CS F211
Data Structures and Algorithms
Assignment - 9

Allowed languages: **C**

March 31, 2021

General Tips

- Try to use functions as much as possible in your code. Functions increase reusability and the pass-by-value feature provides a significant help sometimes. Modularizing your code also helps you to debug efficiently.
- Use `scanf` to read characters/strings from STDIN. Avoid using `getchar`, `getc` or `gets`. Try to read up about character suppression in `scanf` as it will be very helpful in some of the problems.
- Use `printf` instead of `putc`, `putchar` or `puts` to print character/string output on STDOUT.
- Indent your code appropriately and use proper variable names. These increase readability and writability of the code. Also, use comments wherever necessary.
- Use a proper IDEs like Sublime Text or VSCode as they help to run and test your code on multiple test-cases easily. You can install Windows Subsystem Linux (WSL) or MinGW 7.3.0, if you are Windows user to compile and run your programs. Alternatively, you can run and test your codes on Online GDB. If you are using WSL or Linux to run your programs, make sure that the gcc version is `gcc 5.4.1 c99`.

A: Erwin's Gambling Addiction

In a much more mellow world, Erwin's gambling tendencies have got him in quite a pinch and he needs your help to win his money back. He will give you a number N and a six sided unbiased die. You have at most N rolls. You have to tell him how many distinct sequences of rolls exist such that the sum of the numbers on each roll adds up to N.
For example if $N = 3$:

1. $1 + 1 + 1$
2. $2 + 1$
3. $1 + 2$
4. 3

Hence there are 4 ways. Since the answer can be very large, print the answer modulo $10^9 + 7$

Input

The first and only line contains a single integer N ($1 \leq N \leq 10^5$).

Output

Print a single integer, the number of ways to get a sum N modulo $10^9 + 7$

```
input
3

output
4

explanation
The explanation is given in the question
```

B: Another Short Question

You are given a number N in the base 10 format. In one operation, you can choose any digit that appears in the number N, and subtract it from N. Find the minimum number of operations required to make N equal to 0.

Input

The first and only line contains a single integer N ($1 \leq N \leq 10^5$).

Output

Print a single integer, the minimum number of operations required to make N equal to 0.

---

input
27

output
5

explanation
The operations would go as follows: $27 \to 20 \to 18 \to 10 \to 9 \to 0$

C: Go D.

Usopp has met another overpowered enemy and is running away again. The layout of the castle he's in is an $N \times N$ grid consisting of empty cells and blocked cells. He is currently at the top left corner of the grid and he has to make his way down to the bottom right corner by only walking on empty cells and by performing two moves, either *right* or *down*. However he must not move to any blocked cell on the grid. Empty cells are represented by '.' and blocked cells are represented by '*'. Find the number of distinct ways to reach the bottom right corner by using only those two moves. Since the number can be large, output the answer modulo $10^9 + 7$

Input

The first line contains a single integer N ($1 \leq N \leq 10^3$), the number of rows and columns in the grid. The next N lines contain N characters each, describing the board.

Output

Output a single integer representing the number of distinct paths to get to the bottom right corner.

---

input

```
4
....
.*..
...*
*...
```

output

```
3
```

explanation

The three different sequences of moves are given as follows: $(D \to D \to R \to D \to R \to R)$, $(D \to D \to R \to R \to D \to R)$, $(R \to R \to D \to D \to D \to R)$.

D: Knapsack

You have a Knapsack with you that can handle a maximum weight of W. You also have N items, the i^{th} integer having a weight w_i and cost c_i. You now have to put items in the knapsack such that the total weight of all the items does not exceed W and the total cost is maximised. Note that you can only use each item at most once.

Input

The first line contains two integers N and W ($1 \leq N \leq 100$, $1 \leq W \leq 10^4$). The next line contains N integers representing the weight of each element, w_i ($1 \leq w_i \leq W$). The third line also contains N integers, representing the cost of each element, c_i ($1 \leq c_i \leq 10^3$).

Output

Print a single integer representing the maximum cost possible satisfying the above conditions.

---

input

```
3 8
3 4 5
30 40 60
```

output

```
90
```

explanation

If you take the first and third element, you will get a total cost of $60 + 30 = 90$ which is the maximum possible without exceeding the weight limit of 8.

E: A Useless Timeskip

Since Naruto only knows rasengan variations, he decided to learn some new jutsu to increase his power. He decided to learn three new jutsu, A,B, and C. Each day he chooses one of the jutsu and trains only that jutsu. Since he gets bored easily, he cannot choose the same jutsu on two consecutive days. One thing about this training is that it does not increase his power level uniformly. More specifically, on the i^{th} day, training the jutsu A increases his power level by a_i, jutsu B increases his power level by b_i, jutsu C increases his power level by c_i. He only has N days to train so he wants you to help him find out what his maximum power level can be after N days of training according to the above restrictions. Assume his inital power level before beginning training is 0.

Input

The first line contains one integer N ($1 \leq N \leq 10^5$). The second line contains N integers representing the increase in power level if he trains jutsu A on the i^{th} day, a_i ($1 \leq a_i \leq 10^4$). The third line contains N integers representing the increase in power level if he trains jutsu B on the i^{th} day, b_i ($1 \leq b_i \leq 10^4$). The fourth and last line contains N integers representing the increase in power level if he trains jutsu C on the i^{th} day, c_i ($1 \leq c_i \leq 10^4$).

Output

Print a single integer, the maximum possible power level after training for N days.

---

input

```
3
10 20 30
40 50 60
70 80 90
```

output

```
210
```

explanation

If he trains C on the first day, B on the second day and C again on the third day, his overall power level will be 210 which is the highest possible in this case.

F: Tree Overkill

You are given an k-ary (k is unknown to you) tree and two vertices. You simply have to output the distance between these two vertices. There is one catch. You have to store the trees using pointers and Left-Child-Right-Sibling notation. The definition for a node should look like this or something similar:

```
struct Node {
        int val;
        struct Node *parent
        struct Node *left_child;
        struct Node *next_sibling;
};
```

It is guaranteed that the root node is node 1.

Input

The first line contains three integers N, u, v ($1 \leq u, v \leq N \leq 10^3$), the number of vertices in the tree. The next $N-1$ lines contain two space separated integers, x and y where y is a child of x.

Output

output a single integer, the distance between the vertices u and v.

---

input
4 3 4
1 2
2 3
1 4

output
3

explanation
If you draw the graph, you can see that the distance between vertices 3 and 4 is 3

G. Longest Common Subsequence

You are given two strings s and t. You have to find the length of the longest subsequence that is present in both s and t.
Note that a subsequence of a string x is the string obtained by removing zero or more characters from x and concatenating the remaining characters without changing the order. For example, "*rishiplayingdressup*" and "*geethplayingdress*" are both subsequences of the string "*rishiandgeethplayingdressup*", but the string "*geethandrishidress*" is not because the order of the letters is not maintained.

Input

The first line contains two numbers N and M, the length of strings s and t respectively ($1 \leq N, M \leq 10^3$). The second and third lines contain a single string each, consisting of only lower case English letters.

Output

Print a single integer representing the length of the longest common subsequence in the two strings.

---

input
4 5
axyb
abyxb

output
3

explanation
Two subsequences that have a length 3 in this case are axb and ayb which are the longest possible.

H: Nikee

You just bought a new pair of special shoes that allow you to jump abnormal heights. You want to test this out so you go to a street in the city that has N buildings all adjacent to each other (meaning there's no space in between buildings). You want to get from the leftmost building to the rightmost building by jumping across the tops of the buildings. You start from the leftmost building and at any point in time, if you are currently on the i^{th} building, you can jump to either the $i+1^{th}$ building or the $i+2^{th}$ building (as long as $i+1$ and $i+2$ are less than N). The company that made the shoes is very greedy so they charge you money through paytm for every jump. More specifically, if the difference in heights between your initial position and final position is h, then you have to pay an amount of h. Find the minimum cost required to reach the last building. In summary, from position i you can go to position $i+1$ or $i+2$ and the cost required to get there is the absolute difference in heights between the current position and the position you go to.

Input

The first line contains the integer N ($1 \leq N \leq 10^5$), the number of buildings. The next line contains N integers, h_i being the height of the i^{th} building ($1 \leq h_i \leq 10^4$)

Output

Output one integer, the minimum cost possible.

```
input
6
30 10 60 10 60 50

output
40

explanation
The optimal path to take here would be the indices 1 → 3 → 5 → 6 where the costs
would be |30 − 60| + |60 − 60| + |60 − 50| = 40
```

I: Coin Minimization

You live in a country where there are N types of coins, each worth a different value. The values of the coins are known to you. You need to pay someone only in coins so you want to make a total of x using only these coins. Note that you can use multiple coins of the same type. If it is not possible to get the desired sum, print -1.

Input

The first line contains two integers N and x ($1 \leq N \leq 100$, $1 \leq x \leq 10^4$), the number of types of coins and the sum you need to achieve respectively. The second line contains N integers, c_i representing the value of the i^{th} coin ($1 \leq c_i \leq 100$).

Output

Output one integer the minimum number of coins required or -1 if it is not possible to get the sum x.

---

input

3 11

1 5 7

output

3

explanation

here an optimal solution would be to take $5+5+1$. You could take $7+1+1+1+1$ $5+1+1+1+1+1+1$ or something similar but that would not be the minimum number of coins possible.

J. Finally Some Non-Pointer Trees

You are given a tree with N nodes. You have to find the length of the longest path in the tree. If you are using dfs or bfs to solve this, you may only traverse the tree once. (If you google this question you'll mostly get an answer with 2 dfs's/bfs's). Note that you don't have to use pointers in this problem.

Input

The first line contains N ($1 \leq N \leq 10^5$), the number of nodes in the tree. The next $N-1$ lines contain two integers each, u and v where there is an undirected edge between the node u and the node v.

Output

Print a single integer, the length of the longest path in the tree.

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

output
3

explanation
In this graph there are 2 paths with length 3, 1 → 2 → 4 → 5 and 3 → 2 → 4 → 5
```
