1 Recursion

Every Recursive function has three components.

1. One or more base case(s)
2. One or more ways to break the problem down into a smaller problem
3. Solve the smaller problem(s) recursively and combine the results to solve the original problem

1.1 What is wrong with the following function? How can we fix it?

```python
def factorial(n):
    return n * factorial(n)
```

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

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

    >>> num_digits(0)
    1
    >>> num_digits(1)
    1
    >>> num_digits(7)
    1
    >>> num_digits(1093)
    4
    """
```
1.3 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.

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

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

```python
def bar(f):
    def g(x):
        if x == 1:
            return f(x)
        else:
            return f(x) + g(x - 1)
    return g

f = 4
bar(lambda x: x + f)(2)
```
1.5 Write a function that takes as input a number, n, and a list of numbers, lst, and returns true if we can find a subset of lst that sums up to n.

```python
def add_up(n, lst):
    
    >>> add_up(10, [1, 2, 3, 4, 5])
    True
    >>> add_up(8, [1, 2, 3, 4, 5])
    True
    >>> add_up(-1, [1, 2, 3, 4, 5])
    False
    >>> add_up(100, [1, 2, 3, 4, 5])
    False
    
    if ________________________________:
        return True
    
    if lst == []:
        ________________________________
    
    else:
        first, rest = ____________________, ____________
        
        return ________________________________
```