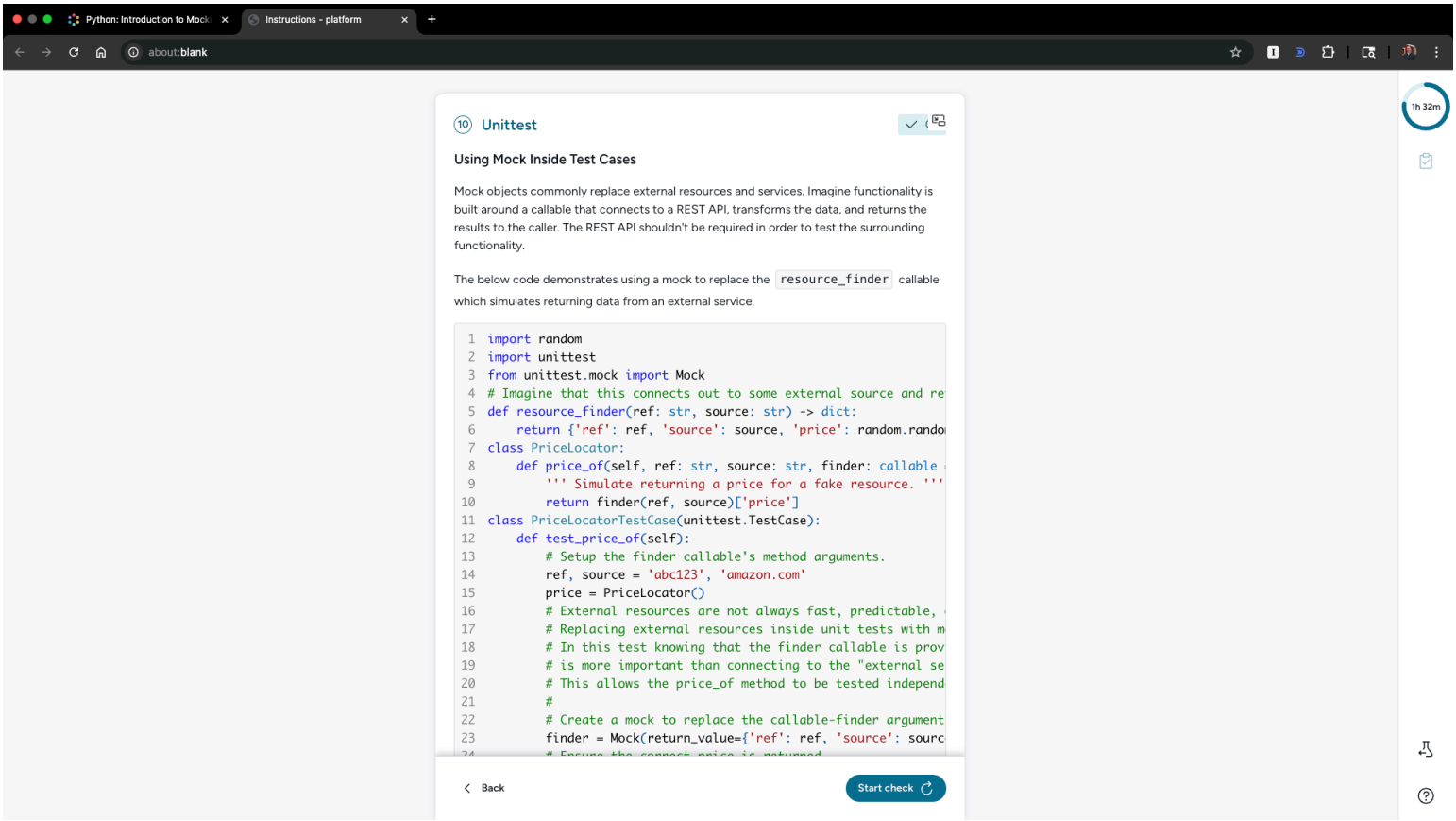
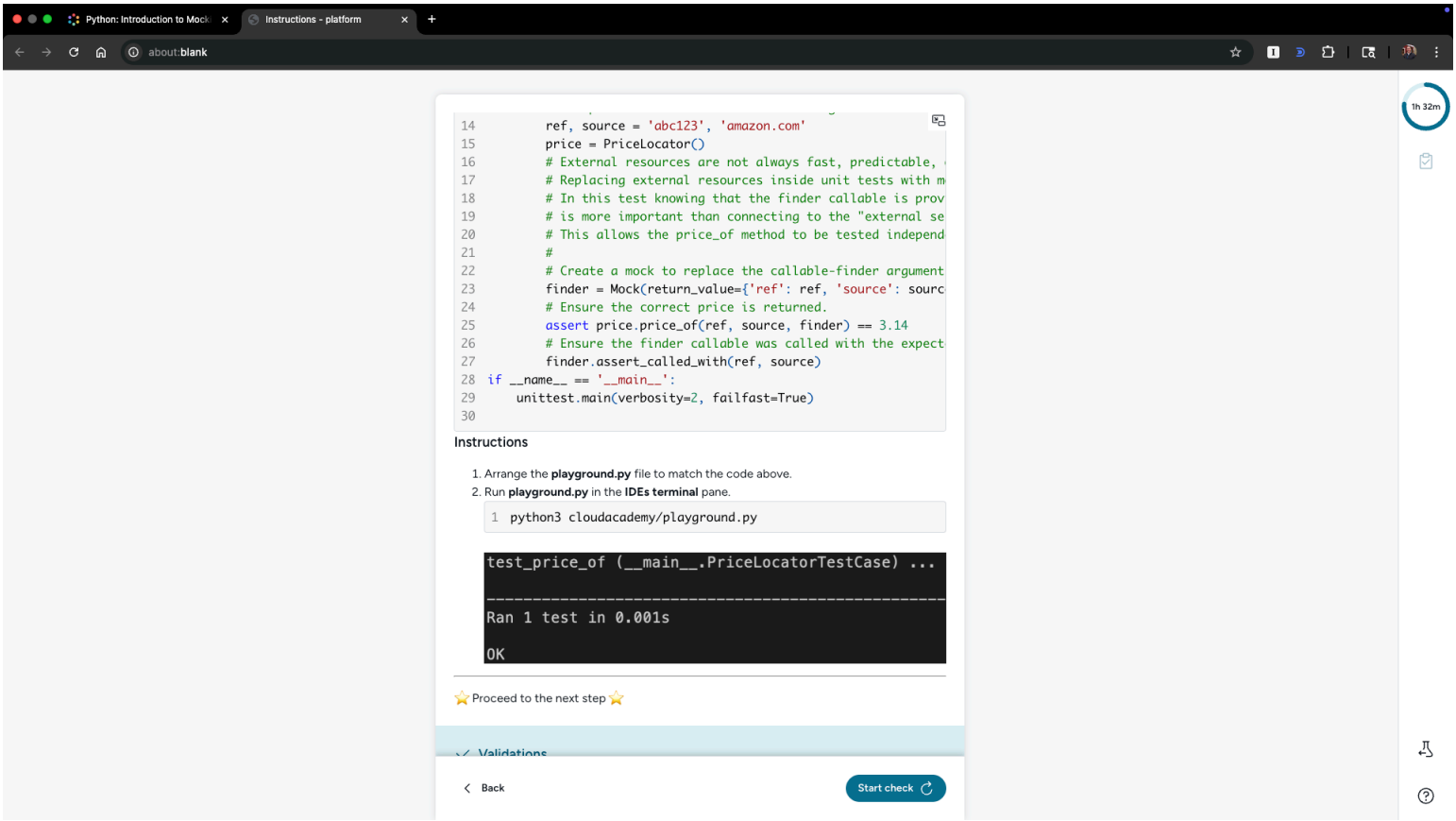

Python: Introduction to Mock
Instructions - platform
about:blank
1h 32m
10 Unittest
Using Mock Inside Test Cases
Mock objects commonly replace external resources and services. Imagine functionality is built around a callable that connects to a REST API, transforms the data, and returns the results to the caller. The REST API shouldn't be required in order to test the surrounding functionality.
The below code demonstrates using a mock to replace the resource_finder callable which simulates returning data from an external service.
import random
import unittest
from unittest.mock import Mock
# Imagine that this connects out to some external source and re
def resource_finder(ref: str, source: str) -> dict:
return {'ref': ref, 'source': source, 'price': random.rando
class PriceLocator:
def price_of(self, ref: str, source: str, finder: callable
''' Simulate returning a price for a fake resource. '''
return finder(ref, source)['price']
class PriceLocatorTestCase(unittest.TestCase):
def test_price_of(self):
# Setup the finder callable's method arguments.
ref, source = 'abc123', 'amazon.com'
price = PriceLocator()
# External resources are not always fast, predictable,
# Replacing external resources inside unit tests with m
# In this test knowing that the finder callable is prov
# is more important than connecting to the "external se
# This allows the price_of method to be tested independ
#
# Create a mock to replace the callable-finder argument
finder = Mock(return_value={'ref': ref, 'source': sourc
Back
Start check


Python: Introduction to Mock
Instructions - platform
about:blank
1h 32m
14 ref, source = 'abc123', 'amazon.com'
15 price = PriceLocator()
16 # External resources are not always fast, predictable,
17 # Replacing external resources inside unit tests with m
18 # In this test knowing that the finder callable is prov
19 # is more important than connecting to the "external se
20 # This allows the price_of method to be tested independ
21 #
22 # Create a mock to replace the callable-finder argument
23 finder = Mock(return_value={'ref': ref, 'source': sourc
24 # Ensure the correct price is returned.
25 assert price.price_of(ref, source, finder) == 3.14
26 # Ensure the finder callable was called with the expect
27 finder.assert_called_with(ref, source)
28 if __name__ == '__main__':
29 unittest.main(verbosity=2, failfast=True)
30
Instructions
1. Arrange the playground.py file to match the code above.
2. Run playground.py in the IDEs terminal pane.
1 python3 cloudacademy/playground.py
test_price_of (__main__.PriceLocatorTestCase) ...
Ran 1 test in 0.001s
OK
Proceed to the next step
Validations
Back
Start check