零，就继续循环。可以看到，*body-statement* 至少会执行一次。

这种通用形式可以被翻译成如下所示的条件和 goto 语句：

```
loop:
    body-statement
    t = test-expr;
    if (t)
       goto loop;
```

也就是说，每次循环，程序会执行循环体里的语句，然后执行测试表达式。如果测试为真，就回去再执行一次循环。

看一个示例，图 3-19a 给出了一个函数的实现，用 do-while 循环来计算函数参数的阶乘，写作 n!。这个函数只计算 $n > 0$ 时 n 的阶乘的值。

练习题 3.22

A. 用一个 32 位 int 表示 n!，最大的 n 的值是多少？

B. 如果用一个 64 位 long 表示，最大的 n 的值是多少？

图 3-19b 所示的 goto 代码展示了如何把循环变成低级的测试和条件跳转的组合。result 初始化之后，程序开始循环。首先执行循环体，包括更新变量 result 和 n。然后测试 $n>1$，如果是真，跳转到循环开始处。图 3-19c 所示的汇编代码就是 goto 代码的原型。条件跳转指令 jg(第 7 行)是实现循环的关键指令，它决定了是需要继续重复还是退出循环。

```
long fact_do(long n)
{
    long result = 1;
    do {
        result *= n;
        n = n-1;
    } while (n > 1);
    return result;
}
```

a）C 代码

```
long fact_do_goto(long n)
{
    long result = 1;
 loop:
    result *= n;
    n = n-1;
    if (n > 1)
        goto loop;
    return result;
}
```

b）等价的goto版本

```
    long fact_do(long n)
    n in %rdi
1   fact_do:
2     movl    $1, %eax        Set result = 1
3   .L2:                    loop:
4     imulq   %rdi, %rax      Compute result *= n
5     subq    $1, %rdi        Decrement n
6     cmpq    $1, %rdi        Compare n:1
7     jg      .L2             If >, goto loop
8     rep; ret                Return
```

c）对应的汇编代码

图 3-19　阶乘程序的 do-while 版本的代码。条件跳转会使得程序循环