Lists and Trees

Computer Science Mentors 61A

February 19 to February 21, 2018

1 Lists

1. Draw box-and-pointer diagrams for the following:

```
>>> a = [1, 2, 3]
>>> a
```

Solution:
```
[1, 2, 3]
```

```
>>> a[2]
```

Solution: 3

```
>>> b = a
>>> a = a + [4, 5]
>>> a
```

Solution:
```
[1, 2, 3, 4, 5]
```

```
>>> b
```

Solution:
```
[1, 2, 3]
```

```
>>> c = a
>>> a = [4, 5]
>>> a
```

Solution:
```
[4, 5]
```

```
>>> c
```

Solution:
```
[1, 2, 3, 4, 5]
```

```
>>> d = c[0:2]
>>> c[0] = 9
>>> d
```

Solution:
```
[1, 2]
```

Solution: Box and pointer diagram in Python Tutor.

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

```
def reverse(lst):
    if len(lst) <= 1:
        return lst
    return reverse(lst[1:]) + [lst[0]]

lst = [1, [2, 3], 4]
rev = reverse(lst)
```

Solution: https://goo.gl/6vPeX9

3. Write a function that takes in a list `nums` and returns a new list with only the primes from `nums`. Assume that `is_prime(n)` is defined. You may use a `while` loop, a `for` loop, or a list comprehension.

```
def all_primes(nums):
```

Solution:

```
    result = []
    for i in nums:
        if is_prime(i):
            result = result + [i]
    return result

    List comprehension:
    return [x for x in nums if is_prime(x)]
```

4. Write a function that takes in a list of positive integers and outputs a list of lists where the i-th list contains the integers from 0 up to, but not including, the i-th element of the input list.

```
def list_of_lists(lst):
"""
>>> list_of_lists([1, 2, 3])
[[0], [0, 1], [0, 1, 2])

>>>list_of_lists([1])
[[0]]

>>>list_of_lists([])
[]
"""
```

Solution:

```
[[x for x in range(y)] for y in lst]
```

Computer Science Mentors CS61A Spring 2018: Chris Allsman and Jennie Chen, with
Ajay Raj, Alex Yang, Annie Tang, Brandon Fong, Catherine Han, Danelle Nachum, Elaine Park, Hyun Jae Moon, Kevin Tsang, Lindsay Yang, Michelle Cheung, Ryan Moughan, Ryan Roggenkemper, Shreya Sahoo, Surya Duggirala, Thomas Zhang

2 Trees

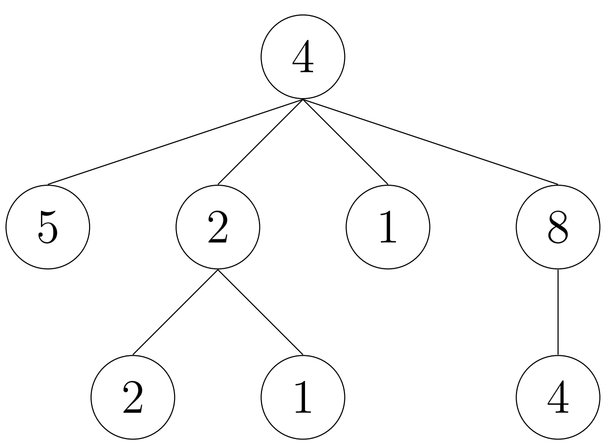
Things to remember:

```
def tree(label, branches=[]):
    return [label] + [branches]

def label(tree):
    return tree[0]

def branches(tree):
    return tree[1:] #returns a list of branches
```

As shown above, the tree constructor takes in a label and a list of branches (which are themselves trees).

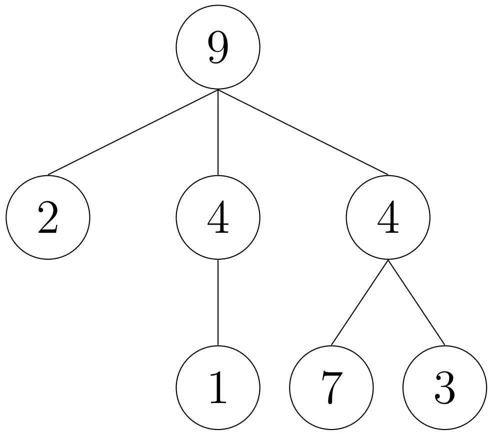
```
tree(4,
    [tree(5, []),
     tree(2,
        [tree(2, []),
         tree(1, [])]),
     tree(1, []),
     tree(8,
        [tree(4, [])])])
```

The above expression constructs a tree that looks like this:



Computer Science Mentors CS61A Spring 2018: Chris Allsman and Jennie Chen, with
Ajay Raj, Alex Yang, Annie Tang, Brandon Fong, Catherine Han, Danelle Nachum, Elaine Park, Hyun Jae Moon, Kevin Tsang, Lindsay Yang, Michelle Cheung, Ryan Moughan, Ryan Roggenkemper, Shreya Sahoo, Surya Duggirala, Thomas Zhang

1. Construct the following tree and save it to the variable `t`.



Solution:
```
t = tree(9, [tree(2, []),
             tree(4, [tree(1, [])]),
             tree(4, [tree(7, []),
                      tree(3, [])])])
```

2. What would this output?
```
>>> label(t)
```

Solution: 9

```
>>> branches(t)[2]
```

Solution:
```
tree(4, [tree(7, []), tree(3, [])])
```

```
>>> branches(branches(t)[2])[0]
```

Solution:
```
tree(7, [])
```

3. Write the Python expression to return the integer 2 from t.

Solution:
```
label(branches(t)[0])
```

4. Write the function `sum_of_nodes` which takes in a tree and outputs the sum of all the elements in the tree.

```
def sum_of_nodes(t):
    """
    >>> t = tree(...) # Tree from question 2.
    >>> sum_of_nodes(t) # 9 + 2 + 4 + 4 + 1 + 7 + 3 = 30
    30
    """
```

Solution:

```
    total = label(t)
    for branch in branches(t):
        total += sum_of_nodes(branch)
    return total

    Alternative solution:
    return label(t) +\
           sum([sum_of_nodes(b) for b in branches(t)])
```

Computer Science Mentors CS61A Spring 2018: Chris Allsman and Jennie Chen, with
Ajay Raj, Alex Yang, Annie Tang, Brandon Fong, Catherine Han, Danelle Nachum, Elaine Park, Hyun Jae Moon, Kevin Tsang, Lindsay Yang, Michelle Cheung, Ryan Moughan, Ryan Roggenkemper, Shreya Sahoo, Surya Duggirala, Thomas Zhang