ShanghaiTech University

CS240 Algorithm Design and Analysis

Spring 2024

Course Project

Due: 23:59, June 21, 2024

1. This project requires you to solve four problems. For each problem, write a program with the input and output formats as specified in the problem.
2. All programs must terminate within 10 seconds and use at most 4GB of memory. Programs exceeding either limit will not be accepted. You can write your programs in C, C++, Java or Python, though certain languages may lead to more efficient implementations.
3. You may not use third-party libraries in your solution, *e.g.* numpy in Python is not allowed. If you are not sure whether a library is third-party or not, you can try to submit your program, and it will cause a Runtime Error or Compile Error if it is.
4. Your program submission is scored by an Online Judge (OJ) platform containing a number of test cases. Your score depends on the percentage of test cases your program passes. The URL of the OJ will be announced later.
5. To avoid abuse of the OJ, you are allowed at most one submission every five minutes. Violations of this rule may result in penalties to your score.
6. You must **NOT**
   (a) Use online resources or automated tools (such as ChatGPT or similar tools) to solve the problems.
   (b) Read or use solutions written by others.
   (c) Allow other people to read or use your solutions.
   (d) Obtain test data by repeated submissions or other means.

Problem 1:

Consider an endlessly repeating sequence of keys, analogous to an infinite piano keyboard. This sequence follows a specific pattern that repeats indefinitely: `wbwbwwbwbwbw`. You need to determine if a continuous section of this sequence can be found which includes exactly W white keys (denoted by `w`) and B black keys (denoted by `b`).

Constraints:

- W: An integer such that $0 \leq W \leq 100$
- B: An integer such that $0 \leq B \leq 100$
- The sum of W and B must be at least 1 ($W + B \geq 1$), ensuring that you are always looking for a non-empty segment.

Input: The input is given from Standard Input in the following format:
W B

Output: If there is a contiguous substring S containing exactly W '`w`' characters and B '`b`' characters, output `Yes`; otherwise, output `No`.

Sample Input 1

3 2

Sample Output 1

Yes

The first 15 characters of S are `wbwbwwbwbwbwwbw`. You can take the 11'th through 15'th characters to form the string `bwwbw`, which is a substring consisting of three occurrences of `w` and two occurrences of `b`.

Sample Input 2

3 0

Sample Output 2

No

The only string consisting of three occurrences of `w` and zero occurrences of `b` is `www`, which is not a substring of S.

Sample Input 3

92 66

Sample Output 3

Yes

Problem 2:

You are presented with two strings `S` and `T`, and a target string `X` of the same length as `S`, initially filled entirely with the character '#'. Your task is to determine if it's possible to transform `X` into `S` by repeatedly overlaying the string `T` onto `X`.

Constraints:

- The string `S` is composed of letters and has a length N.
- The string `T` also consists of letters and has a length M, where M is less than or equal to N.
- You can overlay `T` on any part of `X`, replacing M consecutive characters, as many times as you want.
- Given lengths: $1 \leq N \leq 2 \times 10^5$ and $1 \leq M \leq \min(N, 5)$.

Input: The input is given from Standard Input in the following format:
N M
S
T

Output: Print **Yes** if it is possible to make `X` match `S`; print **No** otherwise.

Sample Input 1

5 3
ABABC
ABC

Sample Output 1

Yes

Below, let $X[l : r]$ denote the part from the l'th through the r'th character of `X`.
You can make `X` match `S` by operating as follows.

1. Replace $X[3 : 5]$ with `T`. `X` becomes '##ABC'.
2. Replace $X[1 : 3]$ with `T`. `X` becomes 'ABCBC'.

Sample Input 2

7 3
ABBCABC
ABC

Sample Output 2

No

It's impossible to make `X` match `S`.

Sample Input 3

12 2
XYXXYXXYYYXY
XY

Sample Output 3

Yes

Problem 3:

You are given an n digit number f. Based on this number, you will generate a new n digit number g. Let g_i denote the i'th digit of g. The generation rule is that g_1 can be any digit from 0 to 9, and the subsequent digits $g_i, i > 1$ are generated according to the following rule:

$$g_i \leftarrow \left\lfloor \frac{f_i + g_{i-1}}{2} \right\rfloor \text{ or } \left\lceil \frac{f_i + g_{i-1}}{2} \right\rceil$$

Note that based on a single number f, multiple different values of g can be generated.

Input:
The first line contains a nonempty sequence consisting of digits from 0 to 9, which is the number f. The sequence length does not exceed 50 .

Output:
Output the single number — the number of possible g's which can be generated.

Sample Input 1:
12345

Sample Output 1:
48

Sample Input 2:
09

Sample Output 2:
15

Problem 4:

You are organizing a conference, and need to choose the conference dates. The conference must span several consecutive days. Each day, one lecturer is needed to give a presentation, and each lecturer cannot give more than one presentation during the conference. There are n lecturers who can participate in the conference in total, the i'th of whom is available from day l_i to day r_i, inclusive of l_i and r_i. Several consecutive days can be chosen to hold the conference if there is a way to invite available lecturers to each of the days, without inviting lecturers more than once. For k from 1 to n, we want to find out how many ways there are to choose k consecutive days as the conference dates.

Input:

The first line of input contains one integer n, indicating the number of lecturers ($1 \leq n \leq 1 \times 10^4$).

Each of the next n lines contains two integers l_i and r_i, indicating the i'th lecturer is available during these days ($1 \leq l_i \leq r_i \leq 2 \times 10^4$).

Output:

Print n lines, where the k'th line contains one integer indicating the number of ways of selecting a conference of k days.

Sample Input 1:

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

Sample Output 1:

```
6
4
3
```

Explanation:

$k = 1$: all the days 1 to 6 are valid, so there are 6 ways.

$k = 2$: $[1, 2], [2, 3], [3, 4], [4, 5]$ are valid, $[5, 6]$ is not valid, so there are 4 ways.

$k = 3$: $[1, 3], [2, 4], [3, 5]$ are valid, 3 ways.

Sample Input 2:

```
5
1 3
1 3
1 3
1 3
1 3
```

Sample Output 2:

```
3
2
1
0
0
```

Explanation:

$k = 1$: days $1, 2, 3$ are valid.
$k = 2$: $[1, 2], [2, 3]$, are valid, 2 ways.
$k = 3$: $[1, 3]$ is valid, 1 way.
$k = 4$ or 5: There are no enough lecturers to invite, 0 ways.