

# 04-B4

栈与队列

调用栈：尾递归

《星期评论》问我“女子解放从那里做起？”我的答案是：  
“女子解放当从女子解放做起。此外更无别法。”

邓俊辉

deng@tsinghua.edu.cn

# 定义

❖ 线性递归中，若递归调用系“最后”一步

❖ `fac(n)` {

```
    if (1 > n) return 1; //base
```

```
    return n * fac( n-1 ); //tail recursion
```

```
}
```

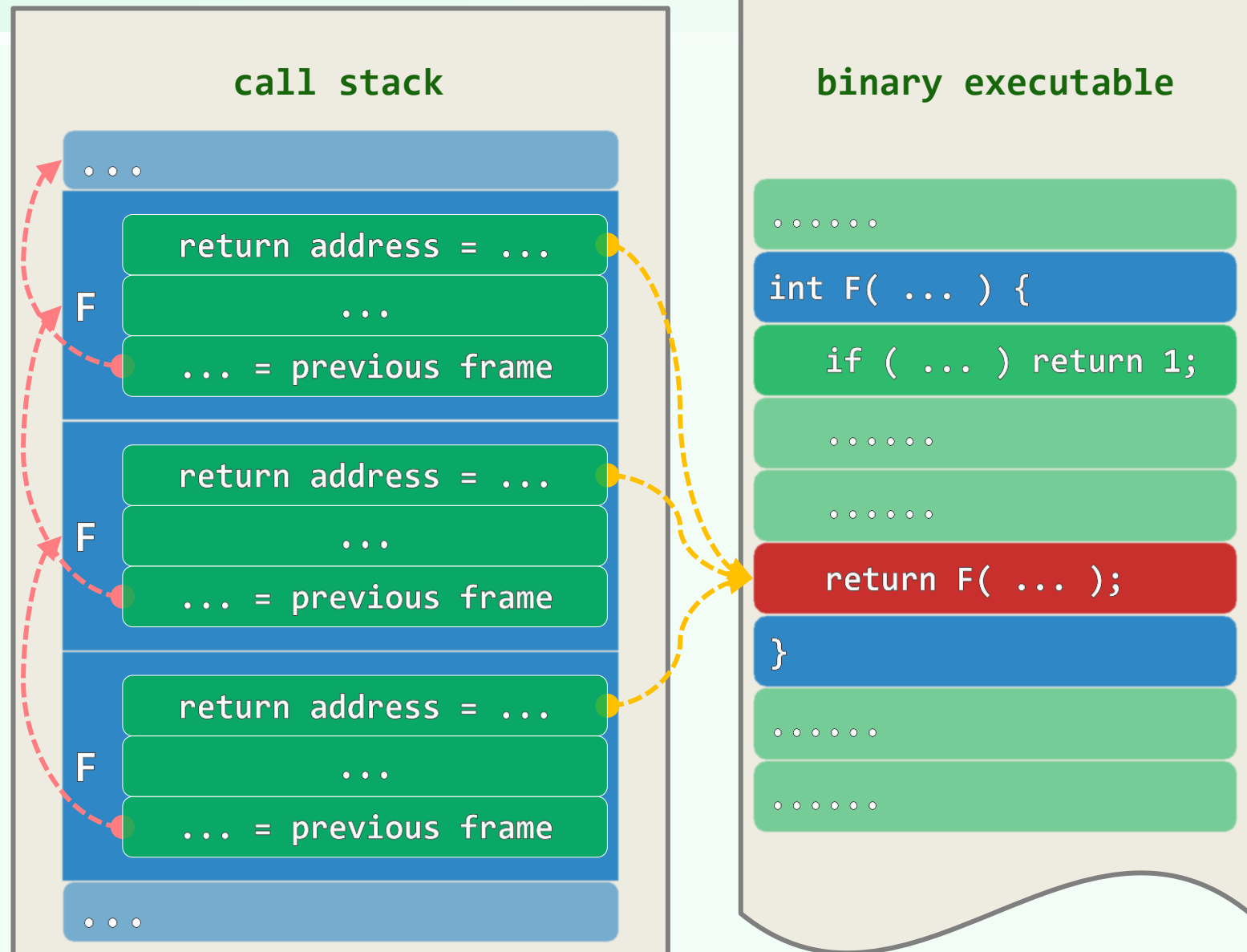
❖ 严格地讲，上例在递归返回前的最后一步是乘法

但这不是本质问题



# 性质

- ❖ 系最简单的递归模式
- ❖ 一旦抵达递归基，便会
  - 引发一连串的return（且返回地址相同）
  - 调用栈相应地连续pop
- ❖ 故不难改写为迭代形式
- ❖ 越来越多的编译器可以自动识别并代为改写
- ❖ 时间复杂度有常数改进  
空间复杂度或有渐进改进



# 消除

<code>fac(n) { //尾递归</code>	<code>fac(n) { //统一转换为迭代</code>	<code>fac(n) { //简捷</code>
	<code>int f = 1; //记录子问题的解</code>	<code>int f = 1;</code>
	<code>next: //转向标志, 模拟递归调用</code>	
<code>if (1 &gt; n) return 1;</code>	<code>if (1 &gt; n) return f;</code>	<code>while (1 &lt; n)</code>
<code>return n * fac( n-1 );</code>	<code>f *= n--;</code>	<code>f *= n--;</code>
	<code>goto next; //模拟递归返回</code>	<code>return f;</code>
<code>} //O(n)时间 + O(n)空间</code>	<code>} //O(n)时间 + O(1)空间</code>	<code>} //O(n)时间 + O(1)空间</code>