

CIS 61 :: Lab 03 - Higher Order Functions - Template

[DO NOT USE RECURSION, FOR LOOP or LISTS IN ANY OF YOUR SOLUTIONS]

Student Name:

Question 1: Lambdas and Currying

Write a function `lambda_curry2` that will curry any two argument functions using lambdas. See the doctest or refer to the textbook if you're not sure what this means.

```
Question1.py
1 def lambda_curry2(func):
2     """
3     Returns a Curried version of a two-argument function FUNC.
4     """
5     from operator import add
6     >>> curried_add = lambda_curry2(add)
7     >>> add_three = curried_add(3)
8     >>> add_three(5)
9     """
10    return lambda outer: (lambda inner: func(outer, inner))
11
```

```
Command Prompt
C:\Users\Ben\OneDrive\Desktop\CIS Repos\CIS61A\Lab3\Code>python -m doctest Question1.py -v
Trying:
    from operator import add
Expecting nothing
ok
Trying:
    curried_add = lambda_curry2(add)
Expecting nothing
ok
Trying:
    add_three = curried_add(3)
Expecting nothing
ok
Trying:
    add_three(5)
Expecting:
    8
ok
1 item had no tests:
    Question1
1 item passed all tests:
    4 tests in Question1.lambda_curry2
4 tests in 2 items.
4 passed.
Test passed.
```

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`.

```
Question2.py
1 def keep_ints(cond, n):
2     """Print out all integers 1..n where cond(i) is true
3     """
4     >>> def is_even(x):
5         ... #Even numbers have remainder 0 when divided by 2.
6         ... return x % 2 == 0
7     >>> keep_ints(is_even, 5)
8     2
9     4
10    """
11    for i in range(1, n+1):
12        if cond(i):
13            print(i)
```

```
Command Prompt
C:\Users\Ben\OneDrive\Desktop\CIS Repos\CIS61A\Lab3\Code>python -m doctest Question2.py -v
Trying:
    def is_even(x):
        #Even numbers have remainder 0 when divided by 2.
        return x % 2 == 0
Expecting nothing
ok
Trying:
    keep_ints(is_even, 5)
Expecting:
    2
    4
ok
1 item had no tests:
    Question2
1 item passed all tests:
    2 tests in Question2.keep_ints
2 tests in 2 items.
2 passed.
Test passed.
```

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`.

```
Question3.py
1 def make_keeper(n):
2     """Returns a function which takes one parameter cond and prints
3     out all integers 1..n where calling cond(i) returns True.
4     """
5     >>> def is_even(x):
6         ... #Even numbers have remainder 0 when divided by 2.
7         ... return x % 2 == 0
8     >>> make_keeper(5)(is_even)
9     2
10    4
11    """
12    def func(cond):
13        for i in range(1, n+1):
14            if cond(i):
15                print(i)
16    return func
```

```
Command Prompt
C:\Users\Ben\OneDrive\Desktop\CIS Repos\CIS61A\Lab3\Code>python -m doctest Question3.py -v
Trying:
    def is_even(x):
        #Even numbers have remainder 0 when divided by 2.
        return x % 2 == 0
Expecting nothing
ok
Trying:
    make_keeper(5)(is_even)
Expecting:
    2
    4
ok
1 item had no tests:
    Question3
1 item passed all tests:
    2 tests in Question3.make_keeper
2 tests in 2 items.
2 passed.
Test passed.
```

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.

Question4.py

```

1 def and_add(f, n):
2     """Return a new function. This new function takes an
3     argument x and returns f(x) + n.
4     """
5     def square(x):
6         ... return x * x
7     new_square = and_add(square, 3)
8     new_square(4) # 4 * 4 + 3
9     19
10    """
11    return lambda x: f(x) + n

```

Command Prompt

```

C:\Users\Ben\OneDrive\Desktop\CIS Repos\CIS61A\Lab3\Code>python -m doctest Question4.py -v
Trying:
def square(x):
    return x * x
Expecting nothing
ok
Trying:
new_square = and_add(square, 3)
Expecting nothing
ok
Trying:
new_square(4) # 4 * 4 + 3
Expecting:
19
ok
1 item had no tests:
Question4
1 item passed all tests:
3 tests in Question4.and_add
3 tests in 2 items.
3 passed.
Test passed.

C:\Users\Ben\OneDrive\Desktop\CIS Repos\CIS61A\Lab3\Code>

```

Question 5 - Draw 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.

