LAMBDAS AND ENVIRONMENT DIAGRAMS 4

COMPUTER SCIENCE 88

September 22, 2021

1 Lambdas

Lambda expressions are one-line functions that specify two things: the parameters and the return expression.

A lambda expression that takes in no arguments and returns 8:

```
lambda: 8
```
(8: return value)

A lambda expression that takes two arguments and returns their product:

```
lambda x, y : x * y
```
(x, y: parameters; x * y: return expression)

Unlike functions created by a `def` statement, the function object that a lambda expression creates has no intrinsic name and is not bound to any variable. In fact, nothing changes in the current environment when we evaluate a lambda expression unless we do something with this expression, such as assign it to a variable or pass it as an argument to a higher order function.

1. What would Python print?

```
>>> a = lambda: 5
>>> a()

>>> a(5)

>>> b = lambda: lambda x: 3
>>> b()(15)

>>> c = lambda x, y: x + y
>>> c(4, 5)

>>> d = lambda x: lambda y: x * y
>>> d(3)

>>> d(3)(3)

>>> e = d(2)
>>> e(5)

>>> f = lambda: print(1)

>>> g = f()
```

2 Environment Diagrams

1. Draw the environment diagram for evaluating the following code

```
def mystery_a(lst):
    def mystery_b(color, count):
        lst.extend([color] * count)
    return mystery_b

colors = ["purple", "pink", "brown"]
f = mystery_a(colors)
f("red", 3)
f("blue", 1)
```

2. If on line 2 and line 4, we replace `mystery_b` with `mystery_a`, what will change in the environment diagram, if anything?

3. If on line 3, we change `lst.extend([color] * count)` to `lst.append([color] * count)`, what will change, if anything?

4. Draw the environment diagram for evaluating the following code

```
def ross(geller, num):
    return geller(monica(num))

def monica(num):
    if num >= 2:
        return tup[0]
    return tup[num]


f = lambda x: x[-1] == "a"
tup = ("hola", "there")
rachel = ross(f, 5)
```

5. Draw the environment diagram for evaluating the following code

```
def anna(olaf):
    return lambda a, b: olaf or [a] * b

hans = [1]
elsa = anna(hans.append(4))
kristoff = elsa(3, 4)
```