

规则很自然地就与函数调用-返回的顺序匹配。这种实现函数调用和返回的方法甚至对更复杂的情况也适用,包括相互递归调用(例如,过程 P 调用 Q, Q 再调用 P)。


```
long rfact(long n)
{
    long result;
    if (n <= 1)
        result = 1;
    else
        result = n * rfact(n-1);
    return result;
}
```

a) C代码

```
long rfact(long n)
n in %rdi
1  rfact:
2      pushq   %rbx           Save %rbx
3      movq    %rdi, %rbx     Store n in callee-saved register
4      movl    $1, %eax       Set return value = 1
5      cmpq    $1, %rdi       Compare n:1
6      jle     .L35           If <=, goto done
7      leaq    -(1(%rdi)), %rdi Compute n-1
8      call    rfact          Call rfact(n-1)
9      imulq   %rbx, %rax      Multiply result by n
10     .L35:                 done:
11     popq    %rbx           Restore %rbx
12     ret                                Return
```

b) 生成的汇编代码

图 3-35 递归的阶乘程序的代码。标准过程处理机制足够用来实现递归函数

 **练习题 3.35** 一个具有通用结构的 C 函数如下:

```
long rfun(unsigned long x) {
    if ( _____ )
        return _____;
    unsigned long nx = _____;
    long rv = rfun(nx);
    return _____;
}
```

GCC 产生如下汇编代码:

```
long rfun(unsigned long x)
x in %rdi
1  rfun:
2      pushq   %rbx
3      movq    %rdi, %rbx
4      movl    $0, %eax
5      testq   %rdi, %rdi
```