ABC053 /ARC068 解説

writer : camypaper

2017 年 1 月 28 日

A : ABC /ARC

整数 x が標準入力から与えられるので x が 1200 未満かどうか判定せよ，という問題です．C++ などのプログラミング言語では以下のように if 文と呼ばれる構文を用いることで条件分岐を行うことが可能です．

```
#include <iostream>
using namespace std;
int main(){
    int x;
    cin >> x;
    if(x < 1200) cout << "ABC" << endl;
    else cout << "ARC" << endl;
}
```

B : A to Z String

文字列 s の部分文字列のうち先頭が A ，末尾が Z であるようなものの最大の長さを求めよ，という問題です．

簡単な考察により， A は可能な限り先頭に近いものを用いるのが最適であり，Z は可能な限り末尾に近いものを選ぶのが最適なことが分かります．それぞれの位置は，$O(|s|)$ で調べることが可能です．

C : X: Yet Another Dice Game

6 面サイコロに対して 90° 回転をしたのち，上を向いている面に書かれた整数を y として y 点得る，という操作を何度も繰り返して合計 x 点以上得るために必要な最小の操作回数を求めよ，という問題です．

はじめ，$1, 6$ 以外の面が上を向くように置き，6 の面と 5 の面が交互に上を向くように転がすのが最適な操作手順です．このように操作を行ったときに必要な操作回数を求めればよく，これは簡単な四則演算で $O(1)$ で求められます．

D : Card Eater

N 枚のカードの山に対して何回か操作を行ってカードに書かれた値が全て異なるようにするとき，最大で何枚のカードを残せるか，という問題です．基本的には 1 枚しかないカードを取り除かれないカードに，余っているカードを取り除かれるカードに選ぶことができるので操作は「2 枚のカードを選んで取り除く」とみなしてほぼ構いません．

カードの山に k 種類のカードがあったとして，k が奇数なら余っているカードは偶数枚あるので答えは k であり，偶数ならばどこかで必ず 1 枚しかないカードを 1 回取り除く必要があるので答えは $k-1$ となります．

E : Snuke-Train

数直線上を原点から一定間隔 d で移動したとき何種類の区間を訪れることが可能か $d = 1, 2, 3, \ldots, M$ について調べよ，という問題です．

間隔 d を固定して考えてみましょう．このとき $r_i - l_i + 1 > d$ を満たす区間は必ず 1 回以上訪れなくてはなりません．また， $r_i - l_i + 1 \leq d$ を満たす区間は最大でも 1 回しか訪れることはできません．

必ず 1 回以上訪れる区間は単に区間の長さによってのみ定まることが分かったので簡単に計算することができます．残っているのは最大でも 1 回しか訪れない区間だけなので，ある位置を含む区間の種類数を累積和などにより求めたのち， d の倍数を全て調べることで $O(N+M)$ で調べることができます．

さて d を固定した場合については解けましたが，$1, 2, 3, \ldots, M$ の M 種類について調べる必要があります．M 以下の数について，1 の倍数,2 の倍数,...,M の倍数と全て調べることは $O(M \log M)$ で行うことが可能ですが，毎回全ての区間について $O(N)$ で調べると $O(NM)$ となり実行時間制限に間に合いません．d を 1 から増加させていくと，必ず 1 回以上訪れる区間の数は減少し，最大でも 1 回しか訪れない区間の数は増加していきます．そのため，N 個の区間を予めその長さごとに分類しておけば各区間について区間に加算する，という操作は 1 回ずつで済み

ます．区間への加算も Fenwick Tree 等を用いることで 1 回あたり $O(\log M)$ で行うことが可能です．解法をまとめると，

- 区間を長さごとに分類する
- d を 1 から M まで順に増加させながら調べる
- i 番目の区間が初めて $r_i - l_i + 1 \leq d$ となったとき，$[l_i, r_i]$ に対して Fenwick Tree 等を用いて 1 加算する
- d の倍数を全て調べる

となり，これらを合わせると $O((M \log M + N) \log M)$ で解くことが可能です．

F: Solitaire

結論から言えば，この問題は最終的に以下のように言い換えられます．

> $(1, 2, \ldots, N)$ の並び替えであるような数列 A のうち，以下の条件を満たすものは何通りあるか？
>
> - $A_k = 1$
> - $k \leq i < j$ なる i, j について $A_i > \max(A_j)$ あるいは $A_i < \min(A_j)$ を満たす．
> - $i < k$ において $A_i > \max(A_j)(i \leq j)$ あるいは $A_i < \min(B_j)(j < i)$ を満たす．

$N - 1$ 個のうち $k - 1$ 個を取って先頭側に，残りを末尾側に条件を満たすように並べることにします．また，末尾側に置いた要素の最大値を m とします．

末尾側の並べ方は 2^{N-k-1} 通りです．先頭側の並べ方について考えます．まず，$A_i > \max(A_j)(i \leq j)$ を満たす要素は赤く塗られている，そうでない要素は青く塗られていると呼ぶことにします．先頭側に置いた要素のうち m 未満のものは全て青く塗る必要があります．m より大きいものは赤く塗ることも青く塗ることも可能ですが，ひとまず赤く塗ったことにしておきます．こうすると，赤く塗られた要素の最小値は青く塗られた要素の最大値よりも大きい，という性質を持つため状態としてまとめやすくなります．

$dp(r, b)$ を「赤く塗られた要素が r 個，青く塗られた要素が b 個あるときの要素の並べ方」とします．この求め方はあとで考えることにすると，このときの並べ方の総数は $\binom{m-2}{N-k-1} dp(N - m, k - 1 - (N - m)) 2^{N-k-1}$ です．なお，$k = N$ の場合は単に $dp(N - 1, 0)$ となることに注意してください．

さて， $dp(r, b)$ は以下の漸化式で定められます．

$$dp(r,b) = \begin{cases} 1 & (r = 0, b = 0) \\ 0 & (r < 0 \text{ または } b < 0) \\ dp(r, b-1) + \sum_{i=0}^{r-1} dp(i, b + r - (i+1)) & (otherwise) \end{cases}$$

この DP では $O(N^3)$ となってしまい間に合いません．しかし，この漸化式の $\sum_{i=0}^{r-1} dp(i, b+r-(i+1))$ という部分の遷移は $r-(i+1)$ 個赤い要素を青く塗り直し，赤い要素を 1 つ取り除く，という形になっています．これを「赤い要素を 1 つ青く塗った次の操作は，赤い要素を 1 つ青く塗るか，赤い要素を 1 つ取り除くかのどちらかでなくてはならない」と考えると，遷移を $O(1)$ にまとめることが可能です．

このようにすると $dp(i,j)$ の前計算に $O(N^2)$ かかり，取得は $O(1)$ となります．まとめて $O(N^2)$ となり正解することができました．

最後に上で示した条件を満たす数列であることと，$2N$ 回の操作で構成可能であることが同値であることを示します．

操作を逆から見ていくことにします．$k \leq i$ を満たす範囲においては A_i は現在デックに含まれるどの数よりも大きい，あるいは小さい，という条件を満たすためデックの中身は 1 が最小値であるような単調増加列となるように並べることが可能です．$i < k$ を満たす範囲においては A_i は現在デックに含まれるどの数よりも大きい，あるいは今後デックに含まれるどの値よりも小さい，という条件を満たします．青く塗られた要素は先頭に，赤く塗られた要素は末尾に挿入することにします．するとデックの中身は途中まで単調減少したのち，1 のある位置を境界としてその後単調増加するような数列になります．こうしてできた数列について 1 を仕切りとして 2 つの単純減少列をマージして新しい単調減少列を作る，と考えると $1, 2, 3, \ldots, N$ という数列を構成することが可能なことが分かります．

次にこの条件を満たさない数列は構成不可能なことを示します．$k \leq i$ を満たす範囲において，A_i が現在デックに含まれるどの数よりも大きい，あるいは小さい，という条件を満たさない箇所があるとすると 1 のある位置を仕切りとした 2 つの単調減少列となるように出来ないため構成することが不可能です．同様に $i < k$ を満たす範囲において，A_i が現在デックに含まれるどの数よりも大きい，あるいは今後デックに含まれるどの値よりも小さい，という条件を満たさない箇所があるとしても，やはり先程と同様に構成することが不可能です．

以上より上で述べた条件を満たす数列であることと，$2N$ 回の操作により構成可能であることが同値であることが示されました．