

# Meeting 23: Objects

IF A RESEARCHER SAYS A COOL  
NEW TECHNOLOGY SHOULD BE  
AVAILABLE TO CONSUMERS IN...

WHAT THEY MEAN IS...

THE FOURTH QUARTER OF NEXT YEAR	THE PROJECT WILL BE CANCELED IN SIX MONTHS.
FIVE YEARS	I'VE SOLVED THE INTERESTING RESEARCH PROBLEMS. THE REST IS JUST BUSINESS, WHICH IS EASY, RIGHT?
TEN YEARS	WE HAVEN'T FINISHED INVENTING IT YET, BUT WHEN WE DO, IT'LL BE AWESOME.
25+ YEARS	IT HAS NOT BEEN CONCLUSIVELY PROVEN IMPOSSIBLE.
WE'RE NOT REALLY LOOKING AT MARKET APPLICATIONS RIGHT NOW.	I LIKE BEING THE ONLY ONE WITH A HOVERCAR.

## Announcements

- HW6 due 11/17
- Project status due 11/17 (in your project repo)

## Class Scoping Puzzle

The image shows Python code with handwritten annotations explaining variable scoping:

```

x = 1
class C:
    print x
    x = 2
    print x
def f():
    print x
    x = 2
    print x
  
```

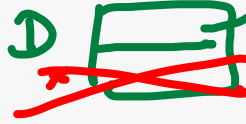
**Annotations:**

- variable in main scope:** Points to the top-level `x = 1`.
- class attribute:** Points to the `print x` line inside the class definition.
- variable vs assign:** A bracket groups the `print x` and `x = 2` lines, with a note "variable vs assign".
- if "repl using" assign:** Points to the `x = 2` line, indicating it's an assignment that updates the class attribute.
- local x:** Points to the `print x` line inside the function `f()`, indicating it refers to a local variable.
- error:** Points to the `print x` line inside `f()` after the assignment, indicating a NameError because the local `x` is not yet defined.
- Diagram:** A box labeled `C` contains a box labeled `x`, representing the class attribute.

result  
the what  
are the assigned to  
of this code

```
class D(C):
    print x    1

print C.x    2
print D.x    2
```



class E(D, C, A)  
- depth-first,  
left-right  
E.foo

```
>>> x = 1
>>>
>>> class C:
...     print x
...     x = 2
...     print x
...
1
2
>>> class D(C):
...     print x
...
1
>>> print C.x
2
>>> print D.x
2
```

class C:

- dynamic copying

x = 12

```
class C:
    if False:
        x = 10
    print x
```

def f():

- static scopy

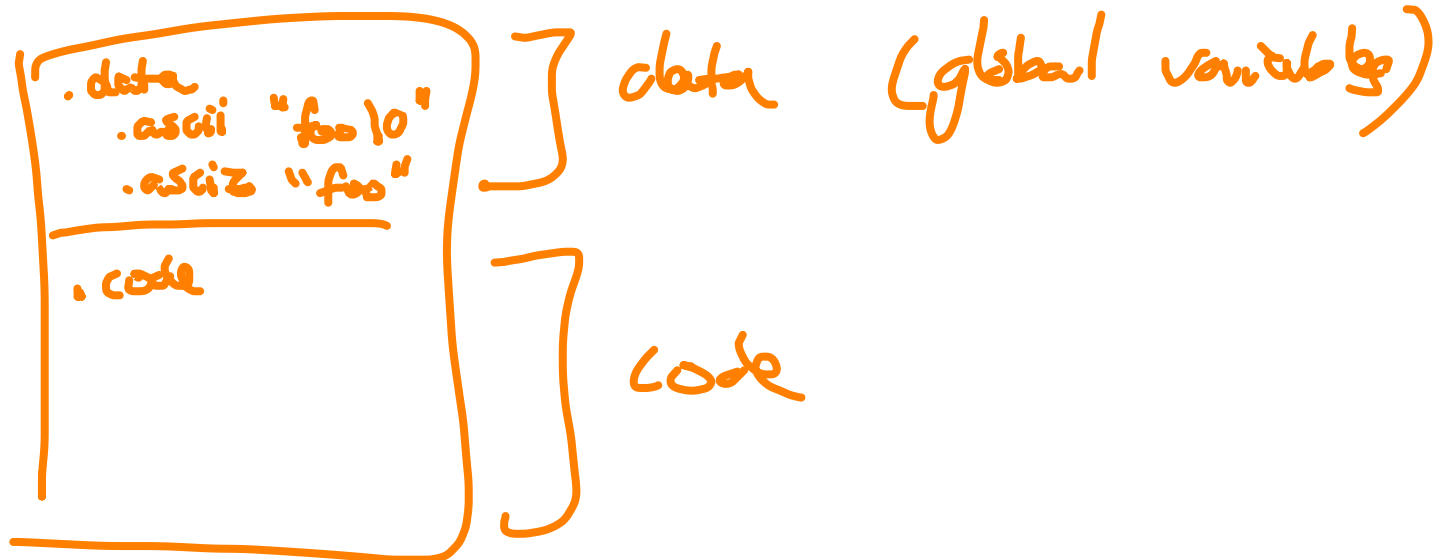
x = 12

```
def f():
    if input False:
        x = 10
```

print x - uninitialized variable

# Questions

- ① Compiling classes 79 ✓
- ② Conditions in loops ✓
- ③ Class attributes ✓
- ④ Strings in assembly ✓



class C:

≡

class B:

≡

topC = create\_class(...)

topB = create\_class(...)

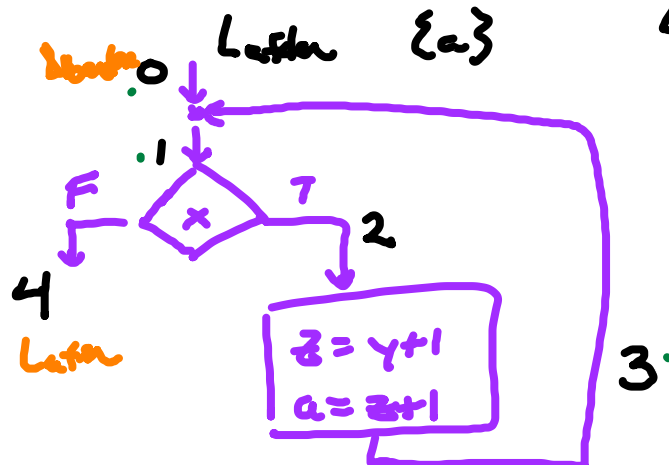
B = topB

↳ set\_attr(topC, "B", ...)

L0	$\emptyset$	$\emptyset$	$\{a, x\}$	$\{a, x\}$	$\{y, x\}$	$\{a, x\}$
L1	$\emptyset$	$\{a, x\}$	$\{a, x\}$	$\{a, x\}$	$\{y, x, a\}$	$\{y, x, a\}$
L2	$\emptyset$	$\emptyset$	$\emptyset$	$\{y, x\}$	$\{a, y\}$	$\{a, y\}$
L3	$\emptyset$	$\emptyset$	$\{a, x\}$	$\{a, x\}$	$\{a, x\}$	$\{y, x, a\}$
L4	$\{a\}$	$\{a\}$	$\{a\}$	$\{a\}$	$\{a\}$	$\{a\}$

while 'x':  
 2 z = y + 1  
 3 a = z + 1

↳ L0  
 ↳ L1  
 ↳ L2  
 ↳ L3  
 ↳ L4



while c:  
 S ← body

↳ *object code*

def instanceof(n, While):  
 L1 = L4 ∪ ... and ...  
 While *fixpoint*  $\text{set}(L1) = L1$   
 ①  $\Delta 3 = \dots - L1 \dots, L0 = \dots - L1 =$   
 ②  $L2 = \text{liveness}(n.\text{body}, L3)$   
 ③  $L1 = L1 \cup L2$

*meta code*

$L_{\text{after}}^0$  while,  $x_{\text{cond}}$  :-  
2  $S_{\text{body}}$   
3

4  $L_{\text{after}} =$

$L_{\text{after}}$

$L_{\text{before}}(s, L)$

Solve def  
 $L_0 \stackrel{\text{def}}{=} L_1$   
 $L_1 \stackrel{\text{def}}{=} L_2 \cup L_4 \cup \{x_{\text{cond}}\}$   
 $L_2 \stackrel{\text{def}}{=} L_{\text{before}}(S_{\text{body}}, L_3)$   
 $L_3 \stackrel{\text{def}}{=} L_1$   
 $L_4 \stackrel{\text{def}}{=} L_{\text{after}}$

and  
return  $L_0$















