Task 1

Unit testing

Solve problems and implement tests for small code units

Here is the background information on your task

Unit testing is the low-level testing where developers verify that isolated sections of the code, called units, work as intended. Unit tests are implemented during the coding phase, by developers, by isolating a part of the code to verify its correctness. For example, it can be isolating functions and write tests for these. The purpose of this is to make sure they work as intended. Finding problems and defects at this stage is important since they may cause delays and additional costs due to defect identification and correction later in the project.

An example of a function and a connected unit-test:

Function:

```
12  def hello_function():
13      return "hello"
```

A function that returns the string "hello".

A connected unit-tests:

```
 9  def test_hello_function():
10      string = "hello"
11      assert (string == hello_function()) is True
```

Tests that the return of hello_function() is equal to "hello".

Here is your task

Part 1:

You are joining an ongoing project, replacing a front-end developer who unexpectedly became long-term ill while developing the "create account" page for a new mobile application. To enter this role, you need to get familiar with the previously written code, for which you have been informed that some of the written functions already have related unit tests. You decide that a good approach is to initially run these tests, but after running them you realize that some of them fail. This concludes that one, or several, functions don't work as intended. Your first task is therefore to edit the code to ensure that the functions work as they should.

Note: The tests are correct, the error are caused by bugs in the main file. So, it is only the main.py that should be modified, ***not*** *the testrunner file.*

Part 2:

After successfully running the implemented tests you notice two recently added functions. The front-end developer has created a draft of unit tests for these, in the file testrunner.py, but have left them uncompleted.

The draft of the new unit tests now looks like this:

```
51
52  # TODO write the unit tests for the new functions, replace pass with
    your code: ----------------------------
53  # 1. Test that two equal passwords return true
54  def test_psw_equal1():
55      pass
56
57  # 2. Test that the function is not case sencative
58  def test_psw_equal2():
59      pass
60
61  # 3. Test that two inequal passwords return false
62  def test_psw_equal3():
63      pass
64
65  # 4. Test that two equal passwords in the correct format and a correct
    email return true
66  def test_check_credentials1():
67      pass
68
69  # 5. Test that two inequal passwords in the correct format and a
    correct email return false
70  def test_check_credentials2():
71      pass
72
73  # 6. Test that two equal passwords in the correct format and an
    incorrect email return false
74  def test_check_credentials3():
75      pass
76
```

Remove the comment notation (''' ''') and change the "assert False" part of each test to finish them and ensure that the functions work as explained in their respective comments. If bugs are found, correct these.

- In this task you will use the website **repl.it**. This allows you to use a pre-setup environment, so you do not have to worry about what OS and python versions you are using. If you do not have a repl account, go to repl.it and create one. **Review section 4 'Resources to help you with the task' below for further information on getting started.**

Resources to help you with the task

Repl.it instructions

1. Login to repl.it
2. Click here to open repl environment
3. Click **FORK** to be able to run and make changes in the code
4. Click on the "run" button in the top middle. This will install and update things in repl and then the tests will run.
5. Find and solve the bug(s) in main.py, re-run the program to all tests pass.
6. In the end of main.py see the functions for question 2 and write the tests in testrunner.py
7. When you feel happy with your result; In the file tap, click on the three dots and download your solution as a zip.
8. Upload your solution in this task, this action will unlock an answer suggestion from Accenture.

Repl.it Environment

Here is access to the pre-setup environment in Repl.it

Click to start your task! →

Link to Python Guide:

https://www.w3schools.com/python/python_conditions.asp