Tenka1 Programmer Contest/Tenka1 Programmer Beginner Contest 2019 解説

DEGwer

2019/04/19

For International Readers: English editorial starts on page 4.

A: On the Way

3 つの座標を読み込み、その大小関係に応じて Yes か No を出力すればよいです。

```
#include<stdio.h>
int main()
{
        int a, b, c;
        scanf("%d%d%d", &a, &b, &c);
        if ((a < c&&c < b) || (b < c&&c < a)) printf("Yes\n");
        else printf("No\n");
}
```

B: *e**** ********e* *e****e* ****e**

問題文通りに文字列操作を行えばよいです。C++ などでは、文字列の添え字は 0 から始まることに注意してください。

```
#include<stdio.h>
int main()
{
        int n, k;
        char s[20];
        scanf("%d%s%d", &n, s, &k);
        k--;
        for (int i = 0; i < n; i++)
        {
```

```
                if (s[i] == s[k]) printf("%c", s[i]);
                else printf("*");
        }
        printf("\n");
}
```

C: Stones

問題文の条件は、左端から連続するいくつかの石を白に、それ以外の石を黒にするという条件です。求めたいものは、各境目に対してそれより左にある黒い石の個数とそれより右にある白い石の個数の和を求めたときの、一番小さい値です。

これは、境目より左にある黒い石の個数と右にある白い石の個数を常に保持しながら、境目を右にすすめていくことで $O(N)$ 時間で計算可能です。

D: Three Colors

与えられる整数の総和を S とします。三角形が作れる条件は、R, G, B のうち一番大きいものが $S/2$ より小さいことに他なりません。

R, G, B のうち一番大きいものが $S/2$ 以上となるような塗り分け方の個数を求めて全体から引くことを考えましょう。R が一番大きいとして一般性を失いません。このとき、$R \geq S/2$ なら $G, B \leq R$ は満たされるので、DP$[t][r]$: 最初の t 個の整数を塗り分け、赤い整数の和が r であるような場合の数 とした DP によってこのような塗り分け方の個数を数えることができます。

このままでは、$R = G = S/2, B = 0$ または $R = B = S/2, G = 0$ のケースを重複して引いてしまいます。このようなものの個数も同じような DP で求められるので、引きすぎたぶんを足してやる必要があります。このような場合の数も同様の DP で求められるので、この問題を解くことができました。時間計算量は $O(N^2 \max a_i)$ です。

E: Polynomial Divisors

整数係数多項式 $f(x)$ が任意の整数 x に対して素数 p の倍数であることは、演算をすべて $\bmod p$ で行うときに f が多項式 $x^p - x$ の倍数であることと同値であることを示します。これが示されれば、2 以上 f の次数以下の全ての素数と、全係数の最大公約数の全ての素因数 p に対し、実際に f を $x^p - x$ で割ってみて割り切れるかどうかを試すことでこの問題を解くことができます。

これは以下のようにして証明できます。フェルマーの小定理より、十分性は明らかです。必要性は、$\bmod p$ で $0, ..., p-1$ がすべて f の根になるという条件より f が p 次式 $x(x-1)...(x-(p-1))$ の倍数であることと、これが $x^p - x$ に一致すること (もし一致しないなら、両者の差を取ることで根を p 個持つ $p-1$ 次以下の多項式が得られて矛盾) から証明されます。

F: Banned X

0 以上 N 以下の全ての整数 K に対して、$1, 2$ のみを K 個並べてどの連続する部分列の中の整数の合計も X にならないようなものの個数が求められれば、それらを 0 の挿入に対応する適切な係数をかけて足し合わせることで答えを求めることができます。以下、この問題を考えることにしましょう。

並んでいるすべての整数の合計を S とします。$S < X$ のときは、任意の並べ方が許されます。$1, 2$ を K 個並べて合計を S 以下にする場合の数を求められればよく、これは

$$\sum_{i=0}^{S-K} \binom{K}{i}$$

です。よって、この場合は解くことができました。なお、これは、K を小さい順に見ていけば、

$$\sum_{i=0}^{t} \binom{K}{i} = 2\sum_{i=0}^{t} \binom{K-1}{i} - \binom{K-1}{t}$$

などを用いて線形時間ですべて計算していくこともできます。

$X \leq S$ のときは、先頭からいくつかの整数の和が $X-1$ になるような場所が存在します (2 以下の整数しか並べられないため)。連続する部分列の中の整数の和は X になれないので、その場所の次の位置に書かれた整数は必ず 2 になります。同様にして、先頭の整数も 2 であることが分かります。以下順繰りに 2 がおかれなければならない場所が決まっていき、結局、条件を満たす列は、

- 先頭の i 個および末尾の i 個の整数はすべて 2
- $2i < X-1$ なら、その他の整数の総和は $X-1-2i$ であり、そうでないなら X は奇数であり並んでいるすべての整数は 2 である

の形をしていることが分かります。後者のパターンは簡単に数えることができます。前者のパターンは、結局、

$$\sum_{i=0}^{t} \binom{2i}{X-1-K}$$

の形をした和を求める問題に帰着できるので、この問題を解くことができました。なお、これは K が大きい順に見ていきながら、

$$\sum_{i=0}^{t} \binom{2i}{j} = \frac{1}{2}\left(\sum_{i=0}^{t} \binom{2i-1}{j-1} + \sum_{i=0}^{2t} \binom{i}{j}\right) = \frac{1}{2}\left(\sum_{i=0}^{t} \binom{2i-1}{j-1} + \binom{2t+1}{j+1}\right)$$

などを用いることで線形時間ですべて計算していくことも可能です。

時間計算量は合計で $O(N^2)$ または $O(N)$ です。

Tenka1 Programmer Contest/Tenka1 Programmer Beginner Contest 2019 Editorial

DEGwer

April 20, 2019

A: On the Way

We just need to read the three coordinates and print Yes or No according to the relation of those values.

```
#include<stdio.h>
int main()
{
        int a, b, c;
        scanf("%d%d%d", &a, &b, &c);
        if ((a < c&&c < b) || (b < c&&c < a))printf("Yes\n");
        else printf("No\n");
}
```

B: *e**** ********e* *e****e* ****e**

We just need to perform the exact operation stated in the problem statement. Note that, in languages such as C++, string indices start at 0.

```
#include<stdio.h>
int main()
{
        int n, k;
        char s[20];
        scanf("%d%s%d", &n, s, &k);
        k--;
        for (int i = 0; i < n; i++)
        {
                if (s[i] == s[k])printf("%c", s[i]);
```

```
                else printf("*");
        }
        printf("\n");
}
```

C: Stones

The condition stated in the problem statement is equivalent to the following: we make some consecutive stones from the left white, and make the other stones black. For each borderline between two stones, we find the sum of the number of black stones to the left of the borderline and the number of white stones to the left of the borderline. Then, the answer is the minimum of these sums.

This can be computed in $O(N)$ time, by maintaining the two counts above and gradually shifting the borderline to the right.

D: Three Colors

Let S be the sum of the given integers. The necessary and sufficient condition for the triangle to exist is the largest of R, G, B being less than $S/2$.

Let us find the number of colorings in which the largest of R, G, B being greater than or equal to $S/2$, and subtract it from the number of all colorings. Without loss of generality, we assume R to be the greatest of the three. Here, if $R \geq S/2$ then $G, B \leq R$ holds, so we can count these colorings by dynamic programming, where DP$[t][r]$ is the number of colorings for the first t integers such that the sum of the red integers is r.

As it is, we would subtract the colorings such that $R = G = S/2, B = 0$ or $R = B = S/2, G = 0$ twice. We need to make up for this duplicated subtraction, and the number of such colorings subtracted twice can be found by the dynamic programming similar to the one above. Thus, the problem is solved with the time complexity of $O(N^2 \max a_i)$.

E: Polynomial Divisors

We will show that the polynomial $f(x)$ with integer coefficients is a multiple of a prime number p for every integer x if and only if f is a multiple of the polynomial $x^p - x$ where all operations are performed mod p. If this is shown, the problem can be solved by attempting to divide f by $x^p - x$ to test for divisibility, for all prime numbers p between 2 and the degree of f and for all prime factors p of the greatest common divisor of all the coefficients.

The proof can be done as follows. From Fermat's little theorem, the sufficiency is obvious. The necessity can be proved from the following fact: since $0, ..., p-1$ are all roots of $f \mod p$, f is a multiple of the p-degree polynomial $x(x-1)...(x-(p-1))$, which is congruent to $x^p - x$ (if they were not congruent, we could take their difference and obtain a polynomial of degree of $p-1$ or less that have p roots, which is a contradiction).

F: Banned X

If we can find, for each integer K from 0 and N, the number of sequences of length K consisting of $1, 2$ in which no contiguous subsequence totals to X, we can find the answer by multiplying those by proper coefficients corresponding to insertion of 0s and summing them up. Let us consider this problem from now on.

Let S be the sum of the integers in the sequence. If $S < X$, any arrangement is allowed. We just need to find the number of sequences of length K consisting of $1, 2$ that total to at most S, and the result is:

$$\sum_{i=0}^{S-K} \binom{K}{i}$$

which solves this case. This can also be computed all in linear time, by dealing with the values of K in ascending order and using, for example, the following fact:

$$\sum_{i=0}^{t} \binom{K}{i} = 2\sum_{i=0}^{t} \binom{K-1}{i} - \binom{K-1}{t}$$

If $X \leq S$, there always exists some position in the sequence such that the sum of the integers from the beginning to that position is $X - 1$ (because we can only use integers that are at most 2). Since the sum of a contiguous subsequence cannot be X, the integer following that position is always 2. Similarly, we see that the integer at the beginning is also 2. More positions successively turn out to contain 2, and in the end we see that the sequence that satisfies the condition has the following form:

- The first i integers and the last i integers are all 2s.
- If $2i < X - 1$, the sum of the other integers is $X - 1 - 2i$. Otherwise, X is odd and all integers in the sequence are 2s.

The sequences in which $2i < X - 1$ does not hold can be easily counted. Counting the sequences in which $2i < X - 1$ holds, reduces to the problem in which we find the sums of the following form:

$$\sum_{i=0}^{t} \binom{2i}{X-1-K}$$

and we have solved this problem. This can also be computed all in linear time, by dealing with the values of K in descending order and using, for example, the following fact:

$$\sum_{i=0}^{t} \binom{2i}{j} = \frac{1}{2}\left(\sum_{i=0}^{t} \binom{2i-1}{j-1} + \sum_{i=0}^{2t} \binom{i}{j}\right) = \frac{1}{2}\left(\sum_{i=0}^{t} \binom{2i-1}{j-1} + \binom{2t+1}{j+1}\right)$$

The total time complexity is $O(N^2)$ or $O(N)$.