Homework

Name: Alperen Şahin ID: 21806955

> I have not received or given any aid in this homework. All the work presented below is my own work.
> Alperen Şahin

Throughout this project, I use pylint as linter and poetry as package management and packaging tool. And also I adopt TDD to get better result and git for version controlling thus, you can find all this code in GitHub.

Question 1

1. Install poetry
2. Install all the dependencies by `poetry install --no-root`
3. You can either run the unit tests by `poetry run pytest test` or you can run the main script to test it manually by `poetry run start`

Development

1. Initializing The Project

I initialized the project by `poetry new python`. It initializes the project with with a skeleton.

2. Converter Module

Then I created two functions,

```python
def celsius_to_fahrenheit(temp):
    '''
    Converts celsius to fahrenheit
    :param float temp: Fahrenheit in degrees
    '''
    return round(temp * 9 / 5 + 32, 4)
```

```python
def fahrenheit_to_celsius(temp):
    '''
    Converts fahrenheit to celsius
    :param float temp: Celsius in degrees
    '''
    return round((temp - 32) * 5 / 9, 4)
```

These functions are used for converting temperatures. I added these docstring as brief explanations.

3. Tests

```
def test_converter():
    '''
    Unit test for fahrenheit_to_celsius function
    '''
    fahrenheit_values = [-40.0, 0.0, 32.0, 68.0, 98.6, 212.0]
    celsius_values = [-40.0, -17.7778, 0.0, 20.0, 37.0, 100.0]

    for i in range(len(fahrenheit_values)):
        fahrenheit_value = fahrenheit_values[i]
        celsius_value = celsius_values[i]
        assert fahrenheit_to_celsius(fahrenheit_value) == celsius_value
```

This test basically compares the function output with pre-defined expected values.

After running the `poetry run pytest tests`

```
2 passed in 0.01s
```

Success!

4. Main

```
'''
Main script
'''
from converter import fahrenheit_to_celsius
def main():
    '''
    main function
    '''
    fahrenheit_values = [-40.0, 0.0, 32.0, 68.0, 98.6, 212.0]
    celsius_values = [-40.0, -17.7778, 0.0, 20.0, 37.0, 100.0]
    # pylint: disable=consider-using-enumerate, line-too-long
    for i in range(len(fahrenheit_values)):
        fahrenheit_value = fahrenheit_values[i]
        celsius_value = celsius_values[i]
        print(f"Expected value: {fahrenheit_to_celsius(fahrenheit_value)}
-- Output value: {celsius_value}")

if __name__ == "__main__":
    main()
```

You can run the main file by `poetry run start` And observe the output.

Result

1. Semantic versioning
2. Linters
3. Python modules
4. Project structers

Question 2

1. Install poetry
2. Install all the dependencies by `poetry install --no-root`
3. You can either run the unit tests by `poetry run pytest test` or you can run the main script to test it manually by `poetry run start`

Development

1. Initializing The Project

I initialized the project by `poetry new python`. It initializes the project with with a skeleton.

2. Machine Module

Machine module consists of 3 different file which are math, converter and logger.

> As converter module is same with the question 1 i didn't mention it.

Math Module

This module has two functions called square and abs

```
'''
Redefined math module
'''
# pylint: disable = redefined-builtin, invalid-name
def square(x):
    '''
    Returns square of number
    :param float x: Number
    '''
    return x * x

def abs(x):
    '''
    Returns the absolute of number
    :param float x: Number
    '''
    if x < 0:
        return -x
    # This is a fallback actually, if we don't enter the if it will return
the x
    return x
```

> There is also built-in math module

Logger Module

This module only has 1 function

```
'''
Logger module
'''
from .converter import fahr_to_cent

def print_fahr_to_cent(fahr):
    '''
    Converts fahrenheit to celsius and logs it
    :param float fahr: Fahrenheit in degrees
    '''
    result = fahr_to_cent(fahr)
    print(result)
```

Main

This is the main file.

```
# pylint: disable = invalid-name, unused-variable
'''
Main script
'''
from machine import math
from machine import fahr_to_cent
from machine import cent_to_fahr
from machine import print_fahr_to_cent

def main():
    '''
    main function
    '''
    x = 42
    result = math.square(3) + math.square(4)
    print(x)
    boiling = fahr_to_cent(212)
    cold = cent_to_fahr(-40)
    print(result)
    print(math.abs(-22))
    print_fahr_to_cent(32)

if __name__ == "__main__":
    main()
```

> I didn't write tests for this part as mocking is not something that i should consider right now.

Question 3

> I used Google Colab for this part as it is easy and fast

Google Colab

Result

1. Google Colab is cool and fast