**TEST 1**

**FAILED RED**

def test\_author\_1(setup\_data):  
 #Test only surnames  
 (Surname, FirstNames) = bibtex.extract\_author(setup\_data['simple\_author\_1'])  
 assert (Surname, FirstNames) == ('Smith', '')  
  
 (Surname, FirstNames) = bibtex.extract\_author(setup\_data['simple\_author\_2'])  
 assert (Surname, FirstNames) == ('Jones', '')

**PASSED GREEN**

def extract\_author(str):  
 parts = str.split()  
 if len(parts) == 1: #We only got the surname  
 return parts[0], ''

**TEST 2**

**FAILED RED**

def test\_author\_2(setup\_data):  
 #Test simple firstname author  
 (Surname, First) = bibtex.extract\_author(setup\_data['author\_1'])  
 assert (Surname, First) == ("Smith", "John")  
  
 (Surname, First) = bibtex.extract\_author(setup\_data['author\_2'])  
 assert (Surname, First) == ("Jones", "Bob")

**PASSED GREEN**

def extract\_author(str):  
 parts = str.split()  
 if len(parts) == 1: #We only got the surname  
 return parts[0], ''  
 elif len(parts) == 2: #We got both surname and name  
 return parts[-1], parts[0]

**TEST 3**

**FAILED RED**

def test\_author\_3(setup\_data):  
 (Surname, First) = bibtex.extract\_author(setup\_data['author\_3'])  
 assert (Surname, First) == ("Pearson", "Justin Kenneth")

**PASSED GREEN**

def extract\_author(str):  
 parts = str.split()  
 if len(parts) == 1: #We only got the surname  
 return parts[0], ''  
 elif len(parts) == 2: #We got both surname and name  
 return parts[-1], parts[0]  
 else: #Surname or name have multiple parts  
 return parts[-1], ' '.join(parts[:-1])

**TEST 4**

**FAILED RED**

def test\_surname\_first(setup\_data):  
 (Surname, First) = bibtex.extract\_author(setup\_data['surname\_first\_1'])  
 assert (Surname, First) == ("Pearson", "Justin Kenneth")  
  
 (Surname, First) = bibtex.extract\_author(setup\_data['surname\_first\_2'])  
 assert (Surname, First) == ("Van Hentenryck", "Pascal")

**PASSED GREEN**

def extract\_author(str):  
 parts = str.split(', ')  
 if len(parts) == 1: #Only one element so it runs normally  
 parts = str.split()  
 if len(parts) == 1: #We only got the surname  
 return parts[0], ''  
 elif len(parts) == 2: #We got both surname and name  
 return parts[-1], parts[0]  
 else: #Surname or name have multiple parts  
 return parts[-1], ' '.join(parts[:-1])  
 elif len(parts) == 2: #There is a comma so we separate name and surname  
 return parts[0], parts[1]

**TEST 5**

**FAILED RED**

def test\_multiple\_authors(setup\_data):  
 Authors = bibtex.extract\_authors(setup\_data['multiple\_authors\_1'])  
 assert Authors[0] == ('Pearson', 'Justin')  
 assert Authors[1] == ('Jones', 'Bob')

**PASSED GREEN**

def extract\_authors(str):  
 list\_of\_authors = str.split(' and ')  
 result = []  
 for x in list\_of\_authors:  
 parts = x.split(', ')  
 if len(parts) == 1: #Only one element so it runs normally  
 parts = [part.strip() for part in x.split()]  
 if len(parts) == 1: #We only got the surname  
 result.append((parts[0], ''))  
 elif len(parts) == 2: #We got both surname and name  
 result.append((parts[-1], parts[0]))  
 else: #Surname or name have multiple parts  
 result.append((parts[-1], ' '.join(parts[:-1])))  
 elif len(parts) == 2: #There is a comma so we separate name and surname  
 result.append((parts[0].strip(), parts[1].strip()))  
 return result