Environment Diagrams and HOFs Solutions

Computer Science Mentors

February 1 - February 3, 2021

1 Environment Diagrams

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

   We make a new frame in an environment diagram when calling a user-defined function, or when we are applying the operator to the operand(s). This occurs after both the operator and operand(s) are evaluated.

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

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

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

   https://tinyurl.com/y68m6qdj

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

```
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))
```

https://tinyurl.com/y5lc4fez

CSM 61A Spring 2021: Ivan Penev and Joshua Baum, with
Jamie Ip, Kenneth Chi, Kevin Moy, Mathilde McKeever, Cyrus Bugwadia, and
Kunal Agarwal, Luke Liu, John So, Jemmy Zhou, Ruslana Yurtyn, Vivian Lu, Matt Au, Lily Yang, Mihira Patel, Jessica Yu, Aidan Tong, Matthew Guo, Nikhita Anumukonda, Laryn Qi, Grace Yi, Jennifer Huang, Cindy Lin, James Fletcher, Sean Sananikone, Abinaya Srikant, Uma Unni

2 Higher-Order Functions

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

   In practice, we use lambda functions and higher-order functions to write short *adapters* programs, or functions that help us connect two programs together.

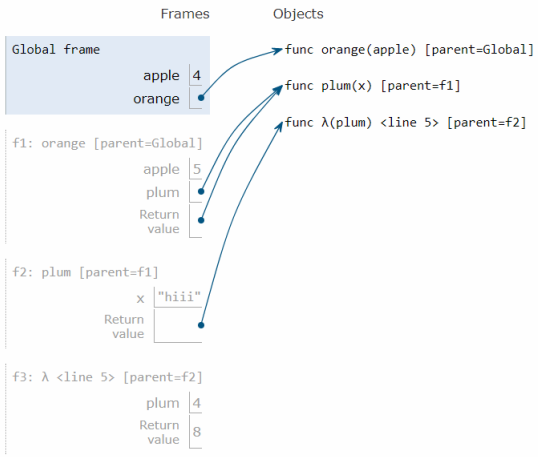
2. Draw the environment diagram that results from running the code.

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

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

   https://tinyurl.com/yxfcvxxa

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)
```



   https://tinyurl.com/y5lo34xb

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: y(x))(3)(lambda z: z*z)()
```

5. Write a function, `print_sum`, that takes in a positive integer, `a`, and returns a function that does the following:
(1) takes in a positive integer, `b`
(2) prints the sum of all natural numbers from 1 to `a*b`
(3) returns a higher-order function that, when called, prints the sum of all natural numbers from 1 to `(a+b)*c`, where `c` is another positive integer.

```
def print_sum(a):
    """
    >>> f = print_sum(1)
    >>> g = f(2) # 1*2 => 1 + 2
    3
    >>> h = g(4) # (1+2)*4 => 1 + 2 + ... + 11 + 12
    78
    >>> i = h(5) # (3+4)*5 => 1 + 2 + ... + 34 + 35
    630
    """
    def helper(b):

        i, total = _____

        while _____:

            _____

            _____

        print(_____)

        return _____

    return _____
```

```
def print_sum(a):
    def helper(b):
        i, total = 0, 0
        while i <= a*b:
            total += i
            i += 1
        print(total)
        return print_sum(a+b)
    return helper
```

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

   Challenge: Write the function body in one line.

```
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
    """

    def helper(y):
        return f(x, y)
    return helper
```

   Challenge solution:

```
    return lambda y : f(x, y)
```

7. What would Python display?

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

9
```

CSM 61A Spring 2021: Ivan Penev and Joshua Baum, with
Jamie Ip, Kenneth Chi, Kevin Moy, Mathilde McKeever, Cyrus Bugwadia, and
Kunal Agarwal, Luke Liu, John So, Jemmy Zhou, Ruslana Yurtyn, Vivian Lu, Matt Au, Lily Yang, Mihira Patel, Jessica Yu, Aidan Tong, Matthew Guo, Nikhita Anumukonda, Laryn Qi, Grace Yi, Jennifer Huang, Cindy Lin, James Fletcher, Sean Sananikone, Abinaya Srikant, Uma Unni