SOLUTIONS FOR 24TH OF JULY

ABSURD TRIANGLE 2

```
for(int i = 1; i <= n; i++) {
  for(int j = 1; j <= n-i; j++) {
    printf(".");
  }
  for(int j = 1; j <= 2*i-1; j++) {
    printf("?");
  }
  for(int j = 1; j <= n-i; j++) {
    printf(".");
  }
  printf("\n");
}
```

BOOKS

This problem is an already used problem that I proposed about 2 years ago:

http://codeforces.com/contest/500/problem/C

You can find the solution here: http://codeforces.com/blog/entry/15488 Read the solution of "500C - New Year Book Reading".

CALCULATE IN C++

```
printf("%d + %d = %d\n", a, b, a + b);
printf("%d - %d = %d\n", a, b, a - b);
printf("%d / %d = %d\n", a, b, a / b);
printf("%d %% %d = %d\n", a, b, a % b);
```

POSSIBLE MISTAKES

1. Not printing spaces.
2. When you use printf, you should use "%%" to print the character '%'.

Difference and GCD

Hint: Try to figure out when the solution exists.

Suppose we found a solution $(a, a+h)$. This automatically satisfies the "difference" condition, so we only need to be certain that the GCD of two integers is g. By the Euclidean Algorithm, we know that:

$$\gcd(a, a+h) = \gcd(a+h, a) = \gcd(a, h) = g$$

The GCD condition is equivalent to $\gcd(a, h) = g$. The condition became more simpler, because the values g and h are known, fixed values.

Now, let's figure out what is the condition of g and h, that there exists at least one such a that $\gcd(a, h) = g$. As "GCD" means "greatest common *divisor*", we know that $\gcd(a, h)$ is a *divisor* of h. As we are free to change the values of a, we can make $\gcd(a, h) = g$ if and only if **g is a divisor of h**!

Then, what can be the value of the minimum a? Simple, $a = g$. If $a < g$, $\gcd(a, h) \le a < g$ must hold, so the GCD condition doesn't hold, so this is the optimal solution. So you can just print out $(g, g+h)$.

Evaluating

```
int len = strlen(a);
int ans = 0, p = 0;
for (int i = 0; i < len; i++) {
        if (a[i] == 'O')p++;
        else p = 0;
        ans += p;
}
printf("%d\n", ans);
```

In the code above, the variable p stores the length of consecutive O-s that ends on position i.

Fair Splitting

Suppose we calculated prefix sums $S_i = a_1 + a_2 + \cdots + a_i$ for all $1 \le i \le n$. We can calculate all the values in $O(n)$ using the recurrence relation $S_i = (a_1 + a_2 + \cdots + a_{i-1}) + a_i = S_{i-1} + a_i$.

Then, we can simulate all possible splits. Suppose I took k cards from the top, and you took the remaining $n-k$ cards. Then,

- $m = a_1 + a_2 + \cdots + a_k = S_k$
- $y = a_{k+1} + a_{k+2} + \cdots + a_n = (a_1 + a_2 + \cdots + a_n) - (a_1 + a_2 + \cdots + a_k) = S_n - S_k$

So, we can calculate $|m - y| = |S_k - (S_n - S_k)|$ in $O(1)$ for each $1 \le k < n$. (Note that $k < n$, since you and I both must get at least one card.) Print the minimum among all possible $|m - y|$s.

General Interpretation Of Two Vectors

There are lots of possible solutions, we will describe the general one.

First, how to calculate θ? We will use the *dot product* of two vectors:

$$a \cdot b = |a||b| \cos\theta \Rightarrow \cos\theta = \frac{a \cdot b}{|a||b|} \Rightarrow \theta = \cos^{-1}\left(\frac{a \cdot b}{|a||b|}\right)$$

So we can calculate this value using doubles and the function `acos` defined in `math.h`. *HOWEVER*, because of precision issues, a *domain error may occur*. By some inspection, we found that if $a = kb$ and k is **negative** (a and b are opposite to each other), then $\frac{a \cdot b}{|a||b|} \approx -1$ but not exactly -1 (slightly smaller than -1), so the `acos` function returns `nan`. We can handle this situation by:

- Calculating $\cos^{-1}\left(\max\left(-1, \frac{a \cdot b}{|a||b|}\right)\right)$.
- Do not use the `acos` function(?!) Use the fact that $\cos x$ is strictly decreasing on $0° < x < 180°$, so apply a binary search.
- Separately consider the case when a is parallel to b. $\theta = 0°$ or $\theta = 180°$.

Also do not forget to convert the value into degrees! (not in radians)

Secondly, how to calculate S? We will use the *cross product* of two vectors (we omit the definition):

$$|a \times b| = |a||b| \sin\theta \;\Rightarrow\; S = \frac{1}{2}|a||b| \sin\theta = \frac{1}{2}|a \times b|$$

If you don't want to write the complicated formula of cross product, there is a simple alternative. As we already calculated the value θ, we can just use $S = \frac{1}{2}|a||b| \sin\theta$ directly! However, this will increase the precision error, but we set the limit generously ($|error| \leq 10^{-4}$), so it could also get accepted.