We all love recursion! Don't we?   
  
Consider a three-parameter recursive function w(a, b, c):   
  
if a <= 0 or b <= 0 or c <= 0, then w(a, b, c) returns:   
1   
  
if a > 20 or b > 20 or c > 20, then w(a, b, c) returns:   
w(20, 20, 20)   
  
if a < b and b < c, then w(a, b, c) returns:   
w(a, b, c-1) + w(a, b-1, c-1) - w(a, b-1, c)   
  
otherwise it returns:   
w(a-1, b, c) + w(a-1, b-1, c) + w(a-1, b, c-1) - w(a-1, b-1, c-1)   
  
This is an easy function to implement. The problem is, if implemented directly, for moderate values of a, b and c (for example, a = 15, b = 15, c = 15), the program takes hours to run because of the massive recursion.

Input

The input for your program will be a series of integer triples, one per line, until the end-of-file flag of -1 -1 -1. Using the above technique, you are to calculate w(a, b, c) efficiently and print the result.

Output

Print the value for w(a,b,c) for each triple.

Sample Input

1 1 1

2 2 2

10 4 6

50 50 50

-1 7 18

-1 -1 -1

Sample Output

w(1, 1, 1) = 2

w(2, 2, 2) = 4

w(10, 4, 6) = 523

w(50, 50, 50) = 1048576

w(-1, 7, 18) = 1

典型的记忆化搜索

但自己写就是写不好！！！！！！

#include <stdio.h>

#include <string.h>

#include<iostream>

using namespace std;

long long f[50][50][50];

bool vis[50][50][50];

long long w(int a,int b,int c)

{

if(a<=0 || b<=0 || c<=0)

return 1;

if(a<=20 && b<=20 &&c<=20 && vis[a][b][c])

return f[a][b][c];

if(a<=20 && b<=20 &&c<=20)

vis[a][b][c]=1;

if(a>20 || b>20 || c>20)

return w(20,20,20);

if(a<b && b<c)

return w(a,b,c-1)+w(a,b-1,c-1)-w(a,b-1,c);

return w(a-1,b,c)+w(a-1,b-1,c)+w(a-1,b,c-1)-w(a-1,b-1,c-1);

}

int main()

{

int a,b,c;

memset(vis,0,sizeof(vis));

for(int i=1;i<=20;i++)

for(int j=1;j<=20;j++)

for(int k=1;k<=20;k++)

f[i][j][k]=w(i,j,k);

while(~scanf("%d%d%d",&a,&b,&c))

{

if(a==-1 && b==-1 && c==-1)

return 0;

printf("w(%d, %d, %d) = %d\n",a,b,c,w(a,b,c));

}

return 0;

}