```
rangeSum:
04    str  lr, [sp, #-4]! @ Save lr to stack
05    str  fp, [sp, #-4]! @ Save fp to stack
06    mov  fp, sp         @ Point fp to bottom of new frame
07    sub  sp, sp, #12    @ Reserve 3 words of frame space
08    str  r0, [fp, #-8]  @ First param second from TOS
09    str  r1, [fp, #-12] @ Second param at TOS
10    mov  r3, #0         @ sum = 0
11    str  r3, [fp, #-4]  @ sum third from TOS
12    b   .L2
13 .L3:
14    ldr  r3, [fp, #-8]  @ r3 = lo
15    add  r2, r3, #1     @ r2 = lo+1
16    str  r2, [fp, #-8]  @ lo = lo+1
17    ldr  r2, [fp, #-4]  @ r2 = sum
18    add  r3, r2, r3     @ sum = sum + lo
19    str  r3, [fp, #-4]  @ Copy back to frame
20 .L2:
21    ldr  r2, [fp, #-8]  @ r2 = lo
22    ldr  r3, [fp, #-12] @ r3 = hi
23    cmp  r2, r3         @ if r2 < r3
24    blt  .L3            @   loop again
25
26    ldr   r0, [fp, #-4]  @ Return value in R0
27    mov   sp, fp         @ Pop off local vars
28    ldr   fp, [sp], #4   @ Restore fp to old val, and reduce sp
29    ldr   pc, [sp], #4   @ Back to caller
30

 main:
40     str     lr, [sp, #-4]!  @ Save lr to stack
41     str     fp, [sp, #-4]!  @ Save fp to stack
42     mov     fp, sp          @ Point fp to bottom of new frame
43     sub     sp, sp, #4      @ Reserve one word of space for total
44     mov     r0, #10
45     mov     r1, #20
46     bl      rangeSum
47     str   r0, [fp, #-4]  @ Store total into frame
48
49     ldr     r0, .L6         @ R0 gets address of format string
50     ldr     r1, [fp, #-4]   @ R1 gets total
51     bl      printf
52
53     mov     r0, #0          @ Return 0 from main
54     mov     sp, fp          @ Pop off local var total
55     ldr     fp, [sp], #4    @ Restore fp to old val, and reduce sp
56     ldr     pc, [sp], #4    @ Back to compiler-provided caller
57
```