed the damage once they are back in town in n days. He needs to replace all the cheap booze he brought with the high quality wine his parents love so much. But he is a student, and what do students have less than bottles of wine? Money to buy such bottles.

Since he is quite popular among his new friends, he might ask them for a little help. They surely have the wine required to propitiate his parents, but Dieter wants to avoid coming off as that one begging friend. Dieter is convinced that it is socially acceptable to borrow some bottles of wine from his friends every m days, but it might not hurt to wait even longer. In the meantime, he will spend some time with his friends and strengthen their friendship.

As we all know, wine is a volatile resource. Some days it might be there, next day it could be gone. Dieter knows exactly when it would be a good day to ask his friends for some wine. He even knows how many bottles (of the good stuff) he would get on a certain day.

Can you help Dieter get the best reimbursement for his parents possible? Figure out, when he should ask his friends about a 'gift to prove their friendship'.

Input

In the first line of the input you are given the integers n and m. Dieters parents will return in $1 \leq n \leq 300000$ days and Dieter found that it would be a good idea to wait for $1 \leq m \leq 10$ days after bumming his friends. Lastly, there is one line with n integers $x_1, \ldots x_n$ ($0 \leq x_i \leq 32$) where x_i describes the number of bottles he can get from his friends on the i-th day. *Note that Dieter can wait longer than m days between begging for wine. If Dieter asked his friends for wine on the i-th day, he will not ask them again until the $(i + m)$-th day.*

Output

Dieter wants you to send him the best begging strategy in two lines of a single Jabber message.

In the first line, state the number of bottles Dieter can get from his friends, as well as the number of days d on which he needs to ask them to get said bottles. In the second line, state the date of each of those d days in increasing order. If there are multiple optimal solutions, you may send Dieter any of them.

Sample Input 1

```
4 2
1 1 1 1
```

Sample Output 1

```
2 2
1 4
```

Sample Input 2

```
9 3
0 1 2 2 3 2 3 2 1
```

Sample Output 2

```
6 3
2 5 8
```

Sample Input 3

```
12 3
4 8 0 9 4 9 8 4 4 8 7 0
```

Sample Output 3

```
29 4
1 4 7 10
```

Careful Cashier

Problem ID: cashier
Time limit: 3 seconds

Und lassen Sie mich auch hier Dank aussprechen an Menschen, denen zu selten gedankt wird. Wer in diesen Tagen an einer Supermarktkasse sitzt oder Regale befüllt, der macht einen der schwersten Jobs, die es zurzeit gibt. Danke, dass Sie da sind für ihre Mitbürger und buchstäblich den Laden am Laufen halten.

— Angela Merkel, März 2020

And let me at this point thank people, who are thanked too rarely. Who in these days is sitting at a cash desk or refills shelves is taking care of one of the hardest jobs right now. Thank you for being there for your fellow citizens and literally keeping the store running.

— Angela Merkel, March 2020

After disinfecting your hands for the fifth time today, you return to checkout 3. The second half of the queue from the open checkout immediately rushes into your direction after you yell "Kasse 3 ist offen". For a short moment, you are happy about the glass pane separating you from the next customer, as he didn't cover his nose with his facial protection. However, after saying "6.21€ bitte" you see him taking out his wallet and digging for the right coins, which makes you become a little bit mad inside. He makes some incomprehensible noises and gives you a 10€ note. While staring at him, you wonder: What is the least number of coins you need to touch and get out of the cash register, such that they sum up to the required change of 3.79€?

Input

The first line contains two integers C and N. $0 \leq C \leq 100\,000$ is the change you need to return to the customer (in cents) and $1 \leq N \leq 500$ is the number of distinct types of coins there are in the currency. The second line contains N integers $v_1, \ldots, v_N$, where $1 \leq v_i \leq 100\,000$ is the value of the i-th coin type (in cents). It is guaranteed that these values are distinct.

Output

If there is no possibility of choosing coins whose values sum up to C, print `impossible`. Otherwise, print one number: The least number of coins you need to get from the cash register in such a way that their values sum up to C. You can assume that there are infinitely many coins of every type in the cash register.

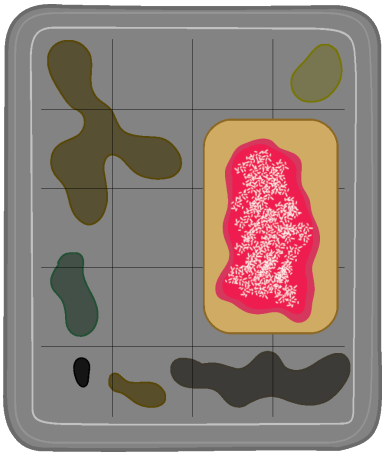
Sample Input 1	Sample Output 1
379 8 1 2 5 10 20 50 100 200	7

Sample Input 2	Sample Output 2
3 4 2 5 10 20	impossible

Sample Input 3	Sample Output 3
6 2 4 3	2

Baking Cake

Problem ID: cake
Time limit: 6 seconds



It was your fault. Not even you can try to pretend it was not. You showed up late to the meeting. Now, your group expects you to bring cake to the next meeting.

You are well prepared: plenty of dough is ready, the oven is on. All you need to do now is to properly lay out the dough of your sheet cake. Unfortunately, the only baking tray available to you is the shared one of your dormitory. Unfortunately, because not every resident is so keen on cleaning up after themselves like you are.

The tray is a rectangle of size of $h \times w$ millimeters. For each square millimeter of the tray, you made up your mind whether you deem it clean enough to use for baking. In order to save work, you only want to bake a single rectangular piece of sheet cake. Where should you place the cake on the baking tray such that you don't cover any of the gross spots?

Depiction of Sample Input 1 and 3

Subtasks

This problem contains subtasks. The solution to one subtask is usually an improvement over or adaption of the solution to a previous subtask. You should therefore think of subtasks as hints that guide you in finding a correct solution.

Due to constraints of the judge system, each subtask appears as a separate problem on the scoreboard and you have to submit your program to each of them.

- **Subtask 1** (2 points) You have already decided on the dimensions H and W of the cake. Find a clean area of the required size on the baking tray.
- **Subtask 2** (1 point) You noticed that you have dough left over and want to make the cake wider. What is the largest clean area of height H on the tray?

Input

The first line contains two numbers h and w ($1 \leq h, w \leq 100\,000$): The dimension of your baking tray in millimeters. The tray is at most $15\,000\,000$ square millimeters large, i.e. $1 \leq h \cdot w \leq 15\,000\,000$.

The next h lines each contain w numbers x_{ij} ($0 \leq x_{ij} \leq 1$) each. x_{ij} is 0 if and only if the (i, j)-th square millimeter is clean enough to be used.

The last line of the input contains a description of the dimensions of the cake you will bake. For subtask 1, these are two numbers H and W ($1 \leq H \leq h$, $1 \leq W \leq w$), the height and width of the cake. For subtask 2, this is just one number H, the height of the cake.

Output

Determine a possible position for the cake and print the **row and column** of the coordinates of its upper left and lower right corner. For subtask 1, the cake must have exactly the given dimensions. For subtask 2, the height must be as given and the width must be positive and as large as possible. If no clean area of the required size exists, print `-1 -1 -1 -1`.

Warning

This problem has very large input. If you use `cin`, call `ios::sync_with_stdio(false);` at the start of your program.

Sample Inputs

Note that among the five sample inputs below, the first two correspond to subtask 1 and the last three correspond to subtask 2.

Sample Input 1

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

Sample Output 1

```
2 3 4 4
```

Sample Input 2

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

Sample Output 2

```
-1 -1 -1 -1
```

Sample Input 3

```
5 4
1 0 0 1
1 1 0 0
1 0 0 0
1 0 0 0
1 1 1 1
3
```

Sample Output 3

```
2 3 4 4
```

Sample Input 4

```
6 5
0 0 0 0 0
1 0 0 0 1
0 0 0 0 0
0 0 0 0 0
0 1 0 0 0
0 0 0 0 0
3
```

Sample Output 4

```
1 2 3 4
```

Sample Input 5

```
4 4
1 1 1 1
1 0 0 1
1 0 0 1
1 1 1 1
3
```

Sample Output 5

```
-1 -1 -1 -1
```

Global Warming

Problem ID: warming
Time limit: 1 second

Global warming is an important issue and Jessica knows about it. She decided to make an analysis of historical temperatures and to find a subsequence of days (not necessarily consecutive) where the temperature was strictly increasing. It will convince the non-believers!

Jessica has found historical data from n consecutive days. The temperature on the i-th day was t_i. Formally, we are interested in finding the length of the longest increasing subsequence (LIS) of $(t_1, t_2, \ldots, t_n)$.

Only relevant for subtask 3: Jessica wants to find a really long subsequence and that is why she decided to cheat a bit. She will first choose a position $1 \leq i \leq n$ and then increase the temperature of the days $i, i+1, \ldots, n$ by exactly x. A small change like that will probably not be noticed by the community, while at the same time it can make the LIS longer. What is the largest possible length of the LIS after the change?

Subtasks

This problem contains subtasks. The solution to one subtask is usually an improvement over or adaption of the solution to a previous subtask. You should therefore think of subtasks as hints that guide you in finding a correct solution.

Due to constraints of the judge system, each subtask appears as a separate problem on the scoreboard and you have to submit your program to each of them.

- **Subtask 1**  (2 points) $1 \leq n \leq 5\,000$, and Jessica decides not to cheat, i.e. $x = 0$.
- **Subtask 2**  (1 point) $1 \leq n \leq 200\,000$, and Jessica decides not to cheat, i.e. $x = 0$.
- **Subtask 3**  (0 points) $1 \leq n \leq 200\,000$, and Jessica may cheat: $0 \leq x \leq 10^9$.

Input

The first line of the standard input contains two space-separated integers n and x, the number of days and the difference by which some of the temperatures will be increased. The second line contains n integers $t_1, t_2, \ldots, t_n$ ($1 \leq t_i \leq 10^9$) separated by spaces, the sequence of historical temperatures.

Output

Print one integer, the largest possible length of the LIS after the change.

Sample inputs

The first two sample inputs can occur in any subtask. The last sample input is only a valid input for subtask 3.

Explanation of Sample Input 3

If Jessica picks $i = 4$, the resulting sequence is `7 3 5 22 12 17 13 14`. Its longest increasing subsequence is `3 5 12 13 14`, which has length 5.

Sample Input 1	Sample Output 1
`6 0` `1 5 1 2 2 4`	`3`

Sample Input 2

```
11 0
12 1 3 4 8 22 17 8 1 20 14
```

Sample Output 2

```
6
```

Sample Input 3

```
8 10
7 3 5 12 2 7 3 4
```

Sample Output 3

```
5
```