# **CIS 61 :: Lab 03 - Higher Order Functions**

#### **Student Name: Collin Hargreaves**

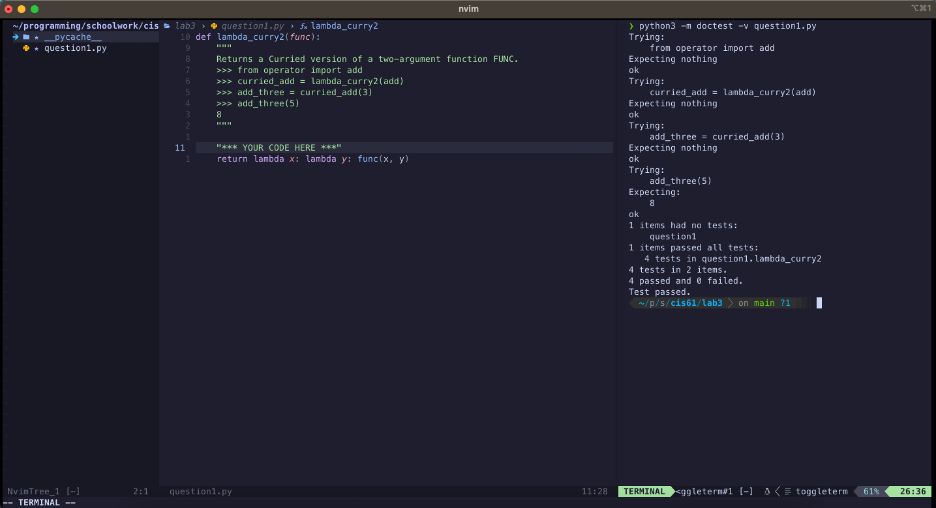
### Question 1: Lambdas and Currying

### We can transform multiple-argument functions into a chain of single-argument, higher order functions by taking advantage of lambda expressions. This is useful when dealing with functions that take only single-argument functions. We will see some examples of these later on.

### Write a function lambda\_curry2 that will curry any two argument functions using lambdas. See the doctest or refer to the [textbook](http://composingprograms.com/pages/16-higher-order-functions.html#currying) if you're not sure what this means.

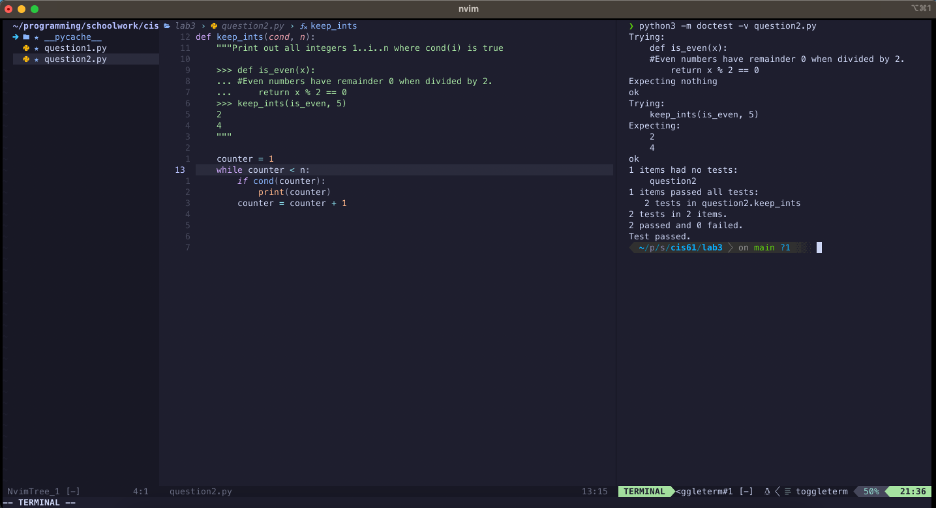
### *Your solution to this problem should fit entirely on the return line.* You can try writing it first without this restriction, but rewrite it after in one line to test your understanding of this topic.

| **def** **lambda\_curry2**(func):  """  Returns a Curried version of a two-argument function FUNC.  >>> from operator import add  >>> curried\_add = lambda\_curry2(add)  >>> add\_three = curried\_add(3)  >>> add\_three(5)  8  """  "\*\*\* YOUR CODE HERE \*\*\*"  **return** \_\_\_\_\_\_ |
| --- |



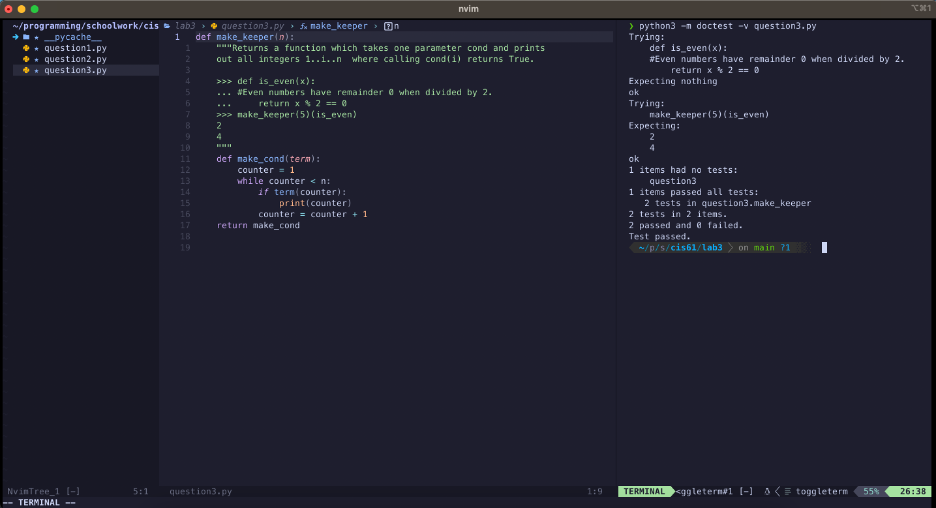
Question 2: Write a function that takes in a function cond and a number n and prints numbers from 1 to n where calling cond on that number returns True.

| def **keep\_ints**(cond, n):  """Print out all integers 1..i..n where cond(i) is true  >>> def is\_even(x):  ... #Even numbers have remainder 0 when divided by 2.  ... return x % 2 == 0  >>> keep\_ints(is\_even, 5)  2  4  """ |
| --- |



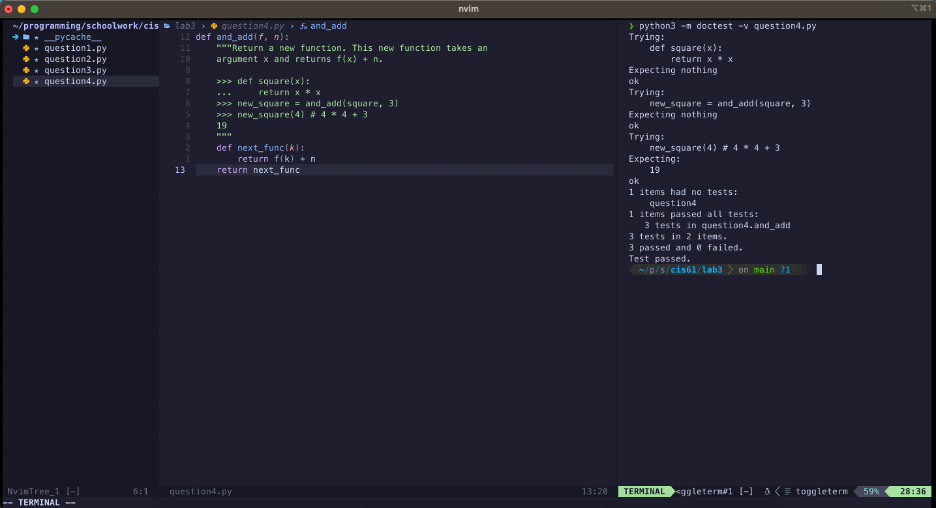
**Question 3:** Write a function similar to keep\_ints like before, but now it takes in a number n and returns a function that has one parameter cond. The returned function prints out numbers from 1 to n where calling cond on that number returns True.

| def **make\_keeper**(n):  """Returns a function which takes one parameter cond and prints  out all integers 1..i..n where calling cond(i) returns True.    >>> def is\_even(x):  ... #Even numbers have remainder 0 when divided by 2.  ... return x % 2 == 0  >>> make\_keeper(5)(is\_even)  2  4  """ |
| --- |



Question 4: Write a function **and\_add** that takes a one-argument function f and a number n as arguments. It should return a function that takes one argument, and does the same thing as the function **f**, except also adds **n** to the result.

| def **and\_add**(f, n):  """Return a new function. This new function takes an  argument x and returns f(x) + n.  >>> def square(x):  ... return x \* x  >>> new\_square = and\_add(square, 3)  >>> new\_square(4) # 4 \* 4 + 3  19  """ |
| --- |

**Question** 5 - Try drawing an environment diagram for the following code and predict what Python will output.

You need to do these problems on paper to develop familiarity with Environment Diagrams. Scan or take a picture of your solution and paste here.

| >>> a = lambda x: x \* 2 + 1  >>> **def** **b**(b, x):  ... return b(x + a(x))  >>> x = 3  >>> b(a, x)  \_\_\_\_\_\_ |
| --- |

