

搜索入门

隐含搜索树的搜索

1735:【NOIP08普及组】传球游戏

难度

思路

容易搞错的地方及编程技巧

代码

搜索入门

隐含搜索树的搜索

1735:【NOIP08普及组】传球游戏

题目描述 上体育课的时候，小蛮的老师经常带着同学们一起做游戏。这次，老师带着同学们一起做传球游戏。游戏规则是这样的：n个同学站成一个圆圈，其中的一个同学手里拿着一个球，当老师吹哨子时开始传球，每个同学可以把球传给自己左右的两个同学中的一个（左右任意），当老师再次吹哨子时，传球停止，此时，拿着球没传出去的那个同学就是败者，要给大家表演一个节目。聪明的小蛮提出一个有趣的问题：有多少种不同的传球方法可以使得从小蛮手里开始传的球，传了m次以后，又回到小蛮手里。两种传球的方法被视作不同的方法，当且仅当这两种方法中，接到球的同学按接球顺序组成的序列是不同的。比如有3个同学1号、2号、3号，并假设小蛮为1号，球传了3次回到小蛮手里的方式有1->2->3->1和1->3->2->1，共2种。

输入 共一行，有两个用空格隔开的整数n，m（ $3 \leq n \leq 30$ ， $1 \leq m \leq 30$ ）。

输出 共一行，有一个整数，表示符合题意的方法数。

样例输入 3 3 **样例输出** 2 40%的数据满足： $3 \leq n \leq 30$ ， $1 \leq m \leq 20$

100%的数据满足： $3 \leq n \leq 30$ ， $1 \leq m \leq 30$

难度

3星

思路

1. 搜索的思路

假设有5个人，传4次,答案是6

0-->1-->2-->1-->0

0-->1-->0-->1-->0

0-->1-->0-->4-->0

0-->4-->0-->1-->0

0-->4-->0-->4-->0

0-->4-->3-->4-->0

在递归的过程中，我们会发现有些点是重复了。

代码

1. 递归

```
1 #include<bits/stdc++.h>
2 using namespace std;
3 int n,m;
4 int f(int st,int k){//从st出发,还有k步可以用
5     if(k==0){
6         if(st==0) return 1;
7         else return 0;
8     }
9     int ans1=f((st-1+n)%n,k-1);
10    int ans2=f((st+1)%n,k-1);
11    return ans1+ans2;
12 }
13 int main(){
14     cin>>n>>m;
15     cout<<f(0,m);
16     return 0;
17 }
```

2. 记忆化优化

```
1 #include<bits/stdc++.h>
2 using namespace std;
3 long long n,m,mm[40][40];
4 long long f(long long st,long long k){
5     if(k==0){
6         if(st==0) {
7             mm[st][k]=1;
8             return 1;
9         }
10        else {
11            mm[st][k]=0;
12            return 0;
13        }
14    }
15    if(mm[st][k]>=0) return mm[st][k];
16    long long ans1=f((st-1+n)%n,k-1);
17    long long ans2=f((st+1)%n,k-1);
18    mm[st][k]=ans1+ans2;
19    return mm[st][k];
20 }
21 int main(){
22     memset(mm,255,sizeof(mm));
23     cin>>n>>m;
24     cout<<f(0,m);
25     return 0;
26 }
```

3. 动态规划

```
1  #include<iostream>
2  using namespace std;
3  int f[40][40],n,m;
4  int main(){
5      cin>>n>>m;
6      f[0][0]=1;// 从0开始,走0步,到达0,有1种可能性
7      for(int i=1;i<=m;i++){
8          for(int k=0;k<=n-1;k++){
9              f[k][i]=f[(k+1)%n][i-1]+f[(k-1+n)%n][i-1];
10         }
11         cout<<f[0][m]<<endl;
12         return 0;
13     }
```