



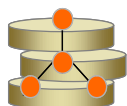
## 第六单元（3）：函数的嵌套调用和递归调用

教学内容	函数的嵌套调用和递归调用
教学目标	
应知	什么是嵌套调用，什么是递归调用
应会	了解嵌套调用与递归调用时程序的执行顺序
	编写简单递归函数
难点	函数的递归

### 教学方法

引探教学法，通过分析题目引导学生思路，尝试写程序的训练。

### 教学过程



1. 求解  $s = 1! + 2! + 3! + 4! + \dots + n!$

- 编写函数 `long fact(int n)` 求  $n$  的阶乘（复习巩固函数的定义、调用、参数等）
- 编写函数 `long factsum(int n)` 求  $1! + 2! + 3! + \dots + n!$ （在此函数中调用 `fact` 函数）
- 编写主函数，输入数据，调用 `factsum` 函数，输出结果。（嵌套调用）

2. 通过求解年龄例子引入递归

3. 将求阶乘 `fact` 函数改为递归函数

4. 分析第一步中的各个  $n$ ，考虑变量的作用域与存储类型

5. 将阶乘函数采用静态变量编写（将阶乘结果定义为静态变量返回）

```
int fac(int n)
{static int f=1;
 f=f*n;
 return f;}
main()
{ int i;
 for(i=1;i<=5;i++)
    printf("%d! = %d\n",i,fac(i)); }
```



## 学生容易出错的地方

- ❏ 混淆函数的声明与定义
- ❏ 参数传递时试图用形参影响实参
- ❏ 函数调用不理解返回值的意义

## 问题与讨论

- ❏ 定义函数时怎样确定参数与返回值



## 小结（可由问题与讨论方式给出）

- ❏ C 语言中，函数可以嵌套调用，不可以嵌套定义
- ❏ 函数递归调用指对函数自身的调用，算法描述为
  1. 递归的算法描述为
    - a. `if ( 递归终止条件) return (条件终止时的值)`
    - b. `else return 递归公式`
  2. 在函数内部定义的变量为局部变量，只在本函数内有效，在函数外定义的变量为全局变量，作用域为从定义位置开始到本源文件结束



## 课后任务

- ❏ 完成项目单 2 的报告