In [30]:
```
def list_flatten(lst):
    """
    Function that uses list comp to flatten a 2-dimensional list.

    Args:
        lst(list): 2-dimensional list of values
    Returns:
        a 1-dimensional list.

    >>> list_flatten([[1], [2,3], [4,5,6]])
    [1, 2, 3, 4, 5, 6]
    """
    return [val for sub_lst in lst for val in sub_lst]

list_flatten([[1], [2,3], [4,5,6]])
```

Out[30]:
```
[1, 2, 3, 4, 5, 6]
```

In [51]:

```
def dataframe_at_home(filepath):
    """
    Function that ingests a CSV file and outputs a dictionary where
    keys are column names and values are lists for that column.

    Args:
        filepath(str): string for filepath reference
    Returns:
        a dictionary with values populated according to CSV

    >>> dataframe_at_home('files/data.csv')
    {'Age': [10, 100, 8, 15, 22], 'Grade': [5, None, 3, 10, 16]}
    """
    output = {}
    with open(filepath, 'r') as f:
        column_names = f.readline().strip().split(',')
        data = f.readlines()

    for i in range(len(column_names)):
        column_data = [eval(x.strip().split(',')[i]) for x in data]
        output[column_names[i]] = column_data
    return output

dataframe_at_home('files/data.csv')
```

Out[51]:

```
{'Age': [10, 100, 8, 15, 22], 'Grade': [5, None, 3, 10, 16]}
```

In [52]:

```
def dataframe_describe_at_home(lst):
    """
    Function that calculates basic summary statistics of
    a given set of numbers.

    Args:
        lst(list): list of numeric values.
    Returns:
        a dictionary with statistics as keys and results
        as associated values.

    >>> dataframe_describe_at_home([10, 100, 8, 15, 22])
    {'mean': 31.0, 'median': 15, 'min': 8, 'max': 100}
    """
    mean = sum(lst)/len(lst)
    lst.sort()
    min_val = lst[0]
    max_val = lst[-1]
    if len(lst)%2 == 1:
        median = lst[len(lst)//2]
    else:
        median = (lst[(len(lst)//2)-1] + lst[(len(lst)//2)]) / 2

    return {'mean':mean,'median':median,'min':min_val,'max':max_val}
dataframe_describe_at_home([10, 100, 8, 15, 22])
```

Out[52]:

```
{'mean': 31.0, 'median': 15, 'min': 8, 'max': 100}
```

In [56]:

```
def check_input_describe(inputs):
    """
    Function that validates the inputs of dataframe_describe_at_home.

    Args:
        inputs (unknown): random inputs to describe
    Returns:
        True (if all asserts pass)
    Raises:
        AssertionError if the input is not valid for dataframe_describe
    >>> check_input_describe([10, 100, 8, 15, 22])
    True
    """
    assert isinstance(inputs, list)
    assert all([isinstance(x, int) or isinstance(x,float) \
                for x in inputs])
    return True

check_input_describe([10, 100, 8, 15, 22])
```

Out[56]:

```
True
```