# TIPS FOR MIDTERM 1

Computer Science 61A . September 15, 2015 . alvinwan.com/cs61a

This tip sheet does not comprehensively cover *all* types of midterm questions. Instead, it covers the three top types of problems.

## 01. ENVIRONMENT DIAGRAMS

See an outline of the steps at http://sumukh.me/?/61a-enviroment-diag. It lists rules for all tasks in an environment diagram. You should also draw environment diagrams for "Interactive Output" questions on your midterm. Here is a recommended checklist to run through after completing your diagram.

1. All frame names contain f<integer>: <intrinsic name>. In other words, check that all of the intrinsic names are actually taken from the right side of your environment diagram. They *are not* the variables defined in a frame. They are the names of the function defined on the right side.
2. Check that the parent of each function is where the function is defined, *not where it is called*.
3. Double-check that you do *not* have frames for built-in or imported functions (you do not know how these were implemented).

## 02. FILL IN THE BLANKS

Midterms feature fill-in-blank questions that cover topics such as recursion, iteration, and lambdas. Note that these mold your approach to coding by encouraging you to understand the most efficient and human-readable way to code.

### Lambda Functions

1. Convert into function form (complete function definitions using def, instead of lambda)
2. If a blank is in the function parameters, check which variable is undefined in the body of the function.
3. If a blank is in a function call, like f(______), determine what f is and then what to pass to f.

### Recursive Functions

1. Determine the types of each variable. (boolean, integer, float, string, None, or function)
2. Identify the three parts to any recursive function: base case, recursive case, and what is being reduced with each function call.
3. According to what is given and if doctests are not provided, list expected output for the program. Find a pattern, and use that solve the problem.
4. If it comes down to it, guess and check. Then, fix.

## 03. ADDITIONAL NOTES

Here are additional notes to remember or write down:

- `f1(f2(f3(5))))` indeed evaluates *the inner function first*. Therefore, you write the innermost frame first. *Operands* are evaluated before operators (arguments are evaluated before the function), but the *call expression* (actually calling expression) occurs starting from the innermost function call. So, the first frame you write is f3.
- Look up undefined variables in the function's parent, which is strictly defined as the frame in which it was defined.