OOPS. 11

a<b sub a b

j-- check sign bit and overflow bit

set__

if jl would jump, then setl sets a 8-bit register to 1 (else 0)

setl al — if last cmp was <, then al (lower 8 of eax) is set to 0x01

Cardinal

<expr> ::= ...
| isint (<expr>)
| isbool (<expr>)

0x80000001 — true
0x00000001 — false
0x0000000A — 5

0x0000000[1010]

isbool (5) ⟹ false

isbool {
shl eax, 31
or eax, 1

0x0000000[1010]
shl 31
0x[0000]0000000

isint :
xor eax, ...

0x???????[????]
shl 31
0x[?000]0000000

and — keep only
(and eax, 0x3 : keep only last two)
or — set
xor — flip

4 + true

true + true ⟹ 1

||

0x80000001
+ 0x80000001
0x80000002

0x00000008
+ 0x80000001
0x80000009

exit 1 :
⋮

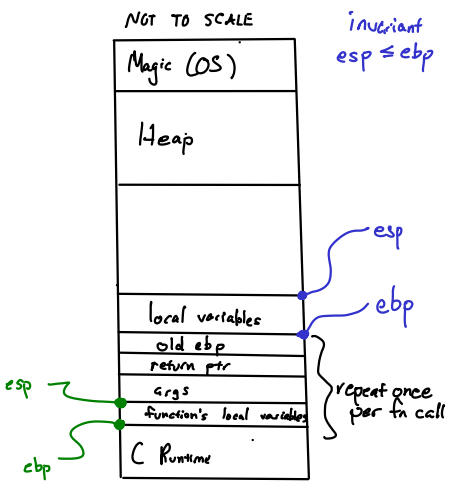
4 + 5

mov eax, ...
and eax, 1
jz good

exit w/ error 1

good:

mov eax, ...
and eax, 1
jnz exit 1



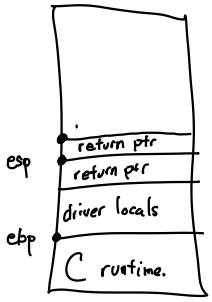
C Calling Conventions

- Caller
- Callee

1a. Caller pushes arg
1b. Caller issues "call" to callee
2a. Callee pushes ebp
2b. Set ebp to esp
2c. Change esp to fit local vars
3. Callee does their thing
4a. Set esp to ebp
4b. Pop into ebp
4c. ret
5a. Caller removes args from stack

In Cardinal:

* Change bird_main to be a C callee
* Change local variable storage to offset from ebp



move esp and write eax →

```
mov eax, 1
push eax
call print
add esp, 4
```

```
let rec memory_of_expr e : int =
  match e with
  | EInt n -> 0
  | EAfter(e') -> memory_of_expr e'
  | EAdd(e1,e2) ->
max (max (memory_of_expr e1)
         (memory_of_expr e2 + 4)
    ) 8
```