Hi friends,

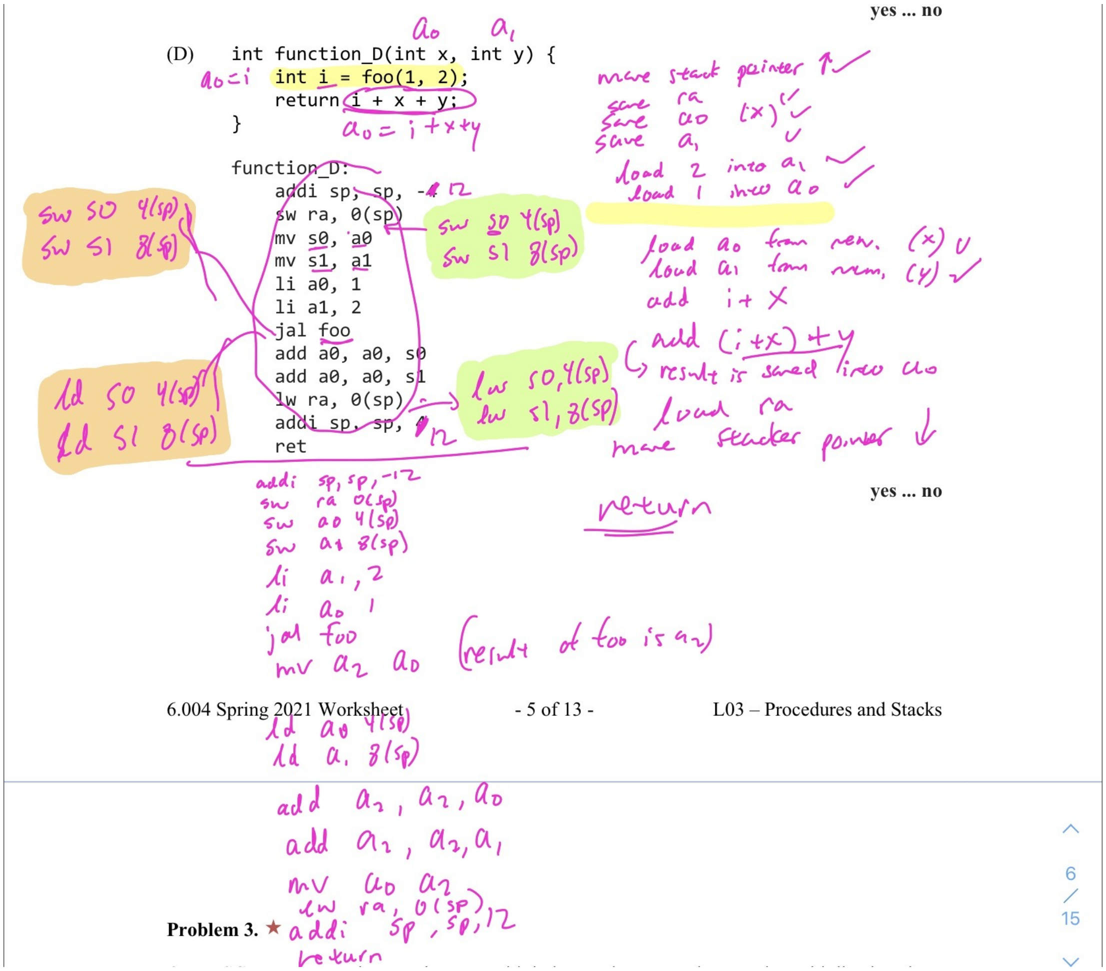
Yasin asked a great question after R2 and I wanted to send out a clarifying email about Callee vs. Caller saved registers based on question 2D from today's recitation.

This is the work I did with R2 for question 2D.
The question was why does the green highlighted part even need to be added? What's the point of having callee vs. caller registers at all?

Noticed that after foo, S0 and S1 were not loaded from memory when they were used in the two "add" functions.

This is because ***S0 and S1 are Callee-saved registers*** so it's ***foo's job to save and load them*** (I put what foo has to do in orange) so that their values are consistent before and after foo is called.

However, why are the green highlighted parts necessary? It's because ***whoever called on function_D is expecting that function_D returns their proper S0 and S1 registers as well***. In this relationship, function_D is the callee. Just like foo is expected to return the right S0 and S1 registers for function_D.



yes ... no

(D)

```
int function_D(int x, int y) {
    int i = foo(1, 2);
    return i + x + y;
}

function_D:
    addi sp, sp, -4
    sw ra, 0(sp)
    mv s0, a0
    mv s1, a1
    li a0, 1
    li a1, 2
    jal foo
    add a0, a0, s0
    add a0, a0, s1
    lw ra, 0(sp)
    addi sp, sp, 4
    ret
```

a0 a1

a0=i

a0 = i+x+y

~~4~~ 12

~~4~~ 12

sw S0 4(sp)
sw S1 8(sp)

sw S0 4(sp)
sw S1 8(sp)

ld S0 4(sp)
ld S1 8(sp)

lw S0, 4(sp)
lw S1, 8(sp)

move stack pointer ↑ ✓
save ra ✓
save a0 (x) ✓
save a1 ✓
load 2 into a1 ✓
load 1 into a0 ✓

load a0 from mem. (x) ✓
load a1 from mem, (y) ✓
add i + X
add (i+x) + y
↳ result is saved into a0
load ra
move stacker pointer ↓

yes ... no

return

addi sp, sp, -12
sw ra 0(sp)
sw a0 4(sp)
sw a1 8(sp)
li a1, 2
li a0 1
jal foo
mv a2 a0 (result of foo is a2)

6.004 Spring 2021 Worksheet - 5 of 13 - L03 – Procedures and Stacks

ld a0 4(sp)
ld a1 8(sp)
add a2, a2, a0
add a2, a2, a1
mv a0 a2
lw ra, 0(sp)
addi sp, sp, 12
return

Problem 3. ★

6
/
15

So what's the TLDR for the difference between Callee vs. Caller saved registers? It all boils down to ***who is responsible for saving the registers.***

Thanks,
Catherine

P.S. The reason I'm not a huge fan of this question is that I think it's much more intuitive to use Caller saved registers for this case. But I think it's important to understand why this is the way it is, so I wanted to send a follow up email to everyone.

Also I used potentially excessive highlighting / bolding so that you don't have to read all of the text if you don't want to xD