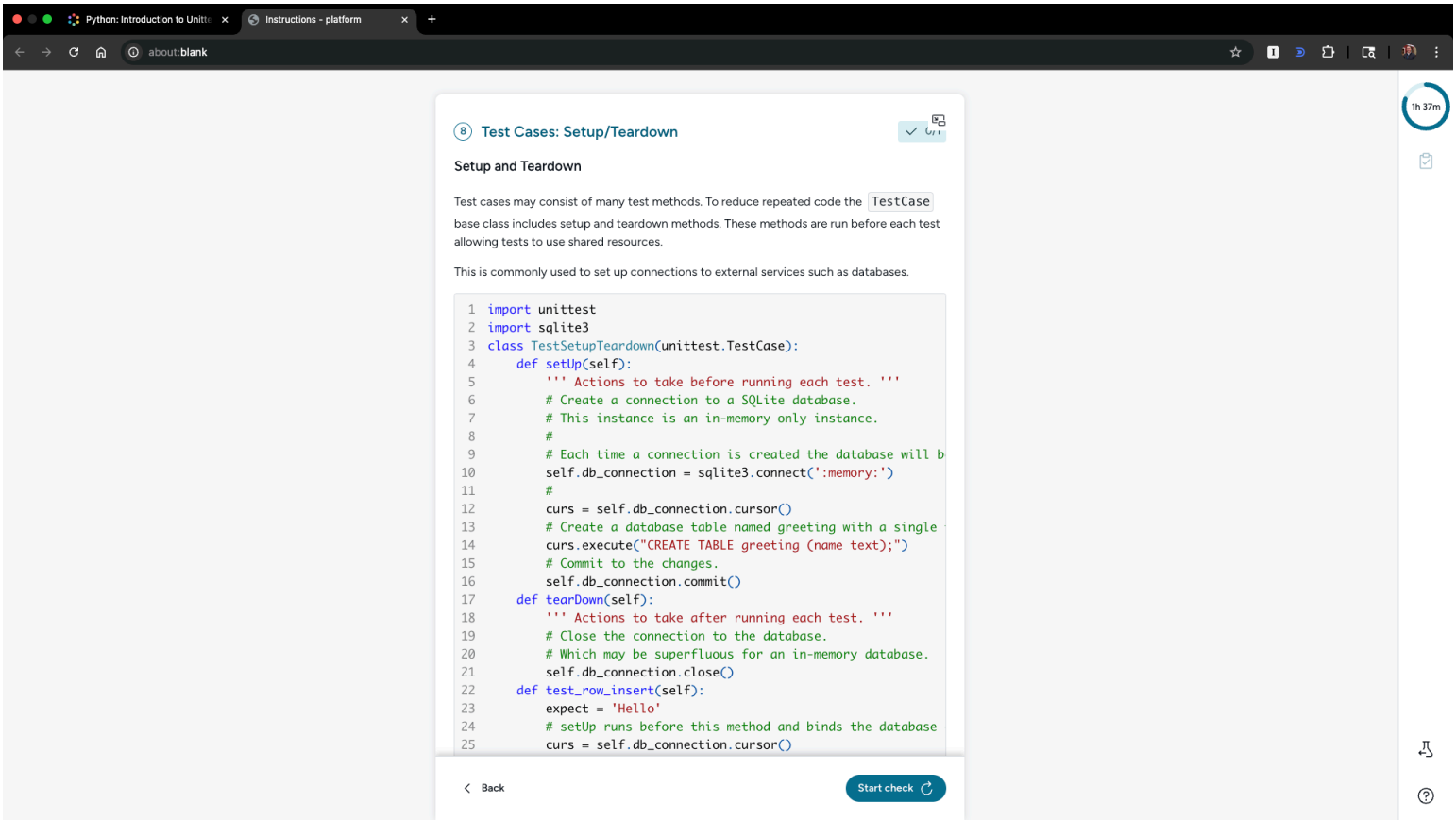
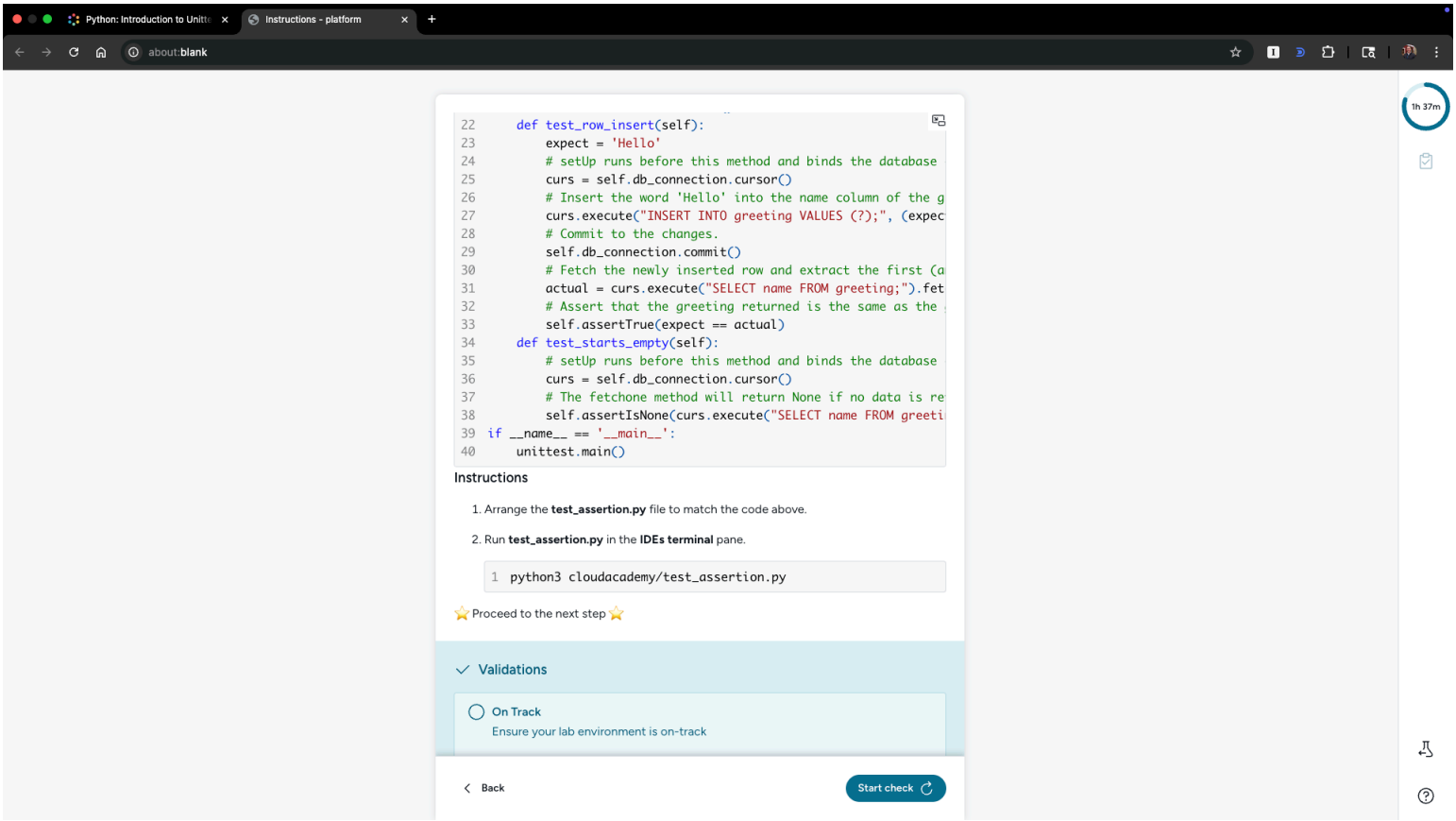

Python: Introduction to Unitte
Instructions - platform
about:blank
1h 37m
8 Test Cases: Setup/Teardown
Setup and Teardown
Test cases may consist of many test methods. To reduce repeated code the TestCase base class includes setup and teardown methods. These methods are run before each test allowing tests to use shared resources.
This is commonly used to set up connections to external services such as databases.
1 import unittest
2 import sqlite3
3 class TestSetupTeardown(unittest.TestCase):
4 def setUp(self):
5 ''' Actions to take before running each test. '''
6 # Create a connection to a SQLite database.
7 # This instance is an in-memory only instance.
8 #
9 # Each time a connection is created the database will b
10 self.db_connection = sqlite3.connect(':memory:')
11 #
12 curs = self.db_connection.cursor()
13 # Create a database table named greeting with a single
14 curs.execute("CREATE TABLE greeting (name text);")
15 # Commit to the changes.
16 self.db_connection.commit()
17 def tearDown(self):
18 ''' Actions to take after running each test. '''
19 # Close the connection to the database.
20 # Which may be superfluous for an in-memory database.
21 self.db_connection.close()
22 def test_row_insert(self):
23 expect = 'Hello'
24 # setUp runs before this method and binds the database
25 curs = self.db_connection.cursor()
Back
Start check


Python: Introduction to Unitte
Instructions - platform
about:blank
1h 37m
22 def test_row_insert(self):
23 expect = 'Hello'
24 # setUp runs before this method and binds the database
25 curs = self.db_connection.cursor()
26 # Insert the word 'Hello' into the name column of the g
27 curs.execute("INSERT INTO greeting VALUES (?);", (expec
28 # Commit to the changes.
29 self.db_connection.commit()
30 # Fetch the newly inserted row and extract the first (a
31 actual = curs.execute("SELECT name FROM greeting;").fet
32 # Assert that the greeting returned is the same as the
33 self.assertTrue(expect == actual)
34 def test_starts_empty(self):
35 # setUp runs before this method and binds the database
36 curs = self.db_connection.cursor()
37 # The fetchone method will return None if no data is re
38 self.assertIsNone(curs.execute("SELECT name FROM greeti
39 if __name__ == '__main__':
40 unittest.main()
Instructions
1. Arrange the test_assertion.py file to match the code above.
2. Run test_assertion.py in the IDEs terminal pane.
1 python3 cloudacademy/test_assertion.py
Proceed to the next step
Validations
On Track
Ensure your lab environment is on-track
Back
Start check