本小节内容

递归调用

递归调用

我们把函数自身调用自身的操作，称为递归函数，递归函数一定要有结束条件，否则会产生死循环!

假设现在要求读者写一个程序来求数字 *n* 的阶乘。读者可能会觉得这很简单，写个 for 循环就可以实现。然而，使用递归来实现更好一些，**因为使用递归在解决一些问题时，可以让问题变得简单，降低编程的难度**。比如接下来的题目：假如有 *n* 个台阶，一次只能上 1 个台阶或 2 个台阶，请问走到第 *n* 个台阶有几种走法？为便于读者理解题意，这里举例说明如下：假如有 3 个台阶，那么总计就有 3 种走法：第一种为每次上 1 个台阶，上 3 次；第二种为先上 2 个台阶，再上 1 个台阶；第三种为先上 1 个台阶，再上 2 个台阶。具体实现请看例 1.1。

【例 1.1】*n* 的阶乘的递归调用实现。

```
#include <stdio.h>

//求 n 的阶乘
int f(int n)
{
	if(1==n)
	{
		return 1;
	}
	return n*f(n-1);
}
//走楼梯
int step(int n)
{
	if(1==n)
	{
		return 1;
	}
	if(2==n)
	{
		return 2;
	}
	return step(n-1)+step(n-2);
}
int main()
```

```
{
    int n;
    int ret;
    scanf("%d",&n);   //请输入数字的大小
    ret=f(n);
    printf("%d\n",ret);
    scanf("%d",&n);   //请输入台阶数
    ret=step(n);
    printf("%d\n",ret);
    return 0;
}
```