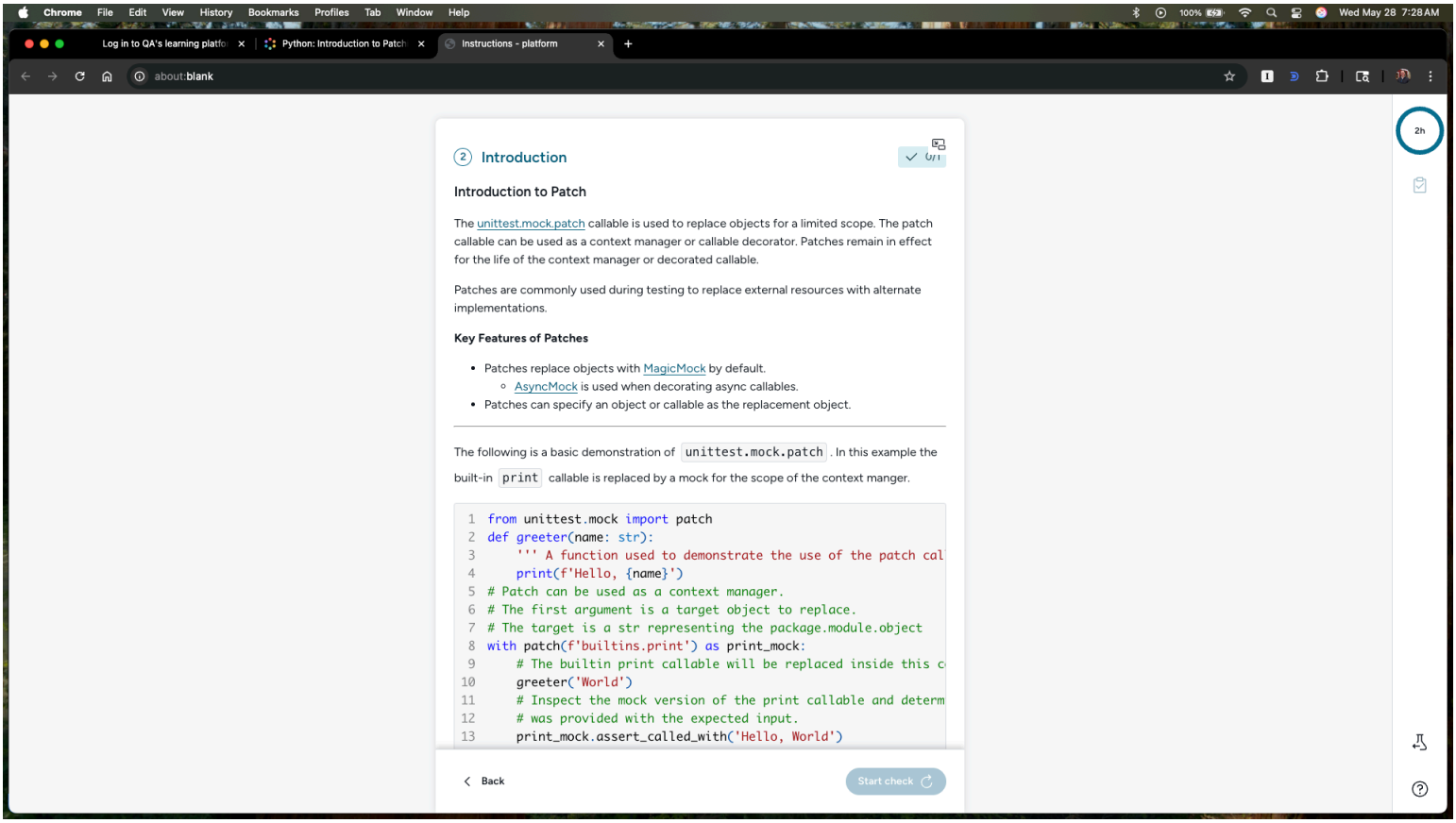
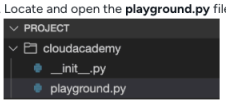

Chrome File Edit View History Bookmarks Profiles Tab Window Help
Wed May 28 7:28 AM
Log in to QA's learning platfo
Python: Introduction to Patch
Instructions - platform
about:blank
2h
2 Introduction
0/1
Introduction to Patch
The unittest.mock.patch callable is used to replace objects for a limited scope. The patch callable can be used as a context manager or callable decorator. Patches remain in effect for the life of the context manager or decorated callable.
Patches are commonly used during testing to replace external resources with alternate implementations.
Key Features of Patches
Patches replace objects with MagicMock by default.
AsyncMock is used when decorating async callables.
Patches can specify an object or callable as the replacement object.
The following is a basic demonstration of unittest.mock.patch . In this example the built-in print callable is replaced by a mock for the scope of the context manger.
1 from unittest.mock import patch
2 def greeter(name: str):
3 ''' A function used to demonstrate the use of the patch cal
4 print(f'Hello, {name}')
5 # Patch can be used as a context manager.
6 # The first argument is a target object to replace.
7 # The target is a str representing the package.module.object
8 with patch(f'builtins.print') as print_mock:
9 # The builtin print callable will be replaced inside this c
10 greeter('World')
11 # Inspect the mock version of the print callable and determ
12 # was provided with the expected input.
13 print_mock.assert_called_with('Hello, World')
Back
Start check

```
    # Inspect the mock version of the print callable and dete
    # was provided with the expected input.
    print_mock.assert_called_with('Hello, World')
    print_mock.assert_called_once()
# Outside the context manager print behaves normally.
greeter('World')
#################################################################
print('No assertion errors')
```

Instructions

1. Locate and open the **playground.py** file using the **IDE's Explorer/project** pane.



2. Arrange the **playground.py** file to match the code above.
3. Run **playground.py** in the **IDEs terminal** pane: **(Terminal > New Terminal)**

```
python3 cloudacademy/playground.py
```

```
Hello, World
No assertion errors
```

⭐ Proceed to the next step ⭐

Validations

On Track - Introduction

Ensure your lab environment is on-track

Python

Back

Start check