Simeon Patton

OSU CS362 – Spring 2021

Extra credit assignment

Unittest and Pytest on two programs and user stories for the social media platform.

The repo for this assignment can be found at:

https://github.com/pattons-OSU/CS362_ExtraCredit

i) Write a program that reverses a sentence and then write tests in both unittest and pytest.

a) As seen below, the program takes input from the user and outputs the sentence in reverse WORD format.

```python
#! python3

"""
Simeon Patton
CS362 OSU Spring 2021
Extra Credit 1.1 - Reverse sentence program

--Write a program that reverses a sentence. Input - ask the user for a sentence:
a long string containing multiple words. Return to the user the same string,
with the words in backward order.

"""

def user_input():
    string = input("\nPlease enter a sentence that you would like reversed:\n\n")
    string = str(string)
    return string


def reverse_string(user_input):
    ## Breaking the string into individual words using
    ## a space as the delimiter
    split_words = user_input.split(' ')
    ## Reversing the list of strings
    reversed_string = split_words[::-1]
    return reversed_string

def merge_reversed(reversed_string):
    word_list = reversed_string
    deliminator = ' '
    ## Using .join method to concantinate into a new string
    merged_string = deliminator.join(word_list)
    return merged_string


if __name__ == '__main__':
    string_input = user_input()
    separated_reversed_list = reverse_string(string_input)
    concantinated_reversed_list = merge_reversed(separated_reversed_list)
    print("\nYour reversed sentence is:\n\n")
    print(concantinated_reversed_list)
    print("\n")
```

b) Unittest Code and Output for reverse sentence

```python
#! python3

"""
Simeon Patton
CS362 OSU Spring 2021
Extra Credit 1.2.1 - Unittest the Reverse sentence program

--Write tests for the reverse_sentence program using unittest and pytest.
(do not run the tests with pytest alone - you should use pytest syntax as well.)

"""

import unittest
import reverse_sentence

## Setting global variables
user_input = reverse_sentence.user_input()
reversed_input = reverse_sentence.reverse_string(user_input)

class testCase(unittest.TestCase):
    ## Testing to make sure that the user is passing through a
    ## String that is capable of being reversed.
    def test_input_type(self):
        self.assertTrue(type(user_input), str)
    ## Testing to make sure that the reversing function is actually
    ## doing the correct function.
    def test_reversing(self):
        self.assertEqual(user_input.split(' ')[::-1], reversed_input)


if __name__ == '__main__':
    unittest.main()
```

```
PS C:\Users\sup_u\Documents\College\CS362\On Git\CS362_ExtraCredit> python .\test_unittest_reverse_sentence.py

Please enter a sentence that you would like reversed:

Hello how are you doing today
..
----------------------------------------------------------------------
Ran 2 tests in 0.001s

OK
PS C:\Users\sup_u\Documents\College\CS362\On Git\CS362_ExtraCredit>
```

c) Pytest Code and output for reverse sentence

***Had to use the “-s” flag to capture user input

```python
#! python3

"""
Simeon Patton
CS362 OSU Spring 2021
Extra Credit 1.2.2 - Pytest the Reverse sentence program

--Write tests for the reverse_sentence program using unittest and pytest.
(do not run the tests with pytest alone - you should use pytest syntax as well.)

"""

import pytest
import reverse_sentence

## Could not get the pytest fixtures working correctly.
## as of this moment, you need to run pytest with the -s
## flag in order to obtain user input.
"""
@pytest.fixture
def user_input():
    user_input = reverse_sentence.user_input()
    return user_input

@pytest.fixture
def reversed_input():
    reversed_input = reverse_sentence.reverse_string(user_input)
    return reversed_input
"""

## Setting global variables
user_input = reverse_sentence.user_input()
reversed_input = reverse_sentence.reverse_string(user_input)

def test_input_type():
    assert type(user_input) == str
def test_reversing():
    assert user_input.split(' ')[::-1] == reversed_input
```

```
PS C:\Users\sup_u\Documents\College\CS362\On Git\CS362_ExtraCredit> pytest -s .\test_pytest_reverse_sentence.py
============================= test session starts =============================
platform win32 -- Python 3.9.0, pytest-6.2.4, py-1.10.0, pluggy-0.13.1
rootdir: C:\Users\sup_u\Documents\College\CS362\On Git\CS362_ExtraCredit
plugins: cov-2.12.0
collecting ...
Please enter a sentence that you would like reversed:

Hello how are you doing today
collected 2 items

test_pytest_reverse_sentence.py ..

============================== 2 passed in 9.45s ==============================
PS C:\Users\sup_u\Documents\College\CS362\On Git\CS362_ExtraCredit>
```

ii) Question 2:

Five user stories from the social media platform in assignment 1;

- As a user I want to be able to post updates to a friend viewable wall in order to share my experiences.
- As a user I want to be able to message other users in order to keep in touch.
- As a user I would like to upload my photos to share them with friends and family.
- As a user I would like to organize my photos into specific albums in order to group like ideas.
- As a user I would like to be notified when another user sends me a message through the messaging system.

iii) Question 3:

Write a program that takes in a target sum and array of values and determines the pair of values that reaches the target sum. Then write unittest and pytest programs for it.

a) Target_sum.py program

```python
#! python3

"""
Simeon Patton
CS362 OSU Spring 2021
Extra Credit 3 - Array and target sum

--Write a program that takes an array of integers - array and an integer target_sum.
Your program should return an array containing two numbers that add up to the target.

    inspiration for array iteration can be found at https://tinyurl.com/cryw3wnj
"""

import math

def array_size():
    ## Forcing array to be smaller than 10 for ease.
    while 1:
        array_size = input("\nHow big would you like the array?\n")
        array_size = int(array_size)

        if array_size <= 10:
            break

        if array_size > 10:
            print("Please enter an array size under 11.")

    return array_size

def target_sum():
    target_sum = input("\nWhat would you like the target sum to be?\n")
    target_sum = int(target_sum)
    return target_sum


def add_to_array(array_size):
    num_array = []

    for i in range(0, array_size):
        add_to_array = input("\nPlease enter a number to add to the array:\n")
        add_to_array = int(add_to_array)
        num_array.append(add_to_array)

    return num_array


def pair_values(num_array, arr_size, sum):

    ## Creating empty list and iterating through user given array
    pair_list = []
    for i in range(0, arr_size):
        for j in range(i + 1, arr_size):
            if (num_array[i] + num_array[j] == sum):
                list_value = (num_array[i], num_array[j])
                pair_list.append(list_value)

    return pair_list


if __name__ == '__main__':
    arr_size = array_size()
    sum = target_sum()
    array = add_to_array(arr_size)
    pair_list = pair_values(array, arr_size, sum)
    print(f"\nYour entered array is {array}, and the value that these numbers must add up to is {sum}.\n")
    print(f"\nThe pair(s) from the given array that will sum to {sum} is/are {pair_list}.\n")

```

b) Unittest code and output;

```python
#! python3

"""
Simeon Patton
CS362 OSU Spring 2021
Extra Credit 3.1 - Unittest the target sum program

--Write tests for the target_sum program using unittest and pytest.
(do not run the tests with pytest alone - you should use pytest syntax as well.)

"""

import unittest
import target_sum


## Setting global variables
test_array_size = target_sum.array_size()
test_target_sum = target_sum.target_sum()
test_array_list = target_sum.add_to_array(test_array_size)
test_pair_value = target_sum.pair_values(test_array_list, test_array_size, test_target_sum)

class testCase(unittest.TestCase):

    ##  Test to make sure that the input type is the correct data type of int
    def test_input_array_type(self):
        self.assertTrue(type(test_array_size), int)

    ## Test to make sure that the input type is the correct data type of int
    def test_input_sum_type(self):
        self.assertTrue(type(test_target_sum), int)

    ## Test to verify that the length of the list is what was intended by the inputs
    def test_add_to_array(self):
        self.assertEqual(len(test_array_list), test_array_size)

    ## Test to verify that the ordered pairs equal the target sum
    def test_pair_vs_sum(self):
        self.assertEqual((test_pair_value[0][0] + test_pair_value[0][1]), test_target_sum)


if __name__ == '__main__':
    unittest.main()
```

```
PS C:\Users\sup_u\Documents\College\CS362\On Git\CS362_ExtraCredit> python .\test_unittest_target_sum.py

How big would you like the array?
5

What would you like the target sum to be?
10

Please enter a number to add to the array:
4

Please enter a number to add to the array:
6

Please enter a number to add to the array:
5

Please enter a number to add to the array:
1

Please enter a number to add to the array:
2
....
----------------------------------------------------------------------
Ran 4 tests in 0.000s

OK
PS C:\Users\sup_u\Documents\College\CS362\On Git\CS362_ExtraCredit>
```

c) Pytest code and output;
***Had to use the “-s” flag to capture user input

```python
#! python3

"""
Simeon Patton
CS362 OSU Spring 2021
Extra Credit 3.2 - Pytest the target sum program

--Write tests for the target sum program using unittest and pytest.
(do not run the tests with pytest alone - you should use pytest syntax as well.)

"""
## Could not get the pytest fixtures working correctly.
## as of this moment, you need to run pytest with the -s
## flag in order to obtain user input.

import pytest
import target_sum

## Setting global variables
test_array_size = target_sum.array_size()
test_target_sum = target_sum.target_sum()
test_array_list = target_sum.add_to_array(test_array_size)
test_pair_value = target_sum.pair_values(test_array_list, test_array_size, test_target_sum)

## Test to make sure that the input type is the correct data type of int
def test_input_array_type():
    assert type(test_array_size) == int

## Test to make sure that the input type is the correct data type of int
def test_input_sum_type():
    assert type(test_target_sum) == int

## Test to verify that the length of the list is what was intended by the inputs
def test_add_to_array():
    assert len(test_array_list) == test_array_size

## Test to verify that the ordered pairs equal the target sum
def test_pair_vs_sum():
    assert test_pair_value[0][0] + test_pair_value[0][1] == test_target_sum
```

```
PS C:\Users\sup_u\Documents\College\CS362\On Git\CS362_ExtraCredit> pytest -s .\test_pytest_target_sum.py
============================= test session starts =============================
platform win32 -- Python 3.9.0, pytest-6.2.4, py-1.10.0, pluggy-0.13.1
rootdir: C:\Users\sup_u\Documents\College\CS362\On Git\CS362_ExtraCredit
plugins: cov-2.12.0
collecting ...
How big would you like the array?
5

What would you like the target sum to be?
10

Please enter a number to add to the array:
4

Please enter a number to add to the array:
6

Please enter a number to add to the array:
5

Please enter a number to add to the array:
2

Please enter a number to add to the array:
3
collected 4 items

test_pytest_target_sum.py ....

============================== 4 passed in 9.56s ==============================
PS C:\Users\sup_u\Documents\College\CS362\On Git\CS362_ExtraCredit>
```