IMC Coding Competition 2025 (Advanced Division)

UNSW CPMSoc

12 March 2025

Contents

Monster Fight

Program time limit: 2 seconds

Program memory limit: 512 MB

You have a horde of N monsters, and are engaged in an epic fight against an opponent who also has N monsters. Each monster will have a strength, given as A_i for yours or B_i for your opponents.

You are allowed to decide the matchups, where each of your monsters will fight one of their's. Your monster will only win if they have a higher strength than the one they are fighting.

What is the most wins you can get?

Input

The first line of input contains an integer N, the number of monsters you and your opponent each control.

The second line contains a decreasing list of N integers A, where A_i represents the strength of your i^{th} monster.

The third line contains a decreasing list of N integers B, where B_i represents the strength of your opponent's i^{th} monster.

Constraints

For all test cases:

- $1 \leq N \leq 1,000,000$.
- $1 \leq A_i, B_i \leq 1,000,000$ for all i.
- $A_i \geq A_{i+1}$ for all $i = 1, 2, 3, ...N - 1$.
- $B_i \geq B_{i+1}$ for all $i = 1, 2, 3, ...N - 1$.

Additionally:

- For Subtask 1 (30% of points): $1 \leq N \leq 1000$.
- For Subtask 2 (70% of points): there are no additional constraints.

Output

Output a single integer, the maximum number of wins you can have.

Sample Python Code

```python
# Read from standard input
N = int(input())
A = list(map(int, input().split()))
B = list(map(int, input().split()))

# Calculate your answer here

# Write to standard output
print(answer)
```

Sample C / C++ Code

```c
// Read from standard input
int A[1000005];
int B[1000005];
```

```
int N;
scanf("%d", &N)

for (int i = 0; i < N; i++) {
    scanf("%d", &A[i]);
}

for (int i = 0; i < N; i++) {
    scanf("%d", &B[i]);
}

int ans = 0;
// Calculate your answer here

// Write to standard output
print("%d", &answer)
```

Sample Input 1

```
4
9 8 7 6
8 6 6 6
```

Sample Output 1

```
3
```

Explanation 1

You can match the monsters so you win three times with 9 vs 8, 8 vs 6, and 7 vs 6. However, the 6 vs 6 will not result in a win. It can be shown that it is not possible to do better and get 4 wins, so the answer is 3.

Sample Input 2

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

Sample Output 2

```
3
```

Explanation 2

You can win three times by matching $7 > 6, 5 > 4, 3 > 2, 1 < 8$.

Sample Input 3

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

Sample Output 3

```
0
```

Explanation 3

You cannot win any battles, since your strongest monster is weaker than their weakest.

Scoring

For each subtask (worth 30% and 70% of points, as per the Constraints section), your program will be run on multiple secret test cases one after another, and if it produces the correct output for **all** test cases, it solves that subtask. Your program will receive the points for each subtask it solves. Recall that your final score on the task is the score of your highest scoring submission.

Phranque's Procrastination

Program time limit: 1 seconds

Program memory limit: 512 MB

Frank wants to procrastinate on his assignment, but isn't sure how this will affect his mark. Franque knows that missing one day will result in a `X`% penalty, but this could be applied in one of three methods.

1. His maximum possible score is reduced by X%. `final = min(original, 100-X)`
2. His score is scaled down by X%, and then rounded down to an integer. `final = floor(original x (100-X)/100)`
3. His score has a flat X% deducted, but won't go below 0%. `final = max(original-X, 0)`

Given the original mark N, penalty amount X, and the penalty method, what is a different penalty method, along with a corresponding value of X, such that Phranque would end up with the same final mark? Note that the new penalty value X does not have to be different.

If there are multiple possible solutions, any of them will be accepted as correct.

Input

The first and only line of input contains three integers

- N, the original mark before any penalty is applied,
- X, the penalty amount,
- P, the penalty method using the numbering of the above dot point list.

You should read from standard input.

In Python, you could use the line `N, X, P = map(int, input().split())`.

In C or C++, you could use the line `int N, X, P; scanf("%d%d%d", &N, &X, &P);`.

Constraints

For all test cases:

- $0 \le N \le 100$,
- $0 \le X \le 100$,
- $1 \le P \le 3$.

Output

In one line, output two integers. The first is the new penalty amount and the second is the penalty method.

You should write to standard output.

In Python, you could use the following code.

```
print(new_X, new_method)
```

In C or C++, you could use the following code.

```
WIP
```

Sample Input 1

```
30 3 2
```

Sample Output 1

```
1 3
```

Explanation 1

Phrenque's mark after applying penalty method 2 will be `floor(29.1) = 29`. This is equivalent to using penalty method 3 and deducting 1 percent.

Sample Input 2

100 5 3

Sample Output 2

5 1

Explanation

Franc's mark after applying penalty method 3 will be `max(95, 0) = 95`. This is equivalent to using penalty method 1 and reducing the maximum possible mark of 100 by 5.

Scoring

Your program will be run on both sample cases and fifteen secret cases one after another, and if it produces the correct output for all test cases, it solves this problem. Recall that your final score on the task is the score of your highest scoring submission.

IMC Banner VI

Program time limit: 1 second
Program memory limit: 512 MB

You are making a repeating string of `IMC`s from Scrambled tiles (no relation to anything else), when you realise that some of the tiles have been jumbled. You have exactly $N \div 3$ copies of the letter `I`, $N \div 3$ copies of the letter `M`, and $N \div 3$ copies of the letter `C`, but they are now in no particular order.

You can make a swap by selecting two tiles and switching them around. What is the minimum number of swaps required to rearrange the string into a repeating pattern of `IMC`s?

Input

The first line of input will contain an integer N, the total number of tiles. N will be a multiple of 3.

The second line will be a string S of length N, consisting of exactly $N \div 3$ occurrences of `I`, `M`, and `C`.

You should read from standard input.

In Python, you could use the following code.

```python
N = int(input())
S = input()
```

In C or C++, you could use the following code.

```c
int N;
scanf("%d", &N);
char S[N + 1];
scanf("%s", S);
```

Constraints

For all test cases:

- $3 \leq N \leq 999,999$.

Output

A single integer for the minimum number of swaps required to transform the string into the desired pattern.

You should write to standard output.

In Python, you could use the line `print(answer)`.

In C or C++, you could use the line `printf("%d\n", answer);`.

Sample Input 1

```
6
CIMIMC
```

Sample Output 1

```
2
```

Explanation 1

Swapping the first `C` with the first `I` and then swapping that `I` with the first `M` results in the correct sequence `IMCIMC`.

Sample Input 2

```
9
IMCIMCIMC
```

Sample Output 2

```
0
```

Explanation 2

The string is already in the correct order, so no swaps are needed.

Scoring

Your program will be run on both sample cases and fifteen secret cases one after another, and if it produces the correct output for all test cases, it solves this problem. Recall that your final score on the task is the score of your highest scoring submission.

Squarey

Program time limit: 1 seconds

Program memory limit: 512 MB

Deep in the heart of an ancient kingdom lies the legendary Concentric Fortress, a stronghold protected by layers of magical barriers. Each barrier is either solid and impenetrable or transparent and invisible, creating a mesmerizing pattern when viewed from above.

Your task is to generate a map of the fortress, displaying its layered walls and hidden paths using bricks `#` and spaces.

Input

The first and only line of input contains the integer N, the size of the fortress.

You should read from standard input.

In Python, you could use the line `N = int(input())`.

In C or C++, you could use the line `int N; scanf("%d", &N);`.

Output

Print an $N×N$ map of the castle, where the outermost ring is made of stars. Note that depending on whether N is even or odd, the centre can end up as either a $2x2$ or $1x1$ square.

You should write to standard output.

In Python, you could use the following code.

```python
for row in answer:
    print(row)
```

In C or C++, you could use the following code.

```c
for (int i = 0; i < r; i++) {
    printf("%s\n", answer[i]);
}
```

Constraints

For all test cases:

- $1 \leq N \leq 200$.

Sample Input 1

```
8
```

Sample Output 1

```
########
#      #
# #### #
# #  # #
# #  # #
# #### #
#      #
########
```

Sample Input 2

9

Sample Output 2

```
#########
#       #
# ##### #
# #   # #
# # # # #
# #   # #
# ##### #
#       #
#########
```

Explanation

The outermost barrier is always filled with #. The next layer inside has spaces. The pattern continues inward, alternating between # and spaces, forming nested squares.

Scoring

Your program will be run on both sample cases and fifteen secret cases one after another, and if it produces the correct output for all test cases, it solves this problem. Recall that your final score on the task is the score of your highest scoring submission.

Dero Subs

Program time limit: 1 second

Program memory limit: 512 MB

Nate loves letters but hates change, so has come up with a game to blend the two preferences. The rules are as follows:

1. He begins by looking at a string $S = S_1 S_2 S_3 ... S_N$.
2. For every pair of integers i, j where $1 \leq i \leq j \leq N$, Nate creates substring $S_{ij} = S_i S_{i+1} ... S_{j-1} S_j$. Even if two substrings have the same characters, they are considered as distinct since they have a different i, j pair.
3. For every substring, he labels it as "derogatory" if it alternates between vowels and consonants. He doesn't care which it starts on, nor which it ends on. Any substring of length 1 is automatically considered as "derogatory".
4. As an extra challenge, in each substring, for each instance of `y`, Nate can decide if it should be treated as a vowel or a consonant. If possible, he will try to make decisions such that the substring is "derogatory".
5. He then declares the total number of substrings which are "derogatory".
6. Since this number can get very big, he gives the answer modulo $10^9 + 7$.

Can you write a program which can play Nate's game efficiently?

Input

The first line of input contains an integer N which will be the length of the string. The second line of input contains a string S which is made up of lower case letters.

You should read from standard input.

In Python, you could use the following code.

```
N = int(input())
S = input()
```

In C or C++, you could use the following code.

```
int N;
scanf("%d", &N);
char S[N + 1];
scanf("%s", S);
```

Constraints

For all test cases:

- $1 \leq N \leq 10^6$.

Additionally:

- For Subtask 1 (30% of points): `y` will not appear in the string, and $N \leq 100$.
- For Subtask 2 (20% of points): `y` will not appear in the string, and $N \leq 10,000$.
- For Subtask 3 (20% of points): `y` will not appear in the string.
- For Subtask 4 (30% of points): there are no additional constraints.

Output

Output the total number of "derogatory" substrings.

You should write to standard output.

In Python, you could use the line `print(answer)`.

In C or C++, you could use the line `printf("%d\n", answer);`.

Sample Input 1

```
4
nate
```

Sample Output 1

```
10
```

Explanation 1

The string S can be broken into 10 substrings, which are `'n'`, `'na'`, `'nat'`, `'nate'`, `'a'`, `'at'`, `'ate'`, `'t'`, `'te'`, `'e'`. All 10 substrings are "derogatory", so the answer is 10.

Sample Input 2

```
6
mayfly
```

Sample Output 2

```
11
```

Explanation 2

The string `mayfly` can be broken into 21 substrings, which are `'m'`, `'ma'`, `'may'`, `'mayf'`, `'mayfl'`, `'mayfly'`, `'a'`, `'ay'`, `'ayf'`, `'ayfl'`, `'ayfly'`, `'y'`, `'yf'`, `'yfl'`, `'yfly'`, `'f'`, `'fl'`, `'fly'`, `'l'`, `'ly'`, `'y'`. Of these, the following 10 are NOT "derogatory": `'mayf'`, `'mayfl'`, `'mayfly'`, `'ayf'`, `'ayfl'`, `'ayfly'`, `'yfl'`, `'yfly'`, `'fl'`, `'fly'`. The remaining 11 are "derogatory".

Scoring

For each subtask (worth 30%, 20%, 20%, and 30% of points, as per the Constraints section), your program will be run on multiple secret test cases one after another, and if it produces the correct output for **all** test cases, it solves that subtask. Your program will receive the points for each subtask it solves. Recall that your final score on the task is the score of your highest scoring submission.

Two Rectangles

Program time limit: 1 second

Program memory limit: 512 MB

Jerrie is learning to draw freehand circles on a rectangular piece of paper and decides to challenge themselves. They cut out a smaller rectangle from the page, and try to draw the circle such that it fully encloses the cutout. Wanting to get as much practice as possible, they repeat this multiple times, with a new piece of paper each time.

The coordinate system used is that the lower left corner of the page is treated as $(0, 0)$ and the upper right corner has coordinates (W, H). The smaller rectangle being cut out is aligned with the sides of page.

However, in some situations Jerrie is unsure if it is even possible to draw a circle that stays fully on the page. They ask you, in each scenario, if it was possible, in which case it was a skill issue, or if it was impossible.

Input

The first line of input contains three integers W, H, Q, where

- W is the width of the page
- H is the height of the page
- Q is the number of scenarios

The following Q lines will each contain four integers X_1, Y_1, X_2, Y_2, which describe the inner rectangle being removed.

Constraints

For all test cases:

- $1 \leq Q \leq 100,000$,
- $0 < X_1 < X_2 < W \leq 1,000$
- $0 < Y_1 < Y_2 < H \leq 1,000$

Output

For each scenario, output "Yes" if it is possible for Jerrie to draw a circle, otherwise output "No".

Each scenario's output should be on a new line.

Sample Input 1

```
10 11 3
4 4 6 6
7 7 8 8
1 1 9 10
```

Sample Output 1

```
Yes
Yes
No
```

Explanation

The first two scenarios both have circles that can go enclose the inner rectangle while remaining on the page. The third scenario will either not enclose the inner rectangle, or exceed the page

Scoring

Your program will be run on the sample case and seven secret cases one after another, and if it produces the correct output for all test cases, it solves this problem. Recall that your final score on the task is the score of your highest scoring submission.

Buses

Program time limit: 4 second

Program memory limit: 512 MB

You are given a N people in a queue, each with an intended location "a", "b" or "c". You are also given 3 possible bus capacities: C_1, C_2 and C_3.

At each moment of time, you may do one of three things:

- Add a bus with capacity C_1, C_2 or C_3 and intended destination "a", "b" or "c"
- Add the person at the front of the queue to a bus with the same intended destination
- Remove a full bus

You must move everyone from the queue onto a bus. Minimise the maximum amount of buses you have at a time. There may be partially-filled buses at the end.

Constraints

- $N \leq 100000$
- $C_1, C_2, C_3 \leq 100$

Subtask 1

$N \leq 1000$

Subtask 2

No further constraints

Input

The first line of the input contains values N, C_1, C_2, C_3. The second line of the input contains a string of N characters, the i th character being the location the i person needs to go to.

Output

Output a single number, the smallest possible maximum amount of buses you have at a time in order to put everyone on a bus.

Sample Input 1

```
10 2 3 2
aaaaabbbaa
```

Sample Output 1

```
1
```

Sample Input 2:

```
10 3 4 2
bbaaaabccb
```

Sample Output 2:

```
2
```

Sample Input 3:

```
10 3 4 2
babbcbabac
```

Sample Output 3:

```
3
```

Scoring

For each subtask (worth 40% and 60% of points, as per the Constraints section), your program will be run on multiple secret test cases one after another, and if it produces the correct output for **all** test cases, it solves that subtask. Your program will receive the points for each subtask it solves. Recall that your final score on the task is the score of your highest scoring submission.

Pattern Matching

Program time limit: 2 second Program memory limit: 512 MB

Kyle is wondering if his bitcoin wallet key has been leaked. A bitcoin wallet key is simply a string of M characters. Luckily, he has access to the internet traffic and would like your help in determining how many times his bitcoin wallet key can be seen in the internet traffic.

If the given wallet key of M characters matches x substrings of M characters in the internet traffic string, then we say that it has been leaked x times. However, to account for the occasional disturbances in ther internet traffic, Kyle would like to be on the same side and would classify a substring as a match if it also differs by a single character to the wallet key.

Your task is to count how many substrings in a given internet traffic match the bitcoin wallet key, considering that there can be at up to one mismatch.

Input

The first line of input will contain two integers N and M, the number of characters of the internet traffic and the number of characters of the wallet string.

The second line will be a string T of length N (all lowercase), representing the internet traffic.

The third line will be a string P of length M (all lowercase), represnting the wallet key.

You should read from standard input.

Constraints

For all test cases:

- $1 \le M \le N \le 1,000,000$.

Additionally:

- For Subtask 1 (30% of points): $1 \le N \le 1000$.
- For Subtask 2 (30% of points): $1 \le N \le 100,000$.
- For Subtask 3 (40% of points): $1 \le N \le 1,000,000$.

Output

A single integer for the number of times that the wallet key appears in the internet traffic if we allow up to one mismatch.

You should write to standard output.

Sample Input 1

```
9 2
ilovefood
ov
```

Sample Output 1

```
3
```

Explanation 1

There are three matches `ov`, `oo` and `od`.

Sample Input 2

```
8 4
foodfoud
food
```

Sample Output 2

```
2
```

Explanation 2

There are two matches, `food` and `foud`. ### Scoring

Your program will be run on both sample cases and fifteen secret cases one after another, and if it produces the correct output for all test cases, it solves this problem. Recall that your final score on the task is the score of your highest scoring submission.

Umbrella

Program time limit: 2 seconds

Program memory limit: 512 MB

You are given an array of size N of values $A_1, A_2, \cdot, A_N$ and a value K. An umbrella that covers the locations $[L, R]$ inclusive would have a cost of $|K - |A_R - A_L||$.

What is the minimum cost you need to get umbrellas that cover all locations between $1 and N$ inclusive. Umbrellas can overlap.

Constraints:

- $N \leq 200000$
- $A_i, K \leq 1000000000$

Subtask 1 (40 Points):

$N \leq 10000$ ## Subtask 2 (60 Points): No further constraints.

Input

The first line of input contains two characters N and K.

The second line of input contains the array A, which has N values.

Triangles Matching

Program time limit: 1 second

Program memory limit: 512 MB

NOTE: This problem uses non-standard scoring. You are not expected to get full marks for this problem, but any valid solutions will still award some marks.

You are given N points on a cartesian plane where N is a multiple of three. Each point is at (x_i, y_i). Group these points into $N/3$ triangles, such that the sum of the areas of triangles is minimised.

Input

The first line of input contains one integer N, the total number of points.

The following N lines will each contain two integers, X_i, Y_i, which are the coordinates for point i.

Constraints

For all test cases:

- $N \leq 999$,
- $X_i, Y_i \leq 100000000$.

Output

Give you output over $N/3$ lines, where each line contains three integers, referring to the three indicies of the points to group as a triangle. Each point must be included in exactly one triangle.

Note that the points one indexed, so the smallest index is 1, and the biggest index is N.

The ordering of points within a triangle does not matter. The ordering of triangles in the output does not matter.

Scoring

If your output is not valid, you will receive 0 for the subtask.

Otherwise, let X be the sum of triangle areas. Let $M = 2 \times X \div 10^{14}$.

- If $M \leq 1.5$. You will receive full marks for the subtask.
- Otherwise if $M \leq 3.5$, you will receive a score of $1 - (M - 1.5) \cdot 0.15$.
- Otherwise if $M \leq 11.5$, you will receive a score of $0.7 - (M - 3.5) \cdot 0.05$.
- Otherwise, you will receive a score of 0.3 for this subtask.

Your score is the sum of the the scores in the 10 subtasks.

Sample Input 1

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

Sample Output 1

```
1 2 3
```