函数调用机制：

1. 任何函数之间不能嵌套定义，调用函数与被调用函数之间相互独立（彼此可以调用）。
2. 发生函数调用时，被调函数中保存了调用函数的运行环境和返回地址，使得调用函数的状态可以在被调函数返回后完全恢复，而且该状态与被调函数无关。
3. 被调函数的代码体和调用函数的代码体相同，称为递归调用。由于调用点、调用时状态和返回点的不同，被调函数与调用函数运行的栈空间相互独立，函数之间依靠参数传递和返回值来联系，函数看作为黑盒。

例：程序dahua\_test7

int func(int n)

{

*cout* << "n = " << n << *endl*;

int k = 1;

if (n > 0)

{

k += func(--n);

*cout* << "k = " << k << *endl*;

// cout << n << endl;

k += func(--n);

}

return k;

}

int main(void)

{

int a = 2;

*cout* << func(a) << *endl*;

return 0;

}

理解递归函数func，要把被调函数和调用函数看成独立的函数。



func()函数总共被调用了5次，func(2)的最终返回值为5，其中func(2)和func(1)会输出k的值。

输出为：

n = 2

n = 1

n = 0

k = 2

n = -1

k = 4

n = 0

5