大数阶乘代码修改版与注意

// 这是原来的代码，试试debug？

#include <stdio.h>

#include <stdlib.h>

#define SIZE 50

int Fact(int m,int num[]);

int main()

{

int num[SIZE]={0};

int digit=1;

int i,j;

for (i=1;i<=40;i++)

{

printf("%d! = ",i);

digit=Fact(i,num);

for (j=digit;j>0;j--)

{

printf("%d",num[j]);

}

printf("\n");

}

return 0;

}

int Fact(int m,int num[])

{

int i,j;

int digit=1;

num[0]=1;

for (i=1;i<=m;i++)

{

for (j=0;j<=digit;j++)

{

num[j]=num[j]\*i;

if(num[j]>=10)

{

num[j+1]=num[j]/10;

num[j]=num[j]%10;

digit++;

}

return digit;

}

}

}

**代码：**

#include <stdio.h>

#include <stdlib.h>

#define SIZE 50

void Fact(int m,int num[]);

void Print(int i,int num[]);//新增打印函数体

int main()

{

int num[SIZE]={0};

int i;

num[0]=1;//只要定义一次第一位为1，因为我把调用相乘的函数放到了主函数中，下一次乘只要建立在上一次乘的基础上就可以。

//具体解释见下

for (i=1;i<=40;i++)

{

Fact(i,num);

Print(i,num);

printf("\n");

}

//将主函数中的for语句精简，使得下一次乘只要建立在上一次乘的基础上。

return 0;

}

void Fact(int m,int num[])

//函数功能：模拟竖式运算

{

int j,a=0,temp;//新定义变量a存储进位的数，初值置0，因为第一位乘的时候没有进位数;temp干啥用你一会儿就知道了

//num[0]=1;

/\*for (i=1;i<=m;i++) 这两行不需要，不需要每次都从1乘起，因为这个函数的形参是数组，在函数中改变这个形参数组的

值，在主函数的数组中也会有体现。因此主函数每调用一次这个函数，都会改变

主函数中数组的值，也就是下一次乘只要建立在上一次乘的基础上就可以。

\*/

for (j=0;j<50;j++)

{

num[j]=num[j]\*m;

/\*num[j+1]=num[j]/10;

这步你原来写的有问题。你是先把进位赋给了下一位，这一位上原来要拿来乘的数就没掉了。

然后接下来进到下一位的时候你把这个进位的数拿来乘了。

想想你自己做竖式运算的时候，有进位了以后把进位的这个数拿来乘的吗？

你可以定义一个数来存进位的数。

改动如下所示。

\*/

//变量temp存储的是：正在计算的这位还没进位时候的值

temp=num[j];

//解释：某一位数等于这一位数乘完的结果加上进位数对10取余

num[j]=(num[j]+a)%10;

//存储进位数(新进位数是这一位数乘完的结果(由于num[j]被修改，不能再用了，temp就用上了)加上原进位数整除10)

a=(temp+a)/10;

}

}

void Print(int i,int num[])

{

int j,k;

printf("%d! = ",i);

for (j=49;j>=0;j--)

{

if (num[j]!=0)

{

k=j;

break;

}

}

for (j=k;j>=0;j--) printf("%d",num[j]);

//输出函数的思路是：从最高位开始，找到第一个不为0的数时记下这个数的位置。然后从这个位置开始打印到最低位即可。

}