Recursion and Tree Recursion

Computer Science Mentors 61A

September 18 to September 22, 2017

1 Recursion

Every Recursive function has three things.

1. One or more base cases
2. One or more ways to break the problem down into a smaller problem
   - E.g. Given a number as input, we need to break it down into a smaller number
3. Solve the smaller problem recursively; from that, form a solution to the original problem

1. Complete the definition for `num_digits`, which takes in a number `n` and returns the number of digits it has.

```
def num_digits(n):
    """Takes in an positive integer and returns the number of
    digits.

    >>> num_digits(0)
    1
    >>> num_digits(1)
    1
    >>> num_digits(7)
    1
    >>> num_digits(1093)
    4
    """
```

2. Write a function `is_sorted` that takes in an integer `n` and returns true if the digits of that number are increasing from right to left.

```
def is_sorted(n):
   """
   >>> is_sorted(2)
   True
   >>> is_sorted(22222)
   True
   >>> is_sorted(9876543210)
   True
   >>> is_sorted(9087654321)
   False
   """
```

2 Tree Recursion

3. Mario needs to jump over a series of Piranha plants, represented as a string of 0's and 1's. Mario only moves forward and can either *step* (move forward one space) or *jump* (move forward two spaces) from each position. How many different ways can Mario traverse a level without stepping or jumping into a Piranha plant? Assume that every level begins with a 1 (where Mario starts) and ends with a 1 (where Mario must end up).

```
def mario_number(level):
    """
    Return the number of ways that mario can traverse the
    level where mario can either hop by one digit or two
    digits each turn a level is defined as being an integer
    where a 1 is something mario can step on and 0 is
    something mario cannot step on.
    >>> mario_number(10101)
    1
    >>> mario_number(11101)
    2
    >>> mario_number(100101)
    0
    """
    if ___________:
        _________
    elif ________________:
        _________
    else:
        ___________________________________________________
```

4. Implement the function `make_change`.

```
def make_change(n):
    """Write a function, make_change that takes in an
    integer amount, n, and returns the minimum number
    of coins we can use to make change for that n,
    using 1-cent, 3-cent, and 4-cent coins.
    Look at the doctests for more examples.
    >>> make_change(5)
    2
    >>> make_change(6) # tricky! Not 4 + 1 + 1 but 3 + 3
    2
    """
    if _____________:
        return 0
    elif ______________:
        return 1 + make_change(n - 1)
    elif _______________:
        ____________________________________
        ____________________________________
        return _____________________________
    else:
        ____________________________________
        ____________________________________
        ____________________________________
        return _____________________________
```