CSE 131 Discussion 5

Function Calls

"More work" calls

- Has "more work" to do after recursive call

```
(def (fact n)
  (if (< n 2) 1
    (* n (fact (+ n -1))
  )
)
(fact 3 1)
```

“No More Work” Calls

- Function with accumulator has “no more work” to do after the function call at the end:

```
(def (fact n sofar)
  (if (< n 2) sofar
    (fact (- n 1) (* n sofar))
  )
)
(fact 3 1)
```

Two argument calling convention

Caller:

1. Move return address, then current rsp, then arguments
2. Subtract to point rsp at the return address

Callee:

1. First argument in [rsp-16], second in [rsp-32]. ([rsp-8] holds original rsp)
2. Start at a “higher” si=4 for any local vars
3. Expect [rsp] to contain return pointer, use ret.

After call:

1. Rely on old rsp at [rsp-16] (a true constant)
2. Expect answer to be in rax from callee

How the program calls `fact`

```
; (+ 1 (fact 3 1))
mov rax, 3
mov [rsp-8], rax
mov rax, 1
mov [rsp-16], rax
mov rbx, after_label
mov [rsp-24], rbx
mov [rsp-32], rsp
mov rax, [rsp-8]
mov [rsp-40], rax
mov rax, [rsp-16]
mov [rsp-48], rax
sub rsp, 24
jmp fact
after_label:
mov rsp, [rsp-16]
```

RSP	~~0x48~~ ~~0x30~~ ~~0x38~~ 0x48
0x18	1
0x20	3
0x28	0x48
0x30	after_label
0x38	1
0x40	3
0x48	

How `fact` calls `fact`

```
;(fact 2 3)
; evaluate the 1st argument
mov [rsp-32], rax
; evaluate the 2nd argument
mov [rsp-40], rax
mov rax, [rsp-32]
mov [rsp-16], rax
mov rax, [rsp-40]
mov [rsp-24], rax
jmp fact
```

RSP	0x30
0x00	
0x08	3
0x10	2
0x18	~~1~~ 3
0x20	~~3~~ 2
0x28	0x48
0x30	after_label