ARC090 / ABC087 解説

wo01

2018 年 1 月 28 日

A 問題

答えは $X - A$ を B で割った余りとなります。
整数を整数で割った余りを求める演算子は多くの言語で用意されています。
以下に C++ による実装例を示します。

```
#include<cstdio>

using namespace std;

int main(){
    int X, A, B;
    scanf("%d%d%d", &X, &A, &B);
    printf("%d\n", (X - A) % B);
    return 0;
}
```

B 問題

ループを用いて 500 円玉、100 円玉、50 円玉を選ぶ枚数の組 (i, j, k) をすべて試すことで答えが求められます。
以下に C++ による実装例を示します。

```
#include<cstdio>

using namespace std;

int A, B, C;
int X;
```

```
int solve(){
    int ans = 0;
    for(int i = 0; i <= A; ++i){
        for(int j = 0; j <= B; ++j){
            for(int k = 0; k <= C; ++k){
                int tmp = i * 500 + j * 100 + k * 50;
                if(tmp == X) ans++;
            }
        }
    }
    return ans;
}

int main(){
    scanf("%d%d%d%d", &A, &B, &C, &X);
    int ans = solve();
    printf("%d\n", ans);
    return 0;
}
```

C 問題

移動方法は「ある i $(0 \leq i \leq N)$ を選び、以下のように移動する」というものになります。

1. 右に i 回移動する
2. 下に 1 回移動する
3. 右に $N - i$ 回移動する

最初に右に移動する回数 i の候補は $O(N)$ とおりしかなく、その値を決めたとき回収できるアメの個数は $O(N)$ 時間で計算できるので、求める最大値は $O(N^2)$ 時間で計算できます。

D 問題

各 i について $0 \leq x_i \leq 10^9$ であるという条件は考慮しなくて良いことがすぐにわかります。よって、x_i として M 個の情報と矛盾しない限り任意の整数をとることができるものとして考えることにします。

以下のような重み付き有向グラフ G を考えます。

- 頂点は N 個あり、$1, 2, \ldots, N$ の番号が付けられている
- 各 i $(1 \leq i \leq M)$ に対して、頂点 L_i から頂点 R_i に向かう重み D_i の辺と、頂点 R_i から頂点 L_i に向かう重み $-D_i$ の辺が存在する
- それ以外に辺は存在しない

問題の条件を満たす $(x_1, x_2, \ldots, x_N)$ が存在することは、グラフ G の各頂点 v に整数 x_v を割り当てて、G に頂点 u から頂点 v へ向かう重み d の辺があるならば $x_v - x_u = d$ が成り立つようにできることと同値です。

このような割り当てを行うことができるかどうかの判定は、グラフ G に深さ優先探索や幅優先探索を行い、実際に各頂点に値 x を割り当てて矛盾が生じないか確かめることで行えます。

実装の際は、グラフ G が非連結な場合があることに注意してください。

E 問題

各頂点 v について、頂点 S から頂点 v への最短距離を d_v と表すことにします。頂点の組 (u, v) について、頂点 u から頂点 v への最短路を逆向きにすると頂点 v から頂点 u への最短路が得られることに注意します。

頂点 u と頂点 v を結ぶ長さ c の辺 (このような辺を辺 $(\{u, v\}, c)$ と書くことにします) が、頂点 S から頂点 v への最短路に使われうる条件は $d_u + c = d_v$ が成り立つことです。このことを用いると、各頂点 v について、頂点 S から頂点 v への最短路の個数 $dp_1[v]$ を d_v が小さい順に動的計画法を行うことで求めることができます。同様に、頂点 v から頂点 T への最短路の個数 $dp_2[v]$ を d_v が大きい順に動的計画法を行うことで求められます。

dp_1 および dp_2 の値から、以下のことがわかります。

- 2 人が途中で出会うようなものも含めた、2 人の最短路の選び方の組は $dp_1[T]^2$ 通りある
- 2 人が頂点上で出会うような最短路の選び方の組は、$2d_v = d_T$ となる各頂点 v について、$dp_1[v]^2 dp_2[v]^2$ 通りある
- 2 人が辺上で出会うような最短路の選び方の組は、$2d_u < d_T$ かつ $2d_v > d_T$ かつ $d_u + c = d_v$ を満たす各辺 $(\{u, v\}, c)$ について、$dp_1[u]^2 dp_2[v]^2$ 通りある

求めるべき最短路の組の個数は、2 人が途中で出会うようなものも含めた最短路の選び方の組の個数から、2 人が途中で出会うような最短路の選び方の組を引くことで求められます。

F 問題

d 桁の正の整数の個数 $10^d - 10^{d-1}$ を N_d と書くことにします。

$f(r) - f(l) \leq 2$ のとき、$\sum_{l \leq i \leq r} f(i) > (f(l) + 1)N_{f(l)+1}$ となります。$f(r) - 1 \geq 8$ のとき、$(f(l) + 1)N_{f(l)+1} \geq 8N_8 > 10^8$ となるので、問題の条件 $S \leq 10^9$ のもと、$f(r) - f(l) \geq 2$ ならば $f(r) - 1 < 8$ であることがわかります。

このことに注意して、以下のように場合分けを行います。

1. $f(l) = f(r)$ の場合
2. $f(r) < 9$ かつ $f(r) \geq f(l) + 1$ の場合
3. $f(r) \geq 10$ かつ $f(r) = f(l) + 1$ の場合

上記のそれぞれの場合について (l, r) の個数を求め、足し合わせれば良いことになります。

それぞれの場合における (l, r) の個数の求め方について見ていきます。

1. $f(l) = f(r)$ の場合
   値 $d = f(l)$ として考えられるものは S の約数だけです。また、値 $d = f(l)$ を決めると、組 (l, r) の個数は $\max(0, N_d - S/d + 1)$ と求められます。よって、d の値をすべて試すことで組 (l, r) の個数が求められます。
2. $f(r) < 9$ かつ $f(r) \geq f(l) + 1$ の場合
   2 つの組 (l_1, r_1) と (l_2, r_2) がともに条件を満たし、$l_1 < l_2$ が成り立つとき $r_1 < r_2$ も成り立つことに注意して、l としてあり得る値をすべて試すことにより組 (l, r) の個数が求められます。
3. $f(r) \geq 10$ かつ $f(r) = f(l) + 1$ の場合
   整数 t を $10t < S$ かつ t は S の約数でないようなものとします。このとき、$r - l + 1 = t$ であるような (l, r) はちょうど 1 つ存在することがわかります。すなわち、l は $\lfloor S/t \rfloor$ 桁の整数のうち大きい方から $t - (S \mod t)$ 番目、r は $\lceil S/t \rceil$ 桁の整数のうち小さい方から $S \mod t$ 番目となります。
   このことから (l, r) の個数は $S/10$ 未満の整数のうち S の約数でないものの個数と一致し、この値は簡単に求められます。

すべての場合について (l, r) の個数を求めることができたので、これらを足し合わせることで答えが求まります。

ARC090 / ABC087 Editorial

wo01

January 28th, 2018

Problem A

C++ Example:

```
#include<cstdio>

using namespace std;

int main(){
    int X, A, B;
    scanf("%d%d%d", &X, &A, &B);
    printf("%d\n", (X - A) % B);
    return 0;
}
```

Problem B

C++ Example:

```
#include<cstdio>

using namespace std;

int A, B, C;
int X;

int solve(){
    int ans = 0;
    for(int i = 0; i <= A; ++i){
        for(int j = 0; j <= B; ++j){
            for(int k = 0; k <= C; ++k){
                int tmp = i * 500 + j * 100 + k * 50;
                if(tmp == X) ans++;
            }
        }
    }
    return ans;
}

int main(){
    scanf("%d%d%d%d", &A, &B, &C, &X);
    int ans = solve();
    printf("%d\n", ans);
    return 0;
}
```

Problem C

The sequence of movements must be in the following form for some integer i $(0 \leq i \leq N)$:

1. Move to the right i times.
2. Move downward.
3. Move to the right $N - i$ times.

There are $O(N)$ possibilites for i. For a fixed i, we can compute the total number of candies in $O(N)$. Thus, brute force works in $O(N^2)$ time.

Problem D

Consider the following directed graph G:

- There are N vertices numbered $1, 2, \ldots, N$.
- For each i, there are an edge from vertex L_i to vertex R_i with weight D_i, and an edge from vertex R_i to vertex L_i with weight $-D_i$.

The problem asks whether we can assign an integer x_v to each vertex v in G, such that for each edge from u to v with cost d, $x_v - x_u = d$ holds. (Clearly, we can ignore the condition $0 \leq x_i \leq 10^9$.)

We handle each connected component in G independently. For each connected component, we choose an arbitary vertex v and assume that $x_v = 0$. By running a dfs from v, we can uniquly determine the values of x_i in this component. After that, we should check if the conditions are actually satisfied.

Problem E

For each vertex v, let $d1_v$ be the length of the shortest path from S to v. Let $dp_1[v]$ be the number of shortest paths from S to v.

An edge between u and v with cost c can be a part of a shortest path from S to v (in the direction from u to v) if and only if $d1_u + c = d1_v$. By using this, we can compute the values of $dp_1[v]$ from the verteices with smaller values of d_1.

Similarly, define $d2_v$ as the length of the shortest path from T to v, and $dp_2[v]$ as the number of shortest paths from T to v.

From dp_1 and dp_2, we can get the following values:

- The total number of pairs of shortest paths is $dp_1[T]^2$.
- Let $D = d1_T$. For each vertex v such that $d1_v = d2_v = D/2$, there are $dp_1[v]^2 dp_2[v]^2$ pairs of paths such that the two people meet on v.
- Consider an edge between vertices u and v with cost c. If $d1_u < D/2$, $d2_v < D/2$, and $d1_u + d2_v + c = D$, there are $dp_1[u]^2 dp_2[v]^2$ pairs of paths such that the two people meet on this edge (except for endpoints).

The answer is (the first value) - (the second value) - (the third value) mentioned above.

Problem F

Let $N_d = 10^d - 10^{d-1}$ be the number of d-digit integers.

If $f(r) - f(l) \geq 2$, it means that all integers with $l+1$ digits are in the chosen range. Since the total number of digits of 8-digit integers is $8N_8 = 720000000 > 10^8$, $l+1$ must be less than 8 in this case. Now, consider the following two cases:

The case with $f(l) \leq 7$

In this case, $l < 10000000$, and under the constraints of the problem, it turns out that $r < 23000000$. Thus, we can generate the first 23000000 terms of the sequence $f(1), f(2), \cdots$, and we can get the answer by two-pointers method.

The case with $f(l) \geq 8$

Let t be an integer between 1 and $floor(S/8)$. It turns out that, except for special cases, there is exactly one pair (l, r) such that $f(l) \geq 8$, $f(l) + \cdots + f(r) = S$, and $r - l + 1 = t$ (i.e., there are exactly t terms in the chosen range). This is because there is exactly one interval that starts with $t - (S \mod t)$ occurrences of $\lfloor S/t \rfloor$-digit integers followed by $S \mod t$ occurrences of $\lceil S/t \rceil$-digit integers. Thus, we should add $floor(S/8)$ to the answer.

The only exception happens when t is a divisor of S. In this case, there are $N_t - S/t + 1$ such intervals. To handle this, we should add $N_t - S/t$ to the answer for each divisor t between 1 and $floor(S/8)$.

Exercise: can you compute the answer if $S \leq 10^{12}$?