Recursion Exercises

```
def sum_list(L):
    """
    Return the sum of all ints in L.

    @param int|list[int|list[...]] L: possibly-nested list of ints, finite depth

    >>> sum_list([1, [2, 3], [4, 5, [6, 7], 8]])
    36
    >>> sum([])
    0
    """

    if isinstance(L, list):
        return sum([sum_list(x) for x in L])
    else:
        return L
```

1. What helper methods does this function call?

2. So far, we haven't confirmed that the function works in any cases. Trace this call: sum_list(27)

3. Complete the following trace of this call: sum_list([4, 1, 8])

   ```
   sum_list([4, 1, 8]) --> sum( [  sum_list(4),  sum_list(1),  sum_list(8)  ] )
                       --> sum( [
                       -->
   ```

4. Trace this call: sum_list([4])

5. Trace this call: sum_list([])

6. Trace this call: sum_list([4, [1, 2, 3], 8])

7. Trace this call: sum_list([[1, 2, 3], [4, 5], 8])

8. Trace this call: sum_list([1, [2, 2], [2, [3, 3, 3], 2]])

9. Trace this call: sum_list([1, [2, 2], [2, [3, [4, 4], 3, 3], 2]])

10. Are you a believer yet?