ENVIRONMENT DIAGRAMS AND HIGHER ORDER FUNCTIONS

COMPUTER SCIENCE MENTORS 61A

September 7–September 9, 2022

1 Environment Diagrams

1. When do we make a new frame in an environment diagram?

2. Give the environment diagram and console output that result from running the following code.

```python
def swap(x, y):
    x, y = y, x
    return print("Swapped!", x, y)

x, y = 60, 1
a = swap(x, y)
swap(a, y)
```

3. Draw the environment diagram that results from running the following code.

```python
def funny(joke):
    hoax = joke + 1
    return funny(hoax)

def sad(joke):
    hoax = joke - 1
    return hoax + hoax

funny, sad = sad, funny
result = funny(sad(2))
```

2   Higher-Order Functions

1. Why and where do we use lambda and higher-order functions?

2. Give the environment diagram and console output that result from running the following code.

```
x = 20
def foo(y):
    x = 5
    def bar():
        return lambda y: x - y
    return bar

y = foo(7)
z = y()
print(z(2))
```

3. Draw the environment diagram that results from running the code.

```
apple = 4
def orange(apple):
    apple = 5
    def plum(x):
        return lambda plum: plum * 2
    return plum

orange(apple)("hiii")(4)
```

4. Fill in the blanks (*without using any numbers in the first blank*) such that the entire expression evaluates to 9.

```
(lambda x: lambda y: _____)(_____)(lambda z: z*z)()
```

5. Write a function, whole_sum, which takes in an integer, n. It returns another function which takes in an integer, and returns True if the digits of that integer sum to n and False otherwise.

```
def whole_sum(n):
    """
    >>> whole_sum(21)(777)
    True
    >>> whole_sum(142)(10010101010)
    False
    """
    def check(x):

        _____

        while _____:

            last = _____

            _____

            _____

        return _____

    return _____
```

6. Write a higher-order function that passes the following doctests.

Challenge: Write the function body in one line.

```python
def mystery(f, x):
    """
    >>> from operator import add, mul
    >>> a = mystery(add, 3)
    >>> a(4) # add(3, 4)
    7
    >>> a(12)
    15
    >>> b = mystery(mul, 5)
    >>> b(7) # mul(5, 7)
    35
    >>> b(1)
    5
    >>> c = mystery(lambda x, y: x * x + y, 4)
    >>> c(5)
    21
    >>> c(7)
    23
    """
```

7. What would Python display?

```python
>>> foo = mystery(lambda a, b: a(b), lambda c: 5 + square(c))
>>> foo(-2)
```