Drawing Book

Brie's Drawing teacher asks her class to open their n-page book to page number p. Brie can either start turning pages from the front of the book (i.e., page number 1) or from the back of the book (i.e., page number n), and she always turns pages one-by-one (as opposed to skipping through multiple pages at once). When she opens the book, page 1 is always on the right side:

Each page in the book has two sides, front and back, except for the last page which may only have a front side depending on the total number of pages of the book (see the *Explanation* sections below for additional diagrams).

Given n and p, find and print the minimum number of pages Brie must turn in order to arrive at page p.

Input Format

The first line contains an integer, n, denoting the number of pages in the book.
The second line contains an integer, p, denoting the page that Brie's teacher wants her to turn to.

Constraints

- $1 \leq n \leq 10^5$
- $1 \leq p \leq n$

Output Format

Print an integer denoting the minimum number of pages Brie must turn to get to page p.

Sample Input 0

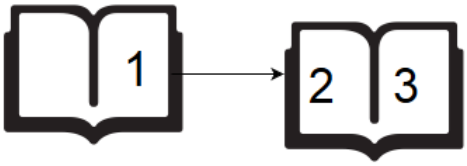
```
6
2
```

Sample Output 0

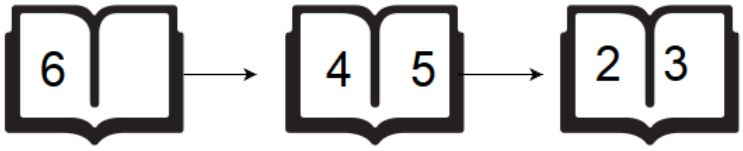
```
1
```

Explanation 0

If Brie starts turning from page 1, she only needs to turn 1 page:



If Brie starts turning from page 6, she needs to turn 2 pages:



Because we want to print the minumum number of page turns, we print 1 as our answer.

Sample Input 1

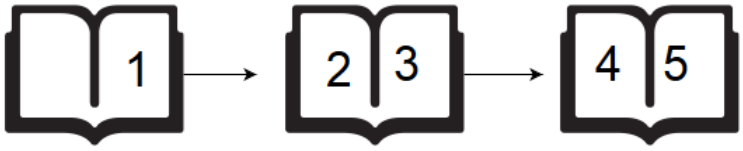
```
5
4
```

Sample Output 1

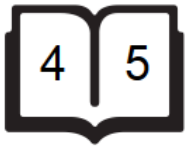
```
0
```

Explanation 1

If Brie starts turning from page 1, she needs to turn 2 pages:



If Brie starts turning from page 5, she doesn't need to turn any pages:



Because we want to print the minimum number of page turns, we print 0 as our answer.

Counting Valleys

Gary is an avid hiker. He tracks his hikes meticulously, paying close attention to small details like topography. During his last hike, he took exactly n steps. For every step he took, he noted if it was an *uphill* or a *downhill* step. Gary's hikes start and end at sea level. We define the following terms:

- A *mountain* is a non-empty sequence of consecutive steps *above* sea level, starting with a step *up* from sea level and ending with a step *down* to sea level.
- A *valley* is a non-empty sequence of consecutive steps *below* sea level, starting with a step *down* from sea level and ending with a step *up* to sea level.

Given Gary's sequence of *up* and *down* steps during his last hike, find and print the number of *valleys* he walked through.

Input Format

The first line contains an integer, n, denoting the number of steps in Gary's hike.
The second line contains a single string of n characters. Each character is $\in \{U, D\}$ (where `U` indicates a step *up* and `D` indicates a step *down*), and the i^{th} character in the string describes Gary's i^{th} step during the hike.

Constraints

- $2 \leq N \leq 10^6$

Output Format

Print a single integer denoting the number of valleys Gary walked through during his hike.

Sample Input

```
8
UDDDUDUU
```

Sample Output

```
1
```

Explanation

If we represent `_` as sea level, a step up as `/`, and a step down as `\`, Gary's hike can be drawn as:

```
_/\    _
  \   /
   \\/
```

It's clear that there is only one valley there, so we print 1 on a new line.

Electronics Shop

Monica wants to buy exactly one keyboard and one USB drive from her favorite electronics store. The store sells n different brands of keyboards and m different brands of USB drives. Monica has exactly s dollars to spend, and she wants to spend as much of it as possible (i.e., the total cost of her purchase must be maximal).

Given the price lists for the store's keyboards and USB drives, find and print the amount money Monica will spend. If she doesn't have enough money to buy one keyboard *and* one USB drive, print `-1` instead.

Note: She will never buy more than one keyboard and one USB drive even if she has the leftover money to do so.

Input Format

The first line contains three space-separated integers describing the respective values of s (the amount of money Monica has), n (the number of keyboard brands) and m (the number of USB drive brands).
The second line contains n space-separated integers denoting the prices of each keyboard brand.
The third line contains m space-separated integers denoting the prices of each USB drive brand.

Constraints

- $1 \leq n, m \leq 1000$
- $1 \leq s \leq 10^6$
- The price of each item is in the inclusive range $[1, 10^6]$.

Output Format

Print a single integer denoting the amount of money Monica will spend. If she doesn't have enough money to buy one keyboard *and* one USB drive, print `-1` instead.

Sample Input 0

```
10 2 3
3 1
5 2 8
```

Sample Output 0

```
9
```

Explanation 0

She can buy the 2^{nd} keyboard and the 3^{rd} USB drive for a total cost of $8 + 1 = 9$.

Sample Input 1

```
5 1 1
4
5
```

Sample Output 1

```
-1
```

Explanation 1

There is no way to buy one keyboard and one USB drive because $4 + 5 > 5$, so we print -1.

Cats and a Mouse

Two cats and a mouse are at various positions on a line. You will be given their starting positions. Your task is to determine which cat will reach the mouse first, assuming the mouse doesn't move and the cats travel at equal speed. If the cats arrive at the same time, the mouse will be allowed to move and it will escape while they fight.

You are given q queries in the form of x, y, and z representing the respective positions for cats A and B, and for mouse C. Complete the function $catAndMouse$ to return the appropriate answer to each query, which will be printed on a new line.

- If cat A catches the mouse first, print `Cat A`.
- If cat B catches the mouse first, print `Cat B`.
- If both cats reach the mouse at the same time, print `Mouse C` as the two cats fight and mouse escapes.

Input Format

The first line contains a single integer, q, denoting the number of queries.
Each of the q subsequent lines contains three space-separated integers describing the respective values of x (cat A's location), y (cat B's location), and z (mouse C's location).

Constraints

- $1 \leq q \leq 100$
- $1 \leq x, y, z \leq 100$

Output Format

For each query, return `Cat A` if cat A catches the mouse first, `Cat B` if cat B catches the mouse first, or `Mouse C` if the mouse escapes.

Sample Input 0

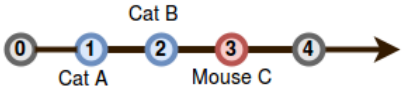
```
2
1 2 3
1 3 2
```

Sample Output 0

```
Cat B
Mouse C
```

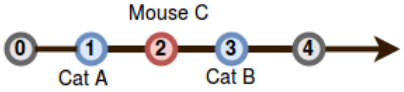
Explanation 0

Query 0: The positions of the cats and mouse are shown below:



Cat B will catch the mouse first, so we print `Cat B` on a new line.

Query 1: In this query, cats A and B reach mouse C at the exact same time:



Because the mouse escapes, we print `Mouse C` on a new line.

Forming a Magic Square

We define a magic square to be an $n \times n$ matrix of distinct positive integers from 1 to n^2 where the sum of any row, column, or diagonal of length n is always equal to the same number: the *magic constant*.

You will be given a 3×3 matrix s of integers in the inclusive range $[1, 9]$. We can convert any digit a to any other digit b in the range $[1, 9]$ at cost of $|a - b|$. Given s, convert it into a magic square at *minimal* cost. Print this cost on a new line.

Note: The resulting magic square must contain distinct integers in the inclusive range $[1, 9]$.

For example, we start with the following matrix s:

```
5 3 4
1 5 8
6 4 2
```

We can convert it to the following magic square:

```
8 3 4
1 5 9
6 7 2
```

This took three replacements at a cost of $|5 - 8| + |8 - 9| + |4 - 7| = 7$.

Input Format

Each of the lines contains three space-separated integers of row $s[i]$.

Constraints

- $s[i][j] \in [1, 9]$

Output Format

Print an integer denoting the minimum cost of turning matrix s into a magic square.

Sample Input 0

```
4 9 2
3 5 7
8 1 5
```

Sample Output 0

```
1
```

Explanation 0

If we change the bottom right value, $s[2][2]$, from 5 to 6 at a cost of $|6 - 5| = 1$, s becomes a magic square at the minimum possible cost.

Sample Input 1

```
4 8 2
4 5 7
6 1 6
```

Sample Output 1

```
4
```

Explanation 1

Using 0-based indexing, if we make

- $s[0][1]$->9 at a cost of $|9 - 8| = 1$
- $s[1][0]$->3 at a cost of $|3 - 4| = 1$
- $s[2][0]$->8 at a cost of $|8 - 6| = 2$,

then the total cost will be $1 + 1 + 2 = 4$.

Picking Numbers

Given an array of integers, find and print the maximum number of integers you can select from the array such that the absolute difference between any two of the chosen integers is ≤ 1.

Input Format

The first line contains a single integer, n, denoting the size of the array.
The second line contains n space-separated integers describing the respective values of $a_0, a_1, \ldots, a_{n-1}$.

Constraints

- $2 \leq n \leq 100$
- $0 < a_i < 100$
- The answer will be ≥ 2.

Output Format

A single integer denoting the maximum number of integers you can choose from the array such that the absolute difference between any two of the chosen integers is ≤ 1.

Sample Input 0

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

Sample Output 0

```
3
```

Explanation 0

We choose the following multiset of integers from the array: $\{4, 3, 3\}$. Each pair in the multiset has an absolute difference ≤ 1 (i.e., $|4 - 3| = 1$ and $|3 - 3| = 0$), so we print the number of chosen integers, 3, as our answer.

Sample Input 1

```
6
1 2 2 3 1 2
```

Sample Output 1

```
5
```

Explanation 1

We choose the following multiset of integers from the array: $\{1, 2, 2, 1, 2\}$. Each pair in the multiset has an absolute difference ≤ 1 (i.e., $|1 - 2| = 1$, $|1 - 1| = 0$, and $|2 - 2| = 0$), so we print the number of chosen integers, 5, as our answer.

Climbing the Leaderboard

Alice is playing an arcade game and wants to climb to the top of the leaderboard and wants to track her ranking. The game uses Dense Ranking, so its leaderboard works like this:

- The player with the highest score is ranked number 1 on the leaderboard.
- Players who have equal scores receive the same ranking number, and the next player(s) receive the immediately following ranking number.

For example, the four players on the leaderboard have high scores of 100, 90, 90, and 80. Those players will have ranks 1, 2, 2, and 3, respectively. If Alice's scores are 70, 80 and 105, her rankings after each game are 4^{th}, 3^{rd} and 1^{st}.

Function Description

Complete the *climbingLeaderboard* function in the editor below. It should return an integer array where each element $res[j]$ represents Alice's rank after the j^{th} game.

climbingLeaderboard has the following parameter(s):

- *scores*: an array of integers that represent leaderboard scores
- *alice*: an array of integers that represent Alice's scores

Input Format

The first line contains an integer n, the number of players on the leaderboard.
The next line contains n space-separated integers $scores[i]$, the leaderboard scores in decreasing order.
The next line contains an integer, m, denoting the number games Alice plays.
The last line contains m space-separated integers $alice[j]$, the game scores.

Constraints

- $1 \leq n \leq 2 \times 10^5$
- $1 \leq m \leq 2 \times 10^5$
- $0 \leq scores[i] \leq 10^9$ for $0 \leq i < n$
- $0 \leq alice[j] \leq 10^9$ for $0 \leq j < m$
- The existing leaderboard, $scores$, is in *descending* order.
- Alice's scores, $alice$, are in *ascending* order.

Subtask

For 60% of the maximum score:

- $1 \leq n \leq 200$
- $1 \leq m \leq 200$

Output Format

Print m integers. The j^{th} integer should indicate Alice's rank after playing the j^{th} game.

Sample Input 0

```
7
100 100 50 40 40 20 10
4
5 25 50 120
```

Sample Output 0

```
6
4
2
1
```

Explanation 0

Alice starts playing with 7 players already on the leaderboard, which looks like this:

Rank	Name	Score
1	Emma	100
1	David	100
2	Caroline	50
3	Ritika	40
3	Tom	40
4	Heraldo	20
5	Riley	10

After Alice finishes game 0, her score is 5 and her ranking is 6:

Rank	Name	Score
1	Emma	100
1	David	100
2	Caroline	50
3	Ritika	40
3	Tom	40
4	Heraldo	20
5	Riley	10
6	**Alice**	**5**

After Alice finishes game 1, her score is 25 and her ranking is 4:

Rank	Name	Score
1	Emma	100
1	David	100
2	Caroline	50
3	Ritika	40
3	Tom	40
4	**Alice**	**25**
5	Heraldo	20
6	Riley	10

After Alice finishes game 2, her score is 50 and her ranking is tied with Caroline at 2:

Rank	Name	Score
1	Emma	100
1	David	100
2	Caroline	50
2	**Alice**	**50**
3	Ritika	40
3	Tom	40
4	Heraldo	20
5	Riley	10

After Alice finishes game 3, her score is 120 and her ranking is 1:

Rank	Name	Score
1	**Alice**	**120**
2	Emma	100
2	David	100
3	Caroline	50
4	Ritika	40
4	Tom	40
5	Heraldo	20
6	Riley	10

Sample Input 1

```
6
100 90 90 80 75 60
5
50 65 77 90 102
```

Sample Output 1

```
6
5
4
2
1
```

The Hurdle Race

Dan is playing a video game in which his character competes in a hurdle race. Hurdles are of varying heights, and Dan has a maximum height he can jump. There is a magic potion he can take that will increase his maximum height by 1 unit for each dose. How many doses of the potion must he take to be able to jump all of the hurdles.

Given an array of hurdle heights $height$, and an initial maximum height Dan can jump, k, determine the minimum number of doses Dan must take to be able to clear all the hurdles in the race.

For example, if $height = [1, 2, 3, 3, 2]$ and Dan can jump 1 unit high naturally, he must take $3 - 1 = 2$ doses of potion to be able to jump all of the hurdles.

Input Format

Complete the function $hurdleRace$ in the editor below. The code stub reads the input at passes it to the function. Inputs are in the following format:

The first line contains two space-separated integers n and k, the number of hurdles and the maximum height Dan can jump naturally.
The second line contains n space-separated integers $height[i]$ where $0 \leq i < n$.

Constraints

- $1 \leq n, k \leq 100$
- $1 \leq height[i] \leq 100$

Output Format

Print an integer denoting the minimum doses of magic potion Dan must drink to complete the hurdle race.

Sample Input 0

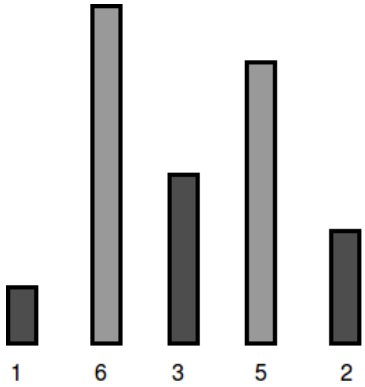
```
5 4
1 6 3 5 2
```

Sample Output 0

```
2
```

Explanation 0

Dan's character can jump a maximum of $k = 4$ units, but the tallest hurdle has a height of $h_1 = 6$:



To be able to jump all the hurdles, Dan must drink $6 - 4 = 2$ doses.

Sample Input 1

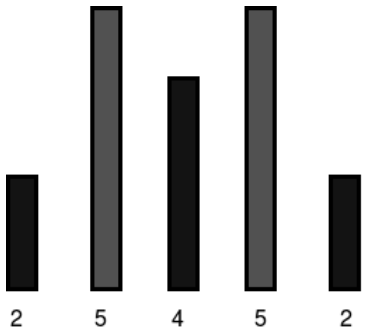
```
5 7
2 5 4 5 2
```

Sample Output 1

```
0
```

Explanation 1

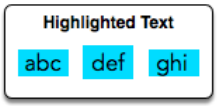
Dan's character can jump a maximum of $k = 7$ units, which is enough to cross all the hurdles:



Because he can already jump all the hurdles, Dan needs to drink 0 doses.

Designer PDF Viewer

When you select a contiguous block of text in a PDF viewer, the selection is highlighted with a blue rectangle. In this PDF viewer, each word is highlighted independently. For example:



In this challenge, you will be given a list of letter heights in the alphabet and a string. Using the letter heights given, determine the area of the rectangle highlight in mm^2 assuming all letters are $1mm$ wide.

Input Format

The first line contains 26 space-separated integers describing the respective heights of each consecutive lowercase English letter, ascii[a-z].
The second line contains a single word, consisting of lowercase English alphabetic letters.

Constraints

- $1 \leq h_? \leq 7$, where $?$ is an English lowercase letter.
- Word contains no more than 10 letters.

Output Format

Print a single integer denoting the area in mm^2 of highlighted rectangle when the given word is selected. Do not print units of measure.

Sample Input 0

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

Sample Output 0

```
9
```

Explanation 0

We are highlighting the word `abc`:

Letter heights are $a = 1$, $b = 3$ and $c = 1$. The tallest letter, `b`, is $3mm$ high. The selection area for this word is $3 \cdot 1mm \cdot 3mm = 9mm^2$.

Note: Recall that the width of each character is $1mm$.

Sample Input 1

```
1 3 1 3 1 4 1 3 2 5 5 5 5 5 5 5 5 5 5 5 5 5 5 7
zaba
```

Sample Output 1

```
28
```

Explanation 1

The tallest letter in $zaba$ is z at $7mm$. The selection area for this word is $4 \times 1mm \times 7mm = 28mm^2$.

Utopian Tree

The Utopian Tree goes through *2* cycles of growth every year. Each spring, it *doubles* in height. Each summer, its height increases by *1* meter.

Laura plants a Utopian Tree sapling with a height of *1* meter at the onset of spring. How tall will her tree be after N growth cycles?

Input Format

The first line contains an integer, T, the number of test cases.
T subsequent lines each contain an integer, N, denoting the number of cycles for that test case.

Constraints
$1 \leq T \leq 10$
$0 \leq N \leq 60$

Output Format

For each test case, print the height of the Utopian Tree after N cycles. Each height must be printed on a new line.

Sample Input

```
3
0
1
4
```

Sample Output

```
1
2
7
```

Explanation

There are *3* test cases.

In the first case ($N = 0$), the initial height ($H = 1$) of the tree remains unchanged.

In the second case ($N = 1$), the tree doubles in height and is 2 meters tall after the spring cycle.

In the third case ($N = 4$), the tree doubles its height in spring ($H = 2$), then grows a meter in summer ($H = 3$), then doubles after the next spring ($H = 6$), and grows another meter after summer ($H = 7$). Thus, at the end of 4 cycles, its height is 7 meters.