Caso de Prueba Unitaria 1 (Caso de éxito):

```
import unittest
from Metodos_distancias import DistanceCalculatorFactory

#CASO DE ÉXITO

def test_CSV_method(self):
    distance = DistanceCalculatorFactory.create_distance_calculator("CSV", "Kolkata", "India", "Lagos", "Nigeria")
    self.assertAlmostEqual(distance, 9219.74141992384, places=2)

def test_API_method(self):
    distance = DistanceCalculatorFactory.create_distance_calculator("API", "Kolkata", "India", "Lagos", "Nigeria")
    self.assertAlmostEqual(distance, 9218.040829550899, places=2)

def test_MOOK_method(self):
    distance = DistanceCalculatorFactory.create_distance_calculator("MOOK", "Kolkata", "India", "Lagos", "Nigeria")
    self.assertAlmostEqual(distance, 9219.74141992384, places=2)
```

Se observa el script para el código de test 1. Aquí se considera el caso de éxito para calcular distancias con la forma CSV, API Y MOCK entre las mismas ciudades. Como ciudad 1 se ingresa Kolkata, como país 1 está India, y como ciudad 2, Lagos en el país Nigeria. Los datos de ciudades y países son iguales para todos los formatos.

Caso de Prueba Unitaria 2 (Caso extremo - una de las ciudades no existe) :

```
#CASO EXTREMO 1:

class TestDistanceCalculator2(unittest.TestCase):

def test_nonexistent_city_csv(self):

# Test when a city does not exist in the CSV data

result = DistanceCalculatorFactory.create_distance_calculator("CSV", "Lima123", "Peru", "Quito", "Ecuador")

self.assertEqual(result, "No se encuentra la ciudad en la base de datos")

def test_nonexistent_city_api(self):

# Test when a city does not exist in the Nominatim API

result = DistanceCalculatorFactory.create_distance_calculator("API", "Lima123", "Peru", "Quito", "Ecuador")

self.assertEqual(result, "No se encuentra la ciudad en la base de datos")
```

Se puede observar el script para el código de test 2 con CSV y la API.

Donde el input de Ciudad 1 es

"Lima123" y como se sabe, no existe esa ciudad en el CSV ni en la API

Caso de Prueba Unitaria 3 (Caso extremo - entregar la misma ciudad dos veces):

A continuación se inserta el script para el código de test. Aquí se considera el caso de extremo 2.

```
#CASO EXTREMO 2:

#CASO EXTREMO 2:

# Class TestDistanceCalculator3(unittest.TestCase):

# Test when both country and citys repeated

# result = DistanceCalculatorFactory.create_distance_calculator("CSV", "Kolkata", "India", "Kolkata", "India")

# self.assertEqual(result, "Las ciudades son iguales")

# Test when both country and citys repeated

# Test when both country and citys repeated

# result = DistanceCalculatorFactory.create_distance_calculator("API", "Kolkata", "India", "Kolkata", "India")

# result = DistanceCalculatorFactory.create_distance_calculator("API", "Kolkata", "India")
```

En la imagen se puede observar el script para el código de test con CSV y la API. Donde el input de Ciudad 1 es "Kolkata" y el input del país 1 es "India", de igual forma para la Ciudad 2 y país 2 respectivamente.

Resultados de los test unitarios:

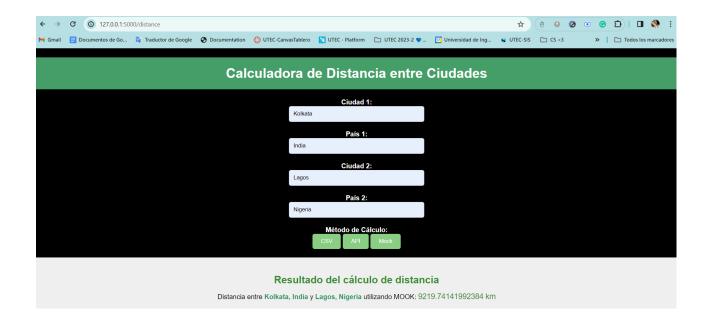
Todos nuestros tests están correctos. Por lo tanto, retorna los siguientes mensajes:

Casos de Prueba Manual:

| Test Case | Precondition | Test Steps | Test Data | Expected Result |
|---|---|---|---|---|
| 1.Verificar que las ciudades ingresadas pertenezcan a sus países respectivos y que la distancia entre ellos sea la misma con la impuesta en el código en cualquiera de los tests para el cálculo de distancia | Que el usuario tenga una conexión de red y que haya clonado nuestro repositorio | Ejecutar la aplicación Ingresa"Kolkata" en la primera casilla de ciudad. Ingresa "India" en la primera casilla del país. Ingresa "Lagos" en la segunda casilla de ciudad. Ingresa "Nigeria" en la segunda casilla de país. Haz clic en el botón "CSV", "MOCK" o "API". | Ciudad 1: Kolkata País 1: India Ciudad 2: Lagos País 2: Nigeria Opción para calcular distancia: CSV, MOCK o API | El usuario deberá observar la distancia en km entre estas ciudades que se encuentran en la data. - Si escoge la opción "CSV", la distancia debe ser: 9219.74141992384 km - Si escoge la opción "MOCK" la distancia debe ser: 9219.74141992384 km - Si escoge la opción "API" la distancia debe ser: 9218.040829550899 km Caso de éxito. |
| 2. Verificar que ambas ciudades con sus respectivos países exitan en los distintos tests para el cálculo de distancia | Que el usuario tenga una conexión de red y que haya clonado nuestro repositorio | Ejecutar la aplicación Ingresa "Lima123" en la primera casilla de ciudad. Ingresa "Peru" en la primera casilla del país. Ingresa "Quito" en la segunda casilla de ciudad. Ingresa "Ecuador" en la segunda casilla de país. Haz clic en el botón "CSV" o | Ciudad 1: Lima123 País 1: Peru Ciudad 2: Quito País 2: Ecuador Opción para calcular distancia: CSV o API | El usuario deberá recibir un mensaje de error para que complete nuevamente los inputs respectivos porque no existen en los distintos tests para el cálculo de distancia. Caso extremo. |

| | | "АРІ". | | |
|--|---|--|---|---|
| 3.Verificar que ambas ciudades con sus respectivos países no se repitan. | Que el usuario tenga una conexión de red y que haya clonado nuestro repositorio | Ejecutar la aplicación Ingresa "Kolkata" en la primera casilla de ciudad. Ingresa "India" en la primera casilla del país. Ingresa "Kolkata" en la segunda casilla de ciudad. Ingresa "India" en la segunda casilla de país. Haz clic en el botón "CSV" o "API". | Ciudad 1: Kolkata País 1: India Ciudad 2: Kolkata País 2: India Opción para calcular distancia: CSV o API | El usuario deberá recibir un mensaje de error para que complete nuevamente los inputs respectivos ya que no deberían repetirse. Caso extremo. |

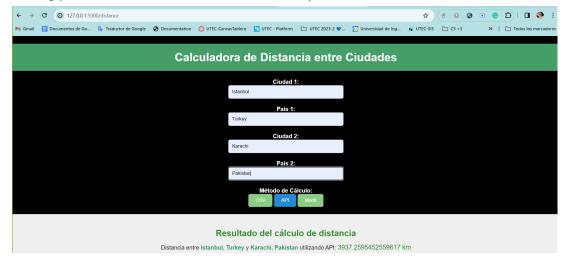
Testing para calcular distancias entre Kolkata y Lagos con Mock:



Testing para calcular distancias entre Istanbul y Karachi con Mock:



Testing para calcular distancias entre Istanbul y Karachi con API:



Testing para calcular distancias entre Kolkata y Lagos con API:



Testing para calcular distancias entre Kolkata y Lagos con CSV:



Testing para calcular distancias entre Istanbul y Karachi con CSV:

