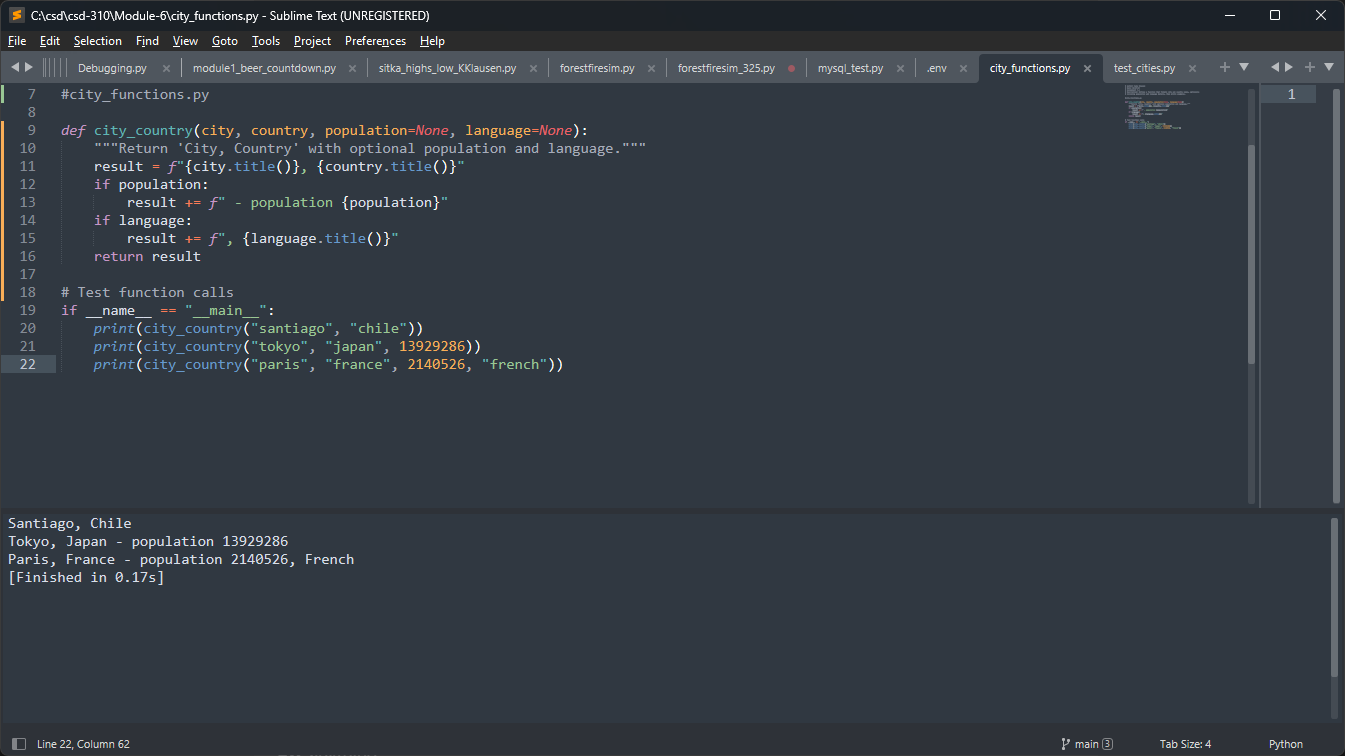
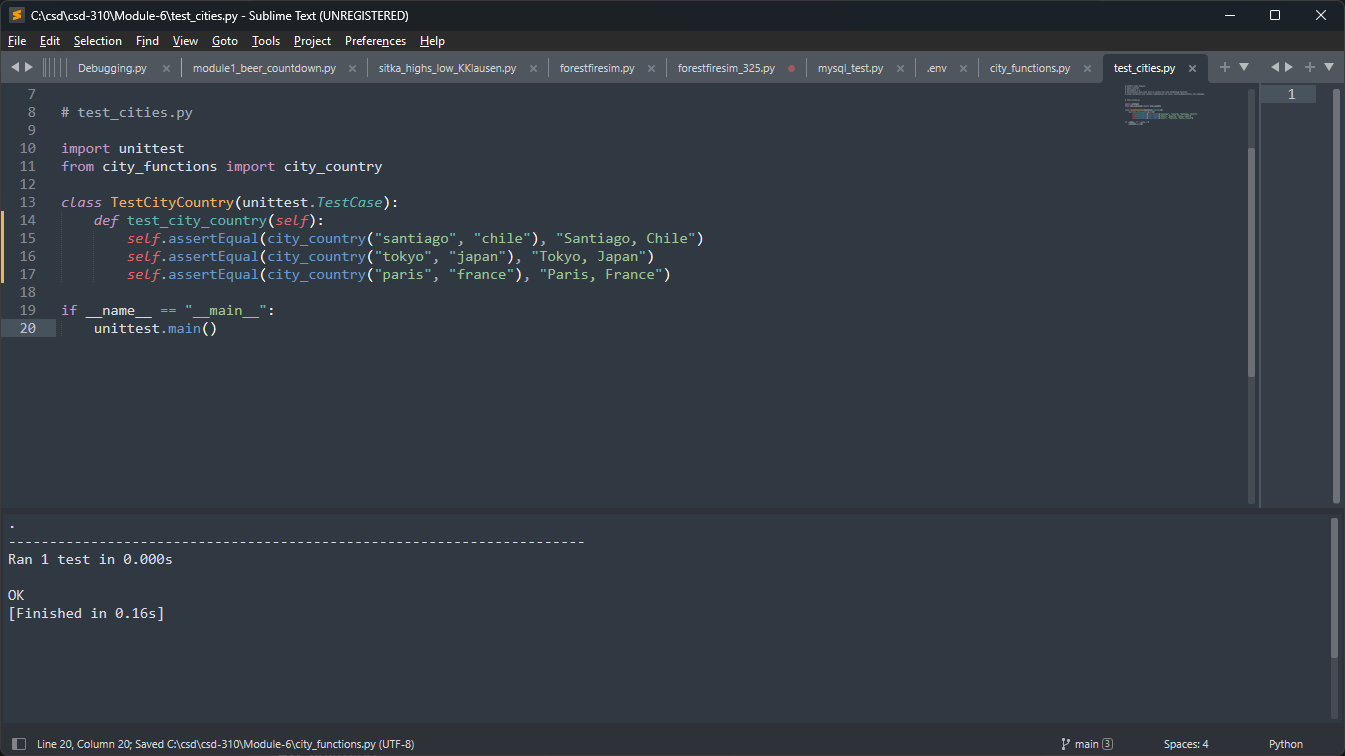
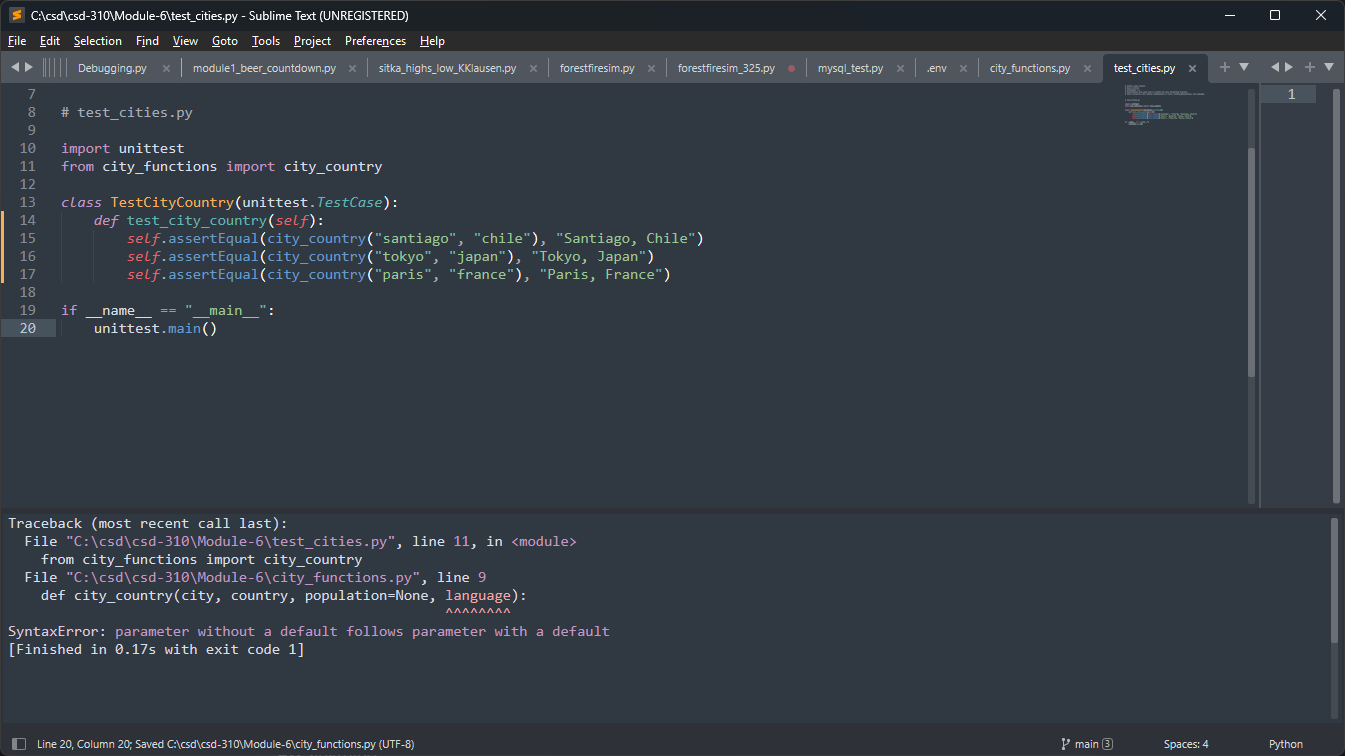
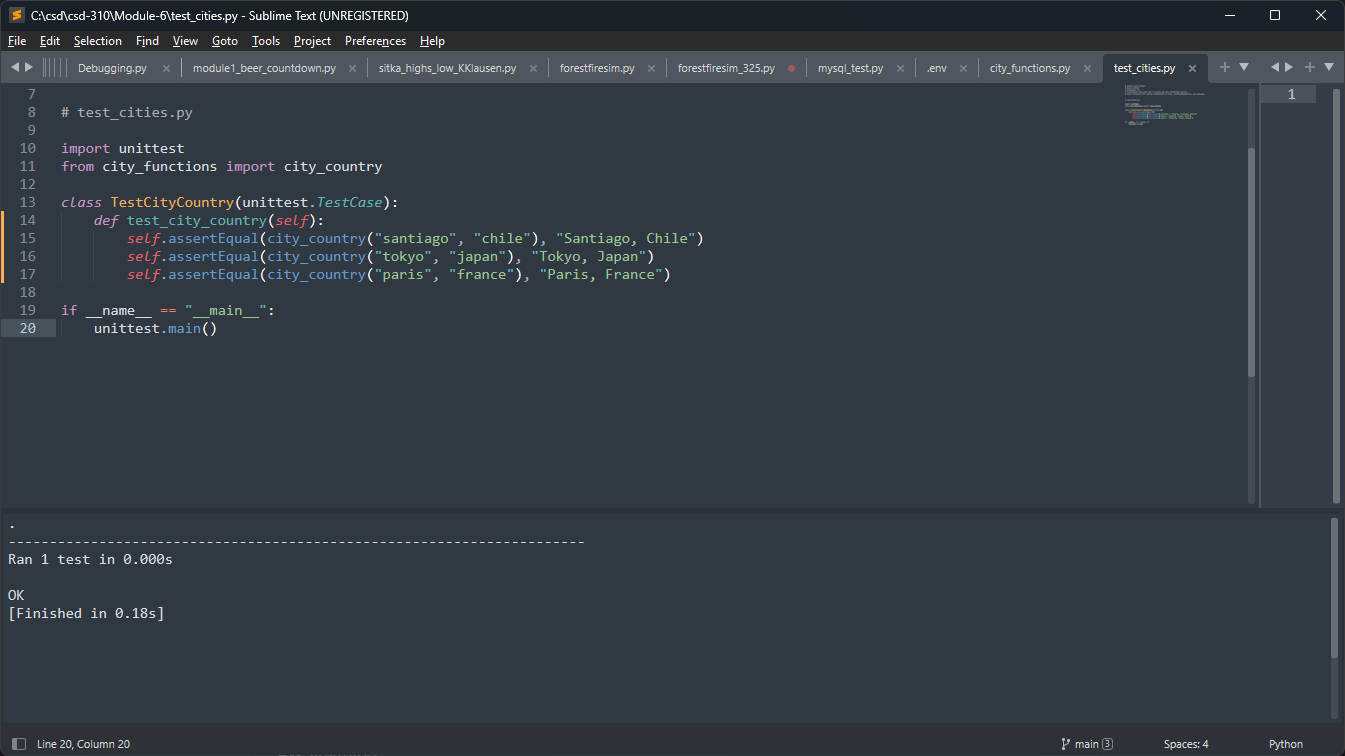
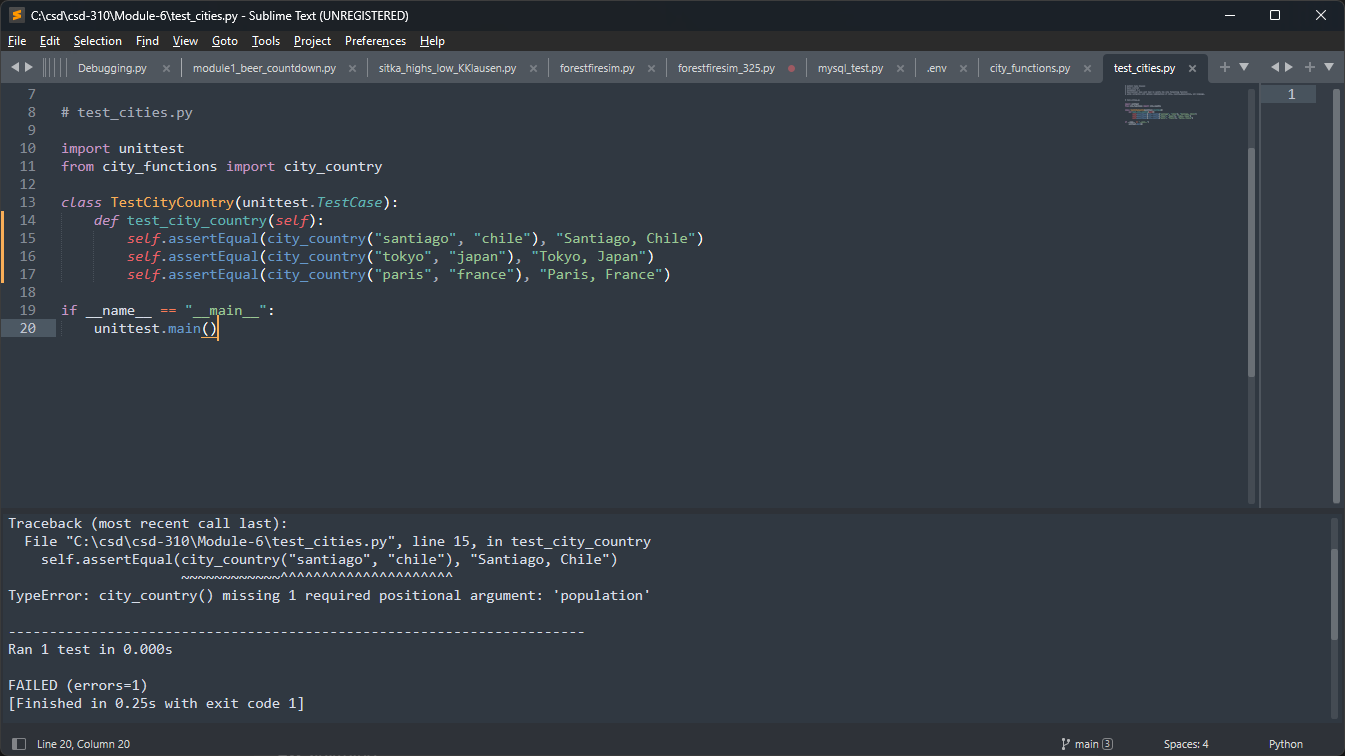
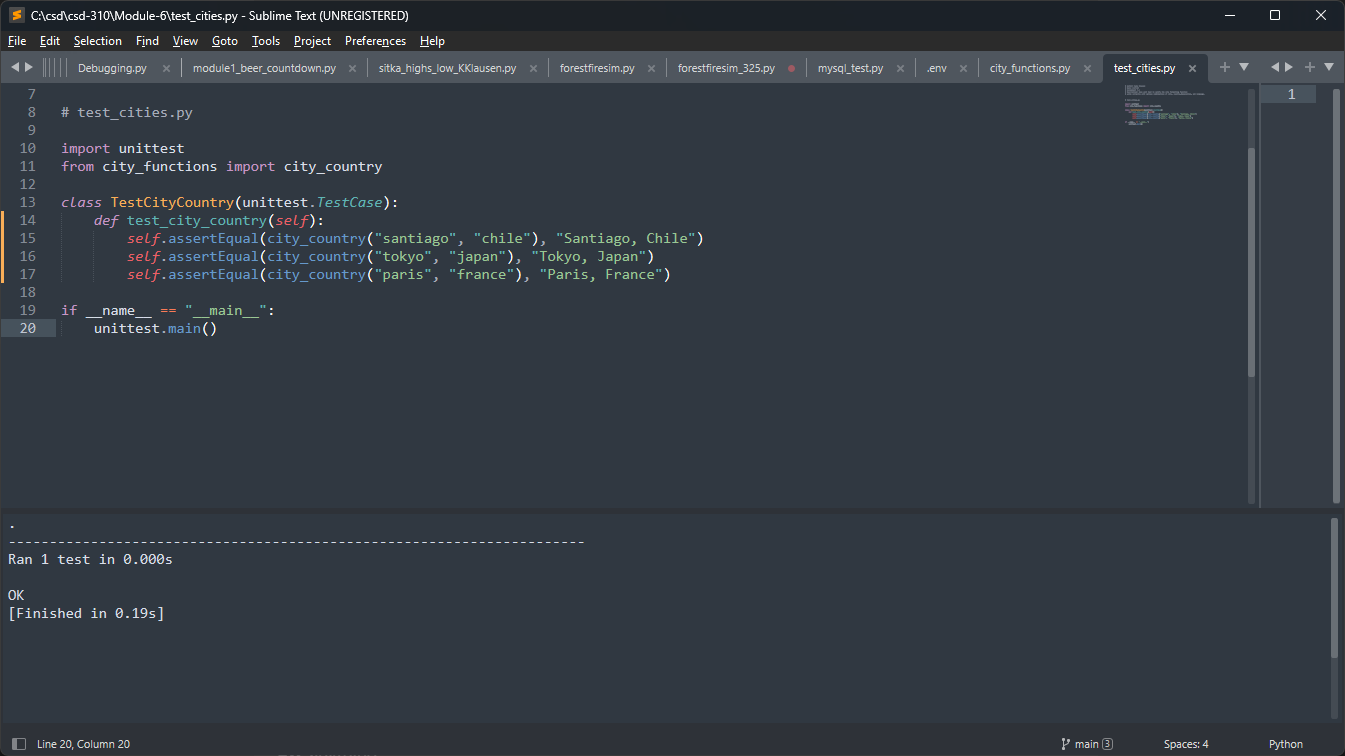
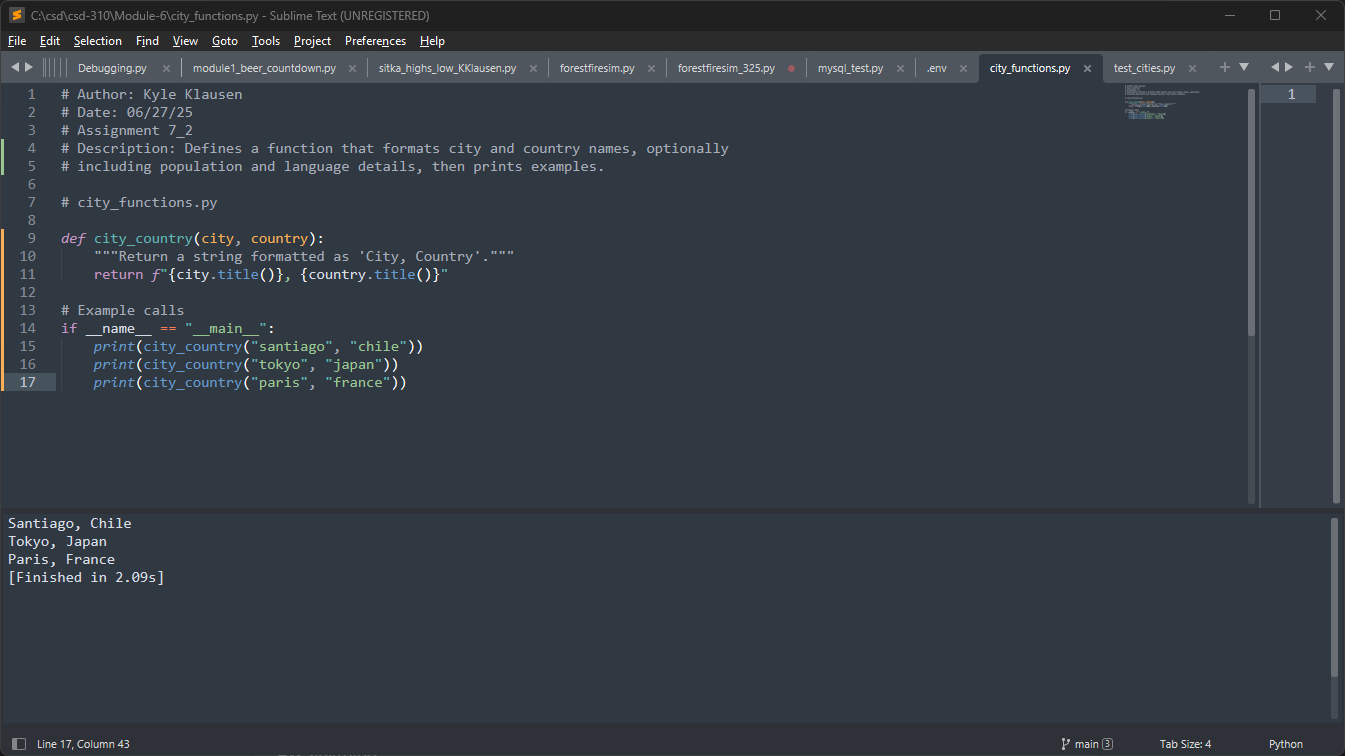
Kyle Klausen

CSD-325

Assignment7\_2

06/27/25



Python code for both programs:

City\_functions:

# Author: Kyle Klausen

# Date: 06/27/25

# Assignment 7\_2

# Description: Defines a function that formats city and country names, optionally

# including population and language details, then prints examples.

#city\_functions.py

def city\_country(city, country, population=None, language=None):

"""Return 'City, Country' with optional population and language."""

result = f"{city.title()}, {country.title()}"

if population:

result += f" - population {population}"

if language:

result += f", {language.title()}"

return result

# Test function calls

if \_\_**name\_\_** == "\_\_**main\_\_**":

print(city\_country("santiago", "chile"))

print(city\_country("tokyo", "japan", 13929286))

print(city\_country("paris", "france", 2140526, "french"))

Test\_cities code:

# Author: Kyle Klausen

# Date: 6/27/25

# Assignment 7\_2

# Description: Uses unit test to verify the city formatting function

# works correctly with various combinations of city, country,#population, and language.

# test\_cities.py

import unittest

from city\_functions import city\_country

class TestCityCountry(unittest.TestCase):

def test\_city\_country(self):

self.assertEqual(city\_country("santiago", "chile"), "Santiago, Chile")

self.assertEqual(city\_country("tokyo", "japan"), "Tokyo, Japan")

self.assertEqual(city\_country("paris", "france"), "Paris, France")

if \_\_**name\_\_** == "\_\_**main\_\_**":

unittest.main()