Backspace problem

Send Feedback

Given a string consisting of lower case characters and hashes(#) where each hash represents a back space .

Find the resultant string after removing the backspaces from the given input string.

(Note : there will be no condition where we use backspace on empty string)

Example :

Input: xy#z

Output: xz

```
import java.util.*;

public class Solution {

    public static String backspace(String s){

    Stack<Character> q = new Stack<Character>();


  for (int i = 0; i < s.length(); ++i)

  {

     if (s.charAt(i) != '#')

        q.push(s.charAt(i));

     else if (!q.isEmpty())

        q.pop();

  }


  // build final string

  String ans = "";


  while (!q.isEmpty())

  {

     ans += q.pop();

  }


  // return final string

  String answer = "";

  for(int j = ans.length() - 1; j >= 0; j--)

  {
```

```
        answer += ans.charAt(j);

    }

    return answer;

}

}
```

Infinite Sequence

Send Feedback

Amit decided to write an infinite sequence. Initially, he wrote 0, and then he started repeating the following process:
* Look at the last element written so far (the l-th element if the sequence has length l so far); let's denote it by x.
*If x does not occur anywhere earlier in the sequence, the next element in the sequence is 0.
*Otherwise, look at the previous occurrence of x in the sequence, i.e. the k-th element, where k<l, this element is equal to x and all elements between the k+1-th and l−1-th are different from x. The next element is l−k, i.e. the distance between the last two occurrences of x.

The resulting sequence is (0,0,1,0,2,0,2,2,1,…): the second element is 0 since 0 occurs only once in the sequence (0), the third element is 1 since the distance between the two occurrences of 0 in the sequence (0,0) is 1, the fourth element is 0 since 1 occurs only once in the sequence (0,0,1), and so on.

Consider the N-th element of the sequence (denoted by x) and the first N elements of the sequence. Find the number of occurrences of x among these N elements.

Input :
The first and only line contains a single integer N.

Output :
print a single line containing one integer — the number of occurrences of the N-th element.

Sample Input :
2

Output :
2

Explanation:
(The 2-nd element is 0. It occurs twice among the first two elements, since the first two elements are both 0.)

```
import java.util.*;

import java.lang.*;

import java.io.*;



class Solution

{

    public static int search(int a[], int x, int index){

        for(int i=index-2;i>=0;i--){

            if(a[i]==x)
```

```
            return i;
        }
        return -1;
    }
        public static void main(String[] args) {
        int a[] = new int[128];
        a[0] = 0;
        for(int i=1;i<128;i++){
            int index = search(a,a[i-1],i);
                if(index==-1)
                    a[i] =0;
            else
                a[i] = i-index-1;
        }
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        int count = 0;
        for(int i=0;i<n;i++){
            if(a[i]==a[n-1])
                count++;
        }

                    System.out.println(count);
            }
}
```