https://course.acciojob.com/idle?question=05f7c96a-217f-4020-a369-35070a1890db

- MEDIUM
- Max Score: 40 Points
-
-

Minimum Number Of Swaps For Bracket Balancing

Ankit and his friend are playing a game in which his friend picks N opening brackets '(' and N closing brackets ')'. He then mixes all of them randomly and generates a string 'BRACKETS'. He asks Ankit to balance 'BRACKETS'.

For example: Here are some examples of balanced BRACKETS "(())", "()()", "(())()".

Ankit can perform the following operation to balance BRACKETS. In one operation, Ankit can pick two adjacent brackets and swap them. His friend challenges him to accomplish the task in minimum possible operations. Ankit needs your help to do this.

Can you help Ankit to make the string 'BRACKETS' balanced in minimum possible swaps?

Input Format

The first line contains one integer 'N' representing the number opening (or closing) brackets.

The second line contains a string' BRACKETS' of length '2*N'.

Output Format

Print the minimum possible swaps required to make string 'BRACKETS' balanced.

Example 1

Input

```
1
()
```

Output

```
0
```

Explanation

The given input string is already balanced. So the minimum number of swaps required to balance ‘BRACKETS’ is 0.

Example 2

Input

```
1
)(
```

Output

```
1
```

Explanation

If we swap position 0 with 1, then the string ‘BRACKETS’ becomes "()" which is balanced. So, the minimum number of swaps needed to balance ‘BRACKETS’ is 1.

Constraints

`1 <= N <= 50000`

Topic Tags

- Math

- Greedy

My code

```
// n java
import java.util.*;

class Solution {
    static int miniNumSwaps(String s, int n) {
        // write code here
                ArrayList<Integer> open = new ArrayList<Integer>();
                char[] v = s.toCharArray();

                for(int i = 0; i < 2*n; i++) {
                        if(v[i] == '(') {
                                open.add(i);
                        }
                }

                int ans = 0, count = 0, p = 0;

                for(int i = 0; i < 2*n; i++) {
                        if(v[i] == '(') {
                                count++;
                                p++;
                        } else {
                                count--;
                        }
                if(count < 0) {
                    int j = open.get(p);
                    ans += (j - i);

                    char t = v[i];
                    v[i] = v[j];
                    v[j] = t;

                    p++;

                    count = 1;
                }
            }

            return ans;
    }
}


public class Main {
    public static void main(String[] args) throws Throwable {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        String brackets = sc.next();
        int ans = Solution.miniNumSwaps(brackets, n);
        System.out.println(ans);
    }
}
```