首先想到，因为有1分硬币的存在，兑换一定可以达成，即使N=0，答案也应该是1种，即哪种硬币都不用

利用这一点，来以3分面值的硬币枚举

假设某种方案要使用i枚3分硬币，那么剩下的就有N-3\*i分需要用2分和1分补全，对于2分硬币，可能使用0，1，·····（N-3\*i）/2枚，剩下的全都用1分硬币即可

也就是说当使用i枚3分硬币时，就会产生出（N-3\*i）/2+1种方案，那么只要枚举i，将所有方案数相加即可~~

+1是全部都是1的情况

#include<cstdio>

#include<algorithm>

#include<cmath>

#include<iomanip>

#include<iostream>

#include<cstring>

#include<map>

using namespace std;

const int INF=1000000007;

int main()

{

//freopen("input.txt","r",stdin);

int n;

while(cin>>n)

{

int ans(0);

for(int i=0;i<=n/3;i++)

ans+=(n-3\*i)/2+1;

cout<<ans<<endl;

}

return 0;

}

完全背包

#include<cstdio>

#include<algorithm>

#include<cmath>

#include<iomanip>

#include<iostream>

#include<cstring>

#include<map>

using namespace std;

const int INF=1000000007;

const int maxn=33000;

int a[5];

int dp[5][33000];

int main()

{

//freopen("input.txt","r",stdin);

int n;

a[1]=1;a[2]=2;a[3]=3;

memset(dp,0,sizeof(dp));

//dp[i][j]代表前i个凑成j元的种类数

for(int i=0; i<=maxn; i++)

dp[1][i]=1;

for(int i=2; i<=3; i++)

for(int j=0; j<=maxn; j++)

{

if(j<a[i])

dp[i][j]=dp[i-1][j];

else

dp[i][j]=dp[i-1][j]+dp[i][j-a[i]];

}

while(cin>>n)

cout<<dp[3][n]<<endl;

return 0;

}

母函数打表：

其实就是在模拟

多项式乘法展开后每个系数的变化过程

(1+ax+bx2)(1+cx3+dx5)

1. 部分 第②部分

那么相乘直接x5会有两部分组成，①的1和②的dx5相乘还有①的bx2和②的cx3相乘

所以x5的系数就是(d+bc)

#include<cstdio>

#include<algorithm>

#include<cmath>

#include<iomanip>

#include<iostream>

#include<cstring>

#include<map>

using namespace std;

const int INF=1000000007;

const int maxn=33000;

int ans[33000];

int trans[33000];

int main()

{

//freopen("input.txt","r",stdin);

for(int i=0;i<maxn;i++)

ans[i]=1;

memset(trans,0,sizeof(trans));

for(int k=2;k<=3;k++)

{

for(int i=0;i<maxn;i++)

for(int j=0;(i+j)<maxn;j+=k)

trans[i+j]+=ans[i];

for(int i=0;i<maxn;i++)

{

ans[i]=trans[i];

trans[i]=0;

}

}

int n;

while(cin>>n)

{

cout<<ans[n]<<endl;

}

return 0;

}