关于C语言中的递归函数

2019年03月14日 11:09:34 滴水石穿裤子 阅读数：70

递归实例：

```
#include <stdio.h>

void up_and_down(int);

int main(void)
{
        up_and_down(1);
        return 0;

}
void up_and_down(int n)
{
        printf("Level %d: n location %p\n", n, &n); // #1
        if (n < 4)
                up_and_down(n + 1);
        printf("LEVEL %d: n location %p\n", n, &n); // #2
}
```

a. 每级函数调用都有自己的变量。 也就是说， 第1级的n和第2级的n不同， 所以程序创建了4个单独的变量， 每个变量名都是n， 但是它们的值各不相
最终返回 up_and_down()的第1 级调用时， 最初的n仍然是它的初值1。

变量	n	n	n	n
第1级调用后	1			
第2级调用后	1	2		
第3级调用后	1	2	3	
第4级调用后	1	2	3	4
从第4级调用返回后	1	2	3	
从第3级调用返回后	1	2		
从第2级调用返回后	1			
从第1级调用返回后	（全部结束）			

b. 每次函数调用都会返回一次。 当函数执行完毕后， 控制权将被传回上一级递归。 程序必须按顺序逐级返回递归， 从某级up_and_down()返回上一
up_and_down()， 不能跳级回到main()中的第1级调用。

c. 递归函数中位于递归调用之前的语句， 均按被调函数的顺序执行。
d. 递归函数中位于递归调用之后的语句， 均按被调函数相反的顺序执行。
e. 虽然每级递归都有自己的变量， 但是并没有拷贝函数的代码。 程序按顺序执行函数中的代码， 而递归调用就相当于又从头开始执行函数的代码。 除
归调用创建变量外， 递归调用非常类似于一个循环语句。实际上， 递归有时可用循环来代替， 循环有时也能用递归来代替。

f.最后， 递归函数必须包含能让递归调用停止的语句。 通常， 递归函数都使用if或其他等价的测试条件在函数形参等于某特定值时终止递归。

尾递归（tail recursion）:最简单的递归形式是把递归调用置于函数的末尾， 即正好在 return 语句之前。

例如计算阶乘：（0！ 等于1， 负数没有阶乘）

```
long rfact(int n) // 使用递归的函数
{
long ans;
if (n > 0)
    ans = n * rfact(n - 1);
```

```
6 else
7     ans = 1;
8 return ans;
9 }
```

递归的优点：

递归为某些编程问题提供了最简单的解决方案。

递归的缺点：

一些递归算法会快速消耗计算机的内存资源。 另外， 递归不方便阅读和维护。

想对作者说点什么