ABC 160 解説

writer: DEGwer, kyopro_friends, latte0119, satashun, ynymxiaolongbao

2020 年 3 月 28 日

For International Readers: English editorial starts on page 7.

A: Coffee

長さ 6 の英小文字からなる文字列に対して、3 文字目と 4 文字目が等しく、5 文字目と 6 文字目も等しいか判定する問題です。

以下は C++ における実装例です。

```
#include <iostream>
using namespace std;
int main(){
  string s;
  cin>>s;
  if(s[2]==s[3]&&s[4]==s[5]){
    cout<<"Yes"<<endl;
  }
  else{
    cout<<"No"<<endl;
  }
}
```

B:Golden Coins

500 円以上持っているのであれば、500 円硬貨 1 枚を手に入れた方が、5 円硬貨 100 枚を手に入れるより嬉しさは大きくなります。よって、できる限り 500 円硬貨を手に入れた後、端数で 5 円硬貨を手に入れるのが最善です。

C 言語での実装例は次のとおりです。

```
#include<stdio.h>
int main(){
    int x;
    scanf("%d",&x);
    int c500 = x/500; 円硬貨の数//500
    int r500 = x%500; 端数//
    int c5 = r500/5; 円硬貨の数//5
    printf("%d\n",c500*1000+c5*5);
}
```

C:Traveling Salesman around Lake

湖の周りを家によって N 個の区間に分けます。セールスマンが全ての家を訪れるとき、N 個の区間のうち通らないものは高々 1 つです。また、N 個の区間のうち最も長いものを通らず、かつ、それ以外の区間をちょうど 1 度だけ通るようにしてすべての家を訪ねることができます。したがってそのような移動方法をとると最短距離となります。最長の区間は $O(N)$ で求めることができるのでこの問題が解けました。実装上は真北 (座標 0) をまたいだ区間の扱いに気を付けてください。

D: Line++

整数 $i, j (1 \leq i < j \leq N)$ を固定します。グラフ G における、頂点 i と頂点 j の最短距離を効率的に求めることを考えます。 頂点 X と頂点 Y を結ぶ辺を使う場合と使わない場合に分けて考えると、求める最短距離は $min\{|j-i|, |X-i|+1+|j-Y|, |Y-i|+1+|j-X|\}$ となります。この式に従って計算することで、 頂点 i と頂点 j の最短距離を $O(1)$ で求めることができます。後は、$1 \leq i < j \leq N$ を満たすすべての i, j について、上記の方法で最短距離を求めれば、 $O(N^2)$ でこの問題を解くことができます。

E: Red and Green Apples

元から赤色のリンゴを X 個以上食べることはないので、美味しさの大きい方から X 個以外は捨てます。元から緑色のリンゴを Y 個以上食べることもないので、美味しさの大きい方から Y 個以外は捨てます。すると残ったリンゴからどのように $X+Y$ 個のリンゴを選んでも、無色のリンゴを適切に着色することで赤色のリンゴ X 個と緑色のリンゴ Y 個にすることができます。よって、残ったリンゴを美味しさの大きい方から $X+Y$ 個食べれば良いです。

F: Distributing Integers

頂点 1 を根とする根付き木として考えます。以下のような値を定義します、

dp_v = 頂点 v を根とする部分木のみを考えたときの、頂点 v に 1 が書かれるような整数の書き方の数

これは、以下の漸化式に従って、ボトムアップに計算することができます。

$$dp_v = (size_v - 1)! \prod_{u \in ch(v)} \frac{dp_u}{size_u!}$$

後は、上記の木 dp を元に全方位木 dp を行うことで、$O(N)$ でこの問題を解くことができます。

A: Coffee

This problems asks to check if a string of length 6 consisting from English lowercase alphabets has the properties that the third and the forth letters are the same and that the fifth and the sixth letters are the same.

The following is a sample code in C++.

```
#include <iostream>
using namespace std;
int main(){
  string s;
  cin>>s;
  if(s[2]==s[3]&&s[4]==s[5]){
    cout<<"Yes"<<endl;
  }
  else{
    cout<<"No"<<endl;
  }
}
```

B:Golden Coins

If he has more than 500 yen, then the happiness will be larger when he has a 500-yen coin than when he has 100 5-yen coins. Therefore, it is optimal to obtain 500-yen coins as much as possible, then obtain 5-yen coins with the fraction.

The following is a sample code in C.

```
#include<stdio.h>
int main(){
    int x;
    scanf("%d",&x);
    int c500 = x/500; //The number of 500-yen coins
    int r500 = x%500; //Fraction
    int c5 = r500/5; //The number of 5-yen coins
    printf("%d\n",c500*1000+c5*5);
}
```

C:Traveling Salesman around Lake

Split the circumference of the pond into the N segments by the houses When the salesman visits all the house, there are at most 1 segment out of the N segments which the salesman does not pass. Also, it is possible to visit all the houses so that the salesman does not pass the longest segment of the N segments while passing the other segments exactly once. The longest segment can be found in a total of $O(N)$ time, so the problem could be solved. When implementing, be careful of the treatment of the segment that strides over the due north.

D: Line++

Fix two integers $i, j (1 \leq i < j \leq N)$. Let us consider an efficient way of finding the shortest distance between vertex i and vertex j. By considering whether or not use the edge between vertex X and vertex Y, the desired shortest distance will be $min\{|j-i|, |X-i|+1+|j-Y|, |Y-i|+1+|j-X|\}$. By following this equation, the shortest distance between vertex i and vertex j can be calculated in an $O(1)$ time. Finally, by finding the shortest distance for all i, j such that $1 \leq i < j \leq N$ with the method above, the problem can be solved in a total of $O(N^2)$ time.

E: Red and Green Apples

Since no more than X red apples will be eaten, we can abandon the apples other than X apples with the largest deliciousnesses. Also, since no more than Y green apples will be eaten, we can abandon the apples other than Y apples with the largest deliciousnesses. Then, no matter how we choose $X + Y$ apples out of the apples left, by properly painting the colorless apples, we can obtain X red apples and Y green apples Therefore, it is optimal to eat the $X + Y$ apples with the largest deliciousness out of the apples left.

F: Distributing Integers

Consider it as a rooted tree with the root being vertex 1. We define the following value:

dp_v =The number of the ways of writing integers, in which integers are only to the subtree with the root being vertex v, such that 1 is written on vertex v.

This value can be calculated bottom-up with the following recurrence equation:

$$dp_v = (size_v - 1)! \prod_{u \in ch(v)} \frac{dp_u}{size_u!}$$

By performing rerooting based on the tree DP above, the problem can be solved in a total of $O(N)$ time.