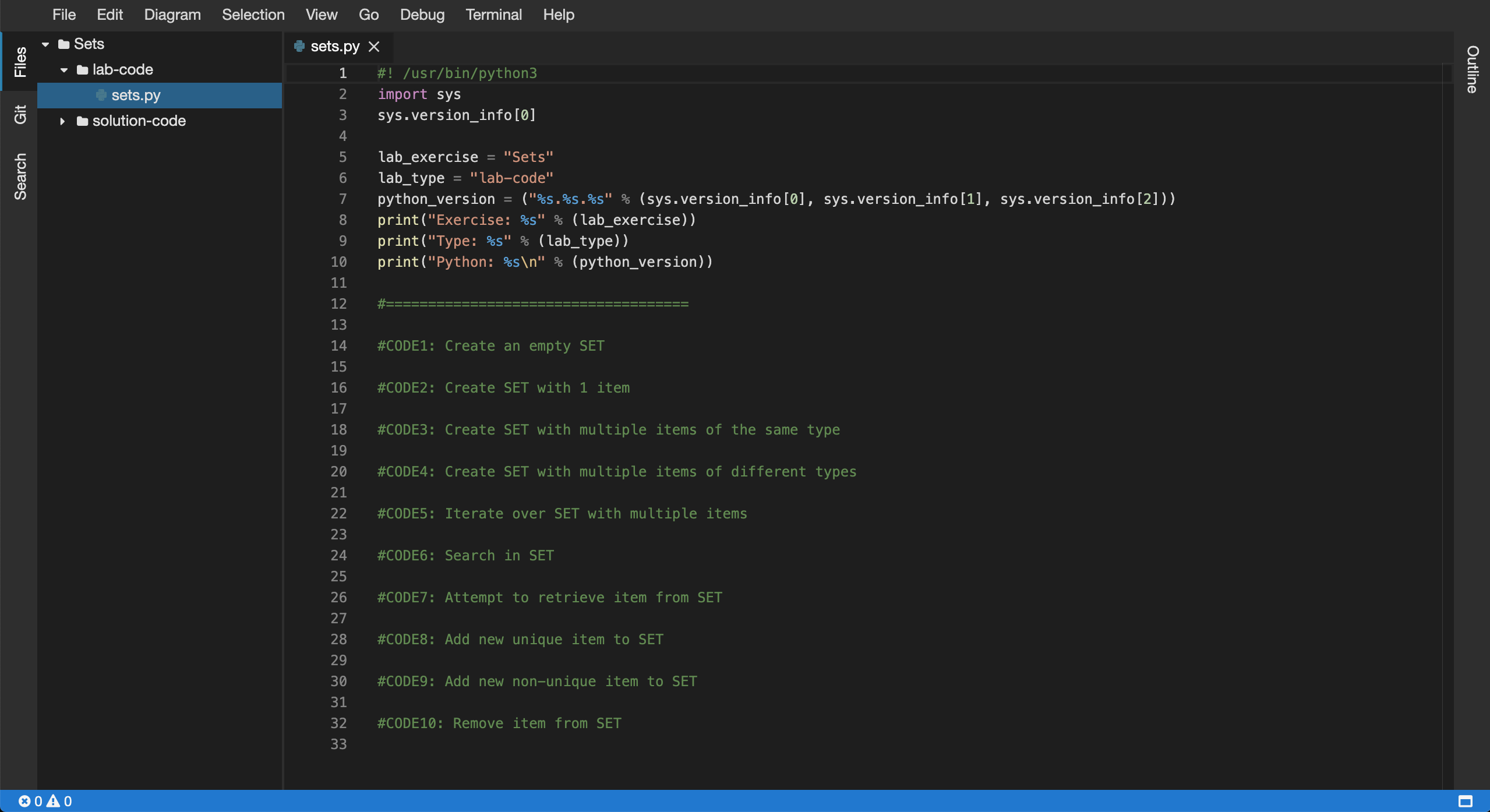
Sets

Open the sets.py source file within the editor. Take some time to review the uncompleted code within this file to understand the intended design:



5. Next, you will be required to complete the code in the following source files:

* sets.py

6. Within the editor ensure that the sets.py file now has focus.

7. Replace the //Code1: comment, create an empty SET:

[**Copy code**](https://app.qa.com/lab/coding-python-collection-types/exercise-3-sets/?context_id=637&context_resource=lp)

#CODE1: Create an empty SET

set1 = set()

print("CODE1:")

print(f"set1 = {set1}")

print(f"data type = {type(set1)}")

print(f"length = {len(set1)}")

print()

8. Replace the //Code2: comment, create SET with 1 item:

[**Copy code**](https://app.qa.com/lab/coding-python-collection-types/exercise-3-sets/?context_id=637&context_resource=lp)

#CODE2: Create SET with 1 item

set2 = {"cloudacademy"}

print("CODE2:")

print(f"set2 = {set2}")

print(f"data type = {type(set2)}")

print(f"length = {len(set2)}")

print()

9. Replace the //Code3: comment, create SET with multiple items of the same type:

[**Copy code**](https://app.qa.com/lab/coding-python-collection-types/exercise-3-sets/?context_id=637&context_resource=lp)

#CODE3: Create SET with multiple items of the same type

set3 = {1, 2, 3, 4, 5}

print("CODE3:")

print(f"set3 = {set3}")

print(f"data type = {type(set3)}")

print(f"length = {len(set3)}")

print()

10. Replace the //Code4: comment, create SET with multiple items of different types:

[**Copy code**](https://app.qa.com/lab/coding-python-collection-types/exercise-3-sets/?context_id=637&context_resource=lp)

#CODE4: Create SET with multiple items of different types

set4 = {"cloud", "academy", 1, True, False}

print("CODE4:")

print(f"set4 = {set4}")

print(f"data type = {type(set4)}")

print(f"length = {len(set4)}")

print()

11. Replace the //Code5: comment, iterate over SET with multiple items:

Note: Ensure the indentation within the following code block is maintained within the editor view when you perform the copy and paste operation.

[**Copy code**](https://app.qa.com/lab/coding-python-collection-types/exercise-3-sets/?context_id=637&context_resource=lp)

#CODE5: Iterate over SET with multiple items

print("CODE5:")

for item in set4:

print(item)

print()

12. Replace the //Code6: comment, search in SET:

[**Copy code**](https://app.qa.com/lab/coding-python-collection-types/exercise-3-sets/?context_id=637&context_resource=lp)

#CODE6: Search in SET

print("CODE6:")

print ("cloud" in set4)

print ("blah" in set4)

print()

13. Replace the //Code7: comment, attempt to retrieve item from SET:

Note: Ensure the indentation within the following code block is maintained within the editor view when you perform the copy and paste operation.

[**Copy code**](https://app.qa.com/lab/coding-python-collection-types/exercise-3-sets/?context_id=637&context_resource=lp)

#CODE7: Attempt to retrieve item from SET

print("CODE7:")

try:

item0 = set4[0]

item1 = set4[1]

print(f"item0 = {item0}")

print(f"item1 = {item1}")

except:

print("Sets do not support indexing!!")

print()

14. Replace the //Code8: comment, add new unique item to SET:

[**Copy code**](https://app.qa.com/lab/coding-python-collection-types/exercise-3-sets/?context_id=637&context_resource=lp)

#CODE8: Add new unique item to SET

print("CODE8:")

set4.add("devops")

print(f"set4 = {set4}")

print(f"data type = {type(set4)}")

print(f"length = {len(set4)}")

print()

15. Replace the //Code9: comment, add new non-unique item to SET:

[**Copy code**](https://app.qa.com/lab/coding-python-collection-types/exercise-3-sets/?context_id=637&context_resource=lp)

#CODE9: Add new non-unique item to SET

print("CODE9:")

set4.add("devops") # added prev

print(f"set4 = {set4}")

print(f"data type = {type(set4)}")

print(f"length = {len(set4)}")

print()

16. Replace the //Code10: comment, remove item from SET:

[**Copy code**](https://app.qa.com/lab/coding-python-collection-types/exercise-3-sets/?context_id=637&context_resource=lp)

#CODE10: Remove item from SET

print("CODE10:")

set4.remove("cloud")

print(f"set4 = {set4}")

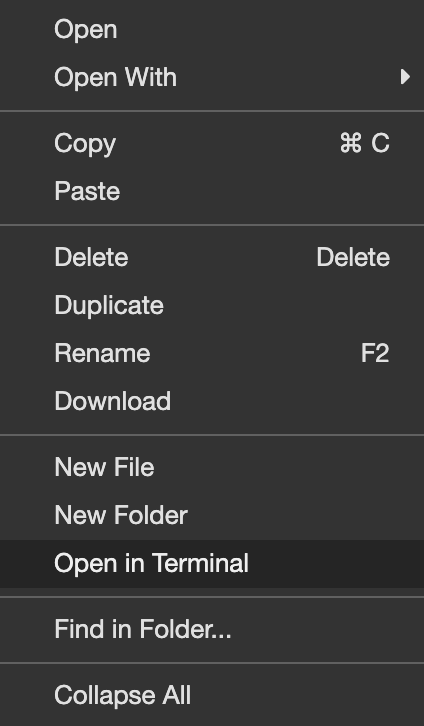
print(f"data type = {type(set4)}")

print(f"length = {len(set4)}")

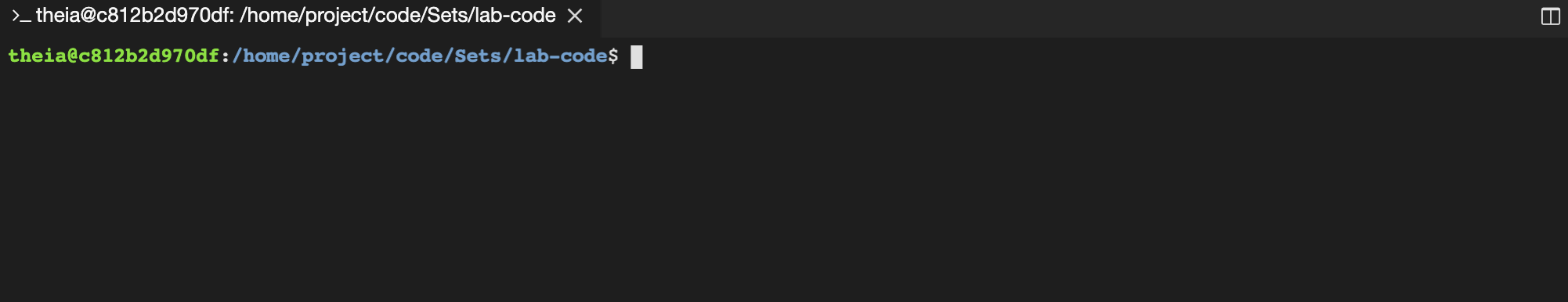
print()

17. Ok the sets.py Python script is now ready to be executed by the Python interpreter. To do so you will need to use the embedded terminal to launch Python.

18. Within the Files tree view (lefthand side menu), select the lab-code folder and right click and select the Open in Terminal option:

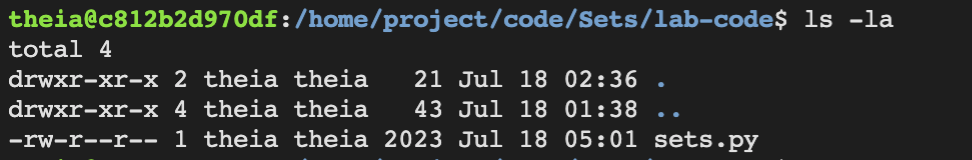


19. Access to the embedded terminal is now provided in the bottom pane of the current view:

  
20. Peform a directory listing on the current directory to ensure the presence of the sets.py file, like so:

[**Copy code**](https://app.qa.com/lab/coding-python-collection-types/exercise-3-sets/?context_id=637&context_resource=lp)

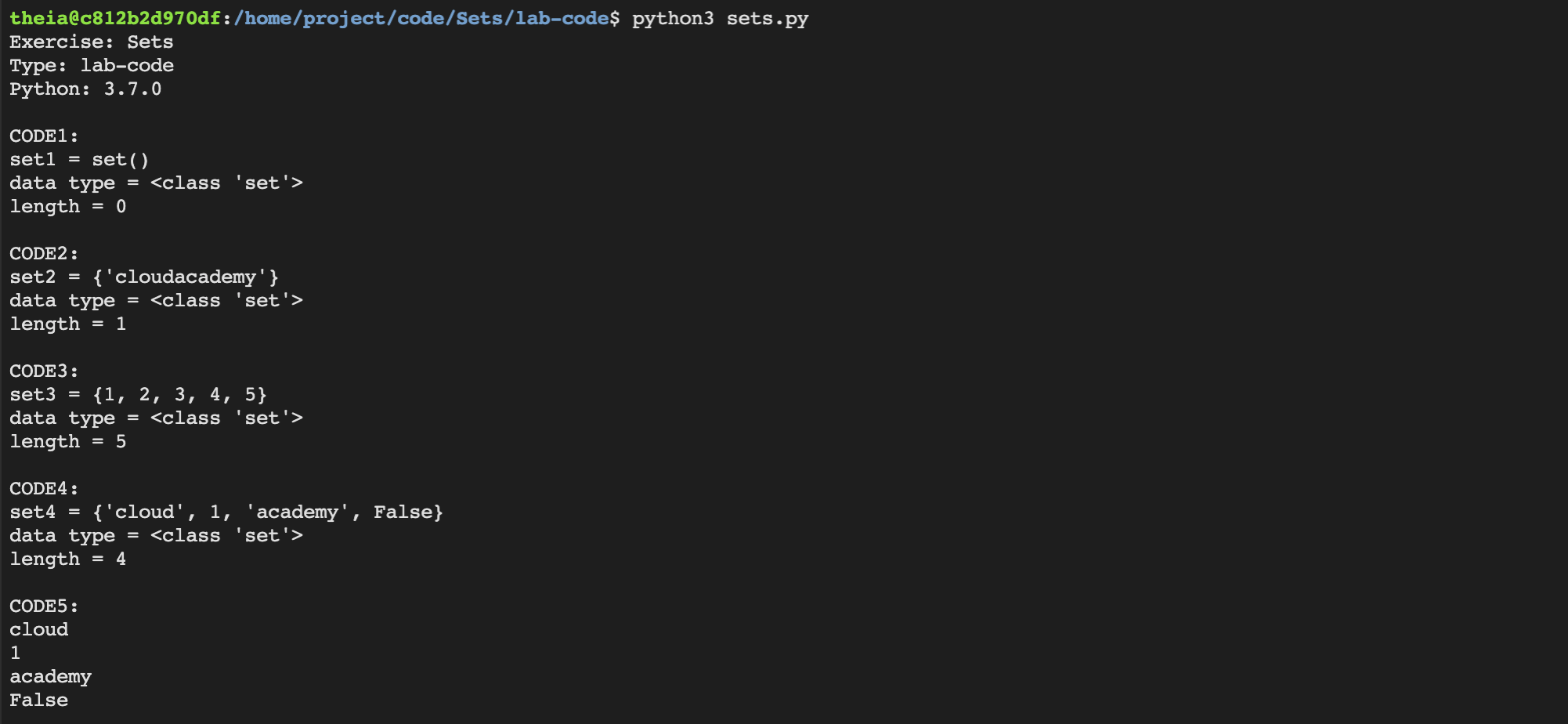
ls -la



21. Let's now execute the updated sets.py script using the python3 command. Within the terminal enter the following command:

[**Copy code**](https://app.qa.com/lab/coding-python-collection-types/exercise-3-sets/?context_id=637&context_resource=lp)

python3 sets.py





22. Great! Your sets.py script has executed successfully! This demonstrates how to work with Sets.

23. Try updating the sets.py script and then re-executing using the same steps above. See the following documentation for ideas:

[https://docs.python.org/3.3/tutorial/datastructures.html#sets](https://docs.python.org/3.3/tutorial/datastructures.html" \l "sets)

### 

### Summary

In this Lab Step, you opened the Sets Workspace and then updated the sets.py file to store and manipulate data using Sets. Next you used the embedded terminal to launch and debug the sets.py script file.

Solution Code

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#! /usr/bin/python3

import sys

sys.version\_info[0]

lab\_exercise = "Sets"

lab\_type = "solution-code"

python\_version = ("%s.%s.%s" % (sys.version\_info[0], sys.version\_info[1], sys.version\_info[2]))

print("Exercise: %s" % (lab\_exercise))

print("Type: %s" % (lab\_type))

print("Python: %s\n" % (python\_version))

#====================================

#CODE1: Create an empty SET

set1 = set()

print("CODE1:")

print(f"set1 = {set1}")

print(f"data type = {type(set1)}")

print(f"length = {len(set1)}")

print()

#CODE2: Create SET with 1 item

set2 = {"cloudacademy"}

print("CODE2:")

print(f"set2 = {set2}")

print(f"data type = {type(set2)}")

print(f"length = {len(set2)}")

print()

#CODE3: Create SET with multiple items of the same type

set3 = {1, 2, 3, 4, 5}

print("CODE3:")

print(f"set3 = {set3}")

print(f"data type = {type(set3)}")

print(f"length = {len(set3)}")

print()

#CODE4: Create SET with multiple items of different types

set4 = {"cloud", "academy", 1, True, False}

print("CODE4:")

print(f"set4 = {set4}")

print(f"data type = {type(set4)}")

print(f"length = {len(set4)}")

print()

#CODE5: Iterate over SET with multiple items

print("CODE5:")

for item in set4:

print(item)

print()

#CODE6: Search in SET

print("CODE6:")

print ("cloud" in set4)

print ("blah" in set4)

print()

#CODE7: Attempt to retrieve item from SET

print("CODE7:")

try:

item0 = set4[0]

item1 = set4[1]

print(f"item0 = {item0}")

print(f"item1 = {item1}")

except:

print("Sets do not support indexing!!")

print()

#CODE8: Add new unique item to SET

print("CODE8:")

set4.add("devops")

print(f"set4 = {set4}")

print(f"data type = {type(set4)}")

print(f"length = {len(set4)}")

print()

#CODE9: Add new non-unique item to SET

print("CODE9:")

set4.add("devops") # added prev

print(f"set4 = {set4}")

print(f"data type = {type(set4)}")

print(f"length = {len(set4)}")

print()

#CODE10: Remove item from SET

print("CODE10:")

set4.remove("cloud")

print(f"set4 = {set4}")

print(f"data type = {type(set4)}")

print(f"length = {len(set4)}")

print()

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

output console

Exercise: Sets

Type: solution-code

Python: 3.7.0

CODE1:

set1 = set()

data type = <class 'set'>

length = 0

CODE2:

set2 = {'cloudacademy'}

data type = <class 'set'>

length = 1

CODE3:

set3 = {1, 2, 3, 4, 5}

data type = <class 'set'>

length = 5

CODE4:

set4 = {False, 'cloud', 'academy', 1}

data type = <class 'set'>

length = 4

CODE5:

False

cloud

academy

1

CODE6:

True

False

CODE7:

Sets do not support indexing!!

CODE8:

set4 = {False, 'devops', 1, 'academy', 'cloud'}

data type = <class 'set'>

length = 5

CODE9:

set4 = {False, 'devops', 1, 'academy', 'cloud'}

data type = <class 'set'>

length = 5

CODE10:

set4 = {False, 'devops', 1, 'academy'}

data type = <class 'set'>

length = 4

theia@production-session-93142-6dd54db79c-z8dtq:/home/project/code/Sets/solution-code$