1. **Singapore flywheel**

Singapore is building another giant wheel, the Duo Wheel. This is created for couples to enjoy Singapore's breathtaking beauty in peace and quiet. There are N cabins in the Duo Wheel. Couples wait at the entrance. Cabins pass the entrance one by one, and when a free cabin passes the entrance, the couple may enter that cabin. The cabins are designed to accommodate only one couple, so if the cabin passing by the entrance is already occupied, the couple waiting at the entrance will have to wait for the next one to arrive. If that cabin is also occupied, the couple will have to wait for the next one after that, and so on, until a free cabin arrives. Whenever a couple is done with their sightseeing, couples exit the cabin before the entrance. The cabins move slow enough to allow couples to exit first and then enter. Couples enter the cabin and then rotate with the wheel for a very long time. For current purposes, all couples sightsee in their cabins till all cabins are filled. Management of Duo Wheel wants to make sure people are not disappointed because of long waiting times, and so have introduced a flexible pricing scheme: when a person approaches the wheel, and the first cabin passing by the entrance is free, they pay N dollars for the ride. If the first cabin is occupied and the couple has to wait for the second one, they pay N-1 dollars for the ride. If the first two cabins are occupied and they have to wait for the third one, they pay N-2 dollars for the ride. Generally, if she has to wait for K occupied cabins to pass by; they pay N-K dollars. In the worst case, when they have to wait for all but one cabin to pass, they will pay just 1 dollar. Let's assume that people approach our wheel at random moments in time, so for each couple approaching the wheel, the next cabin to pass the entrance is taken as the first one they wait for. Let's also assume that nobody will come to the wheel while there's already at least one couple waiting to enter so that we don't have to deal with queues. A couple will always take the first free cabin that passes the entrance. You are given the number of cabins and which cabins are already occupied. How much money are we going to make, on average, until all cabins become occupied? Input The number of cabins followed by a line containing 'O' and 'X'. 'O' denotes free cabins and 'X' denotes occupied ones. Since it is a wheel you can take, the first cabin follows after the last one.

Output: The average amount of money that the Duo Wheel will make when all cabins are occupied. Please calculate up to the 5th decimal point.

|  |  |  |
| --- | --- | --- |
| **Input** | **Expected** | **Actual** |
| 3  O X O | 4.66667 |  |
| 4  X O X O | 6.00000 |  |
| 4  O X X O | 5.75000 |  |

import sys

l=[]  
for line in sys.stdin:  
    l.append(line.strip())

n=int(l[0])  
a=list(map(str,l[1].strip().split()))  
tot=0

for i in range(n):  
    for j in range(n):  
        b=j  
        temp=a[:]  
        start=i  
        wait=0  
        flag=0  
        while(flag!=1):  
            if(temp[start]=='O'):  
                    temp[start]='X'  
                    tot+=n-wait  
                    flag=1  
            elif(temp[start]=='X'):  
                if(start == n-1):  
                    start=0  
                    wait+=1  
                else:  
                    start+=1  
                    wait+=1  
         
        wait=0  
        flag=0  
        while(flag!=1):  
            if(temp[b]=='O'):  
                    tot+=n-wait  
                    flag=1  
            elif(temp[b]=='X'):  
                if(b == n-1):  
                    b=0  
                    wait+=1  
                else:  
                    b+=1  
                    wait+=1  
ans=tot/(n\*n)

print("%.5f"%ans)

1. **Red and green balls**

Red and Green Balls You have a square grid (NxN). Each cell of the grid has either a red ball or a green ball. Your job is to arrange the balls in such a way that all the red balls are either on or below the main diagonal. The main diagonal starts from cell 1x1 and ends at cell NxN. You have only one move which is to swap adjacent rows. You need to achieve the final arrangement in minimal number of moves. If it is not possible to come to a resolution by swapping then print -1

Input:   
First line of input is the number of rows in grid. Rests are the lines in the grid

Output:   
Minimum number of moves

Input   
2   
RG   
RR   
Output   
0   
Input 2   
GR   
RG   
Output   
1

| **Input** | **Expected** | **Actual** |
| --- | --- | --- |
| 2  GR  RG | 1 |  |
| 5  GGRRG  GRRRG  RGGGG  RGGGG  GGGRG | -1 |  |
| 2  RG  RR | 0 |  |
| 5  GGRRG  GRRRG  RGGGG  RGGGG  RGRGG | 6 |  |

def check(mat):  
    for i in range(len(mat)-1):  
        for j in range(i+1,len(mat[i])):  
            if(mat[i][j] == "R"):  
                return False  
    return True  
def find(mat,i):  
    for k in range(i+1,len(mat)):  
        co = 0  
        for c in range(i+1,len(mat)):  
            if(mat[k][c] != "R"):  
                co = co + 1  
                #print(co)  
        if(co == len(mat)-(i+1)):  
            return k  
    return -1  
def swap(mat,i,ans,count):  
    for j in range(ans,i,-1):  
        count = count + 1  
        t = mat[j]  
        mat[j] = mat[j-1]  
        mat[j-1] = t  
    return count  
n = int(input())  
mat = [ ]  
for i in range(n):  
    mat.append(i)  
    mat[i] = input()  
    mat[i] = list(mat[i])

count = 0  
found = False  
if(check(mat)):  
    print(0)  
else:  
    for i in range(len(mat)-1):  
        change = False  
        for j in range(i+1,len(mat)):  
            if(mat[i][j] == "R"):  
                change = True  
        if(change):  
            ans = find(mat,i)  
            #print(ans)  
            if(ans == -1):  
                print(ans)  
                found = True  
                break  
            else:  
                count = swap(mat,i,ans,count)  
                #print(mat)  
        if(check(mat)):  
            print(count)  
            break