Call/Return Protocol - Stack Frame

- observe: registers used for efficient access to temp/intermediate data

- issue: registers are global, software dev. is local

```c
main:
    call foo
    use %ecx

foo:
    use %ecx
    ret
```

```c
access %eax, %ebx
```
one bad solution

every subroutine documents its reg. usage

sub # calls # way to go

=> save registers if to be used in caller routine and caller routine

stack is trad. used for this purpose

who is responsible?
call save registers

main |

lea %eax, %ecx

push %eax
push %ecx

mov %eax, %ecx

mov %ecx, %eax

access %eax

access %ecx

foo |

lea %ecx, %eax

ret
mixture of calls and caller save

caller save Norca, Notca, and Nordic
caller save Nordic, Norca, and Norcy

perform save only when needed based on
local info.
mov %eax to %ebx

main:
    mov %ebx
    mov %ecx
    push %ecx
    call foo
    pop %ecx
    ret
    access %ebx, %ecx

foo:
    ...
Other uses for stack

- passing parameters to subroutines
- local variables within a

3) motivates stack frame

region of stack assoc. w/ particular invocation of func. / subroutine

top of stack always available via ebp
stable reference relating to call base pointer to frame pointer