编译原理第七章第一次作业

李昊宸 2017K8009929044

7.2.4. 下面是两个C语言函数f和g的概述：

int f（int x） { int i；…return i+1；…}

int g（int y） {int j；…f（j+1）；…}

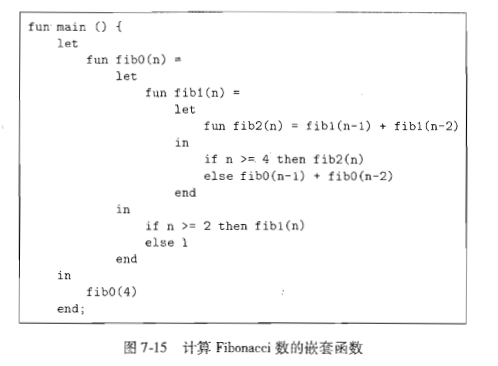
函数g调用函数f。画出在g调用f而f即将返回时，运行时刻栈中从g的活动记录开始的顶端部分。你可以只考虑返回值、参数、控制链以及存放局部数据的空间。你不用考虑存放的机器状态，也不用考虑没有在代码中显示的局部之和临时值。但你应该指出：

1. 哪个函数在栈中为各个元素创建了所使用的空间？
2. 哪个函数写入了各个元素的值
3. 这些元素属于哪个活动记录

答：

|  |  |  |  |
| --- | --- | --- | --- |
| record stack | description | | |
| int y |  | activation record for g |  |
| int g | written by g |
| g control and saved status |  |
| int j | written by g | created by g |
| int x | written by g | activation record for f |
| int f | written by f |
| f control and saved status | written by g |
| int i | written by f | created by f |
|  |  |  |

7.3.1. 图7-15中给出了一个按照非标准方式计算Fibonacci数的ML语言的函数main。函数fib0将计算第n个Fibonacci数（n>=0）。嵌套在fib0中的是fib1，它假设n>=2并计算第n个Fibonacci数。嵌套在fib1中的是fib2，它假设n>=4.请注意，fib1和fib2都不需要检查基本情况。我们考虑从对main的调用开始，直到（对fib0（1）的）第一次调用即将返回的时段，请描述当时的活动记录栈，并给出栈中的各个活动记录的访问链。



答：

先列出活动树：

main()

fib0(4)

fib1(4)

fib2(4)

fib1(3) fib1(2)

fib0(2) fib0(1) fib0(1) fib0(0)

fib1(2)

fib0(1) fib0(0)

黄色标注为第一次调用fib0（1）返回的时刻。橙色路线为当前栈中的活动记录。

|  |
| --- |
| main() |
| fib0(4) |
| access link |
|  |
| fib1(4) |
| access link |
|  |
| fib2(4) |
| access link |
|  |
| fib1(3) |
| access link |
|  |
| fib0(2) |
| access link |
|  |
| fib1(2) |
| access link |
|  |
| fib0(1) |
| access link |
|  |

|  |
| --- |
| d[1] |
| d[2] |
| d[3] |
| d[4] |

7.3.2. 假设我们使用display表来实现下图中的函数。请给出fib0（1）的第一次调用即将返回时的display表。同时指明那时在栈中各个活动记录中保存的display表条目。

|  |
| --- |
| main() |
| fib0(4) |
| saved d[2] |
|  |
| fib1(4) |
| saved d[3] |
|  |
| fib2(4) |
| saved d[4] |
|  |
| fib1(3) |
| saved d[3] |
|  |
| fib0(2) |
| saved d[2] |
|  |
| fib1(2) |
| saved d[3] |
|  |
| fib0(1) |
| saved d[2] |
|  |