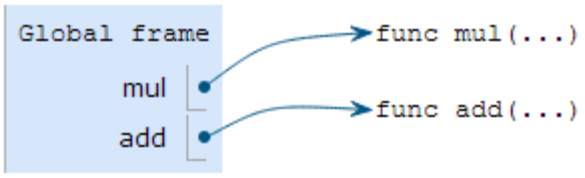
CS61A // Environment Diagrams

This is a general reference sheet for what we've learned so far! There will be additional rules as the semester continues as we learn more complicated functions.

General Environment Diagram Steps

1. Draw your global frame
    - Assign each top-level function definition and global variable to its variable
    - Note: If you import a function, you draw the function sitting outside the global frame
        i.e. "`from operator import add, mul`"
        draw def add(...) and def mul(...) outside your global frame pointed to from their variables within the frame, as follows:



        We use ellipsis (...) because we didn't define the function and thus, we don't know the true formal parameters
2. User-defined function calls
    - Make sure you use the function's *intrinsic* name when you label new frames
        For example, if you've reassigned foo = bar, and you call foo(2,3), you should label the new frame "bar", the intrinsic, or evaluated name
    1. Draw the local frame with its intrinsic name
    2. Assign the formal parameters
    3. Evaluate the body of the function

Tips:

- Remember to always use the *evaluate and apply* logic, evaluating the operator and operands first before applying the operator to the operands.
    - This means that operator and operands will be evaluated in the same frame.
    i.e.

```
def add(a, b):
      return a + b
def sub(a, b):
      return a - b
add(2, sub(5, 2))
```

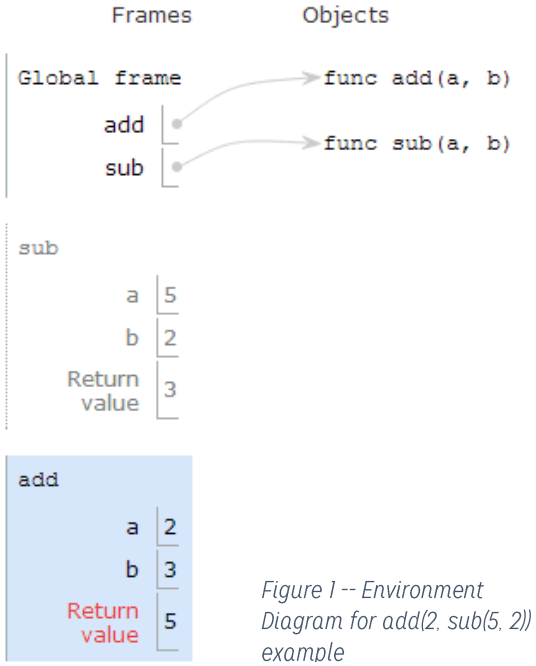


Figure 1 -- Environment Diagram for add(2, sub(5, 2)) example

- No variable name can be repeated within a frame. Cross out previous arrows and reassign the variable to its new value
- Calling built in functions (i.e. print, max, min, etc) don't get a local frame because you didn't define them! Instead, you evaluate them and just assign their value correctly

Created by Michelle Hwang, Fall 2013