

尾调用 (Tail Call)

■ 尾调用：一个函数的最后一个动作是调用函数

□ 如果最后一个动作是调用自身，称为尾递归 (Tail Recursion)，是尾调用的特殊情况

```
void test1() {  
    int a = 10;  
    int b = a + 20;  
    test2(b);  
}
```

```
void test2(int n) {  
    if (n < 0) return;  
    test2(n - 1);  
}
```

■ 一些编译器能对尾调用进行优化，以达到节省栈空间的目的



下面代码不是尾调用

```
int factorial(int n) {  
    if (n <= 1) return n;  
    return n * factorial(n - 1);  
}
```

- 因为它最后1个动作是乘法

尾调用优化 (Tail Call Optimization)

- 尾调用优化也叫做尾调用消除 (Tail Call Elimination)
 - 如果当前栈帧上的局部变量等内容都不需要用了，当前栈帧经过适当的改变后可以直接当作被尾调用的函数的栈帧使用，然后程序可以 jump 到被尾调用的函数代码
 - 生成栈帧改变代码与 jump 的过程称作尾调用消除或尾调用优化
 - 尾调用优化让位于尾位置的函数调用跟 goto 语句性能一样高
- 消除尾递归里的尾调用比消除一般的尾调用容易很多
 - 比如Java虚拟机 (JVM) 会消除尾递归里的尾调用，但不会消除一般的尾调用 (因为改变不了栈帧)
 - 因此尾递归优化相对比较普遍，平时的递归代码可以考虑尽量使用尾递归的形式

尾调用优化前的汇编代码 (C++)

```
void test(int n) {  
    if (n < 0) return;  
    printf("test - %d\n", n);  
    test(n - 1);  
}
```

```
void test(int n) {  
010C1080  push        ebp  
010C1081  mov         ebp,esp  
        if (n < 0) return;  
010C1083  cmp         dword ptr [n],0  
010C1087  jge         test+0Bh (010C108Bh)  
010C1089  jmp         test+2Bh (010C10ABh)  
        printf("test - %d\n", n);  
010C108B  mov         eax,dword ptr [n]  
010C108E  push        eax  
010C108F  push        10C2104h  
010C1094  call        printf (010C1040h)  
010C1099  add         esp,8  
        test(n - 1);  
010C109C  mov         ecx,dword ptr [n]  
010C109F  sub         ecx,1  
010C10A2  push        ecx  
010C10A3  call        test (010C1080h)  
010C10A8  add         esp,4  
}
```

尾调用优化后的汇编代码 (C++)

```
void test(int n) {  
    if (n < 0) return;  
    printf("test - %d\n", n);  
    test(n - 1);  
}
```

■ 汇编教程

▣ <https://ke.qq.com/course/348781>

▣ <https://ke.qq.com/course/336509>

```
void test(int n) {  
00091050  push     ebp  
00091051  mov      ebp,esp  
00091053  and      esp,0FFFFFFF8h  
00091056  push     ecx  
00091057  push     esi  
00091058  mov      esi,ecx  
        if (n < 0) return;  
0009105A  test     esi,esi  
0009105C  js       test+23h (091073h)  
0009105E  xchg     ax,ax  
        printf("test - %d\n", n);  
00091060  push     esi  
00091061  push     offset string "test - %d\n" (0920F8h)  
00091066  call     printf (091010h)  
0009106B  add      esp,8  
        test(n - 1);  
0009106E  sub      esi,1  
00091071  jns      test+10h (091060h)  
}
```