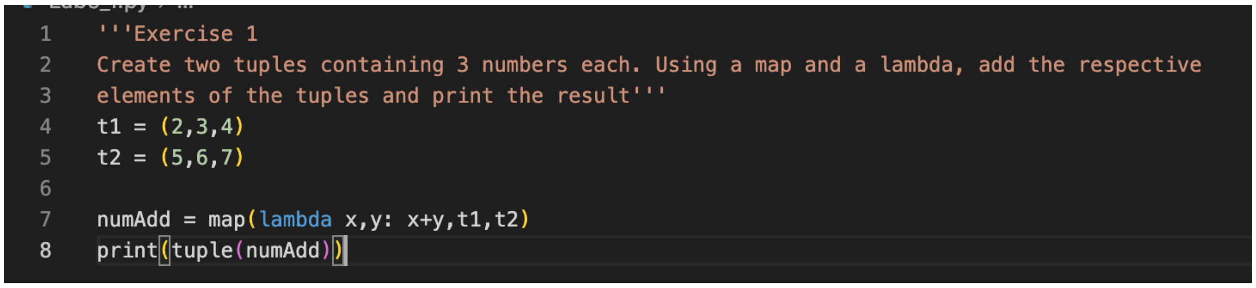
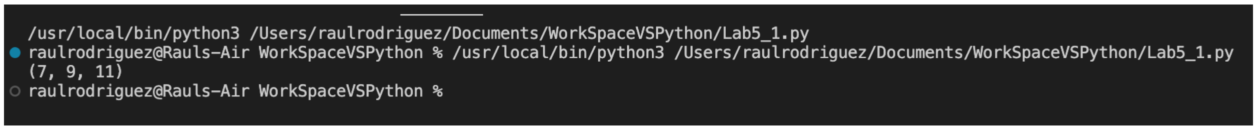
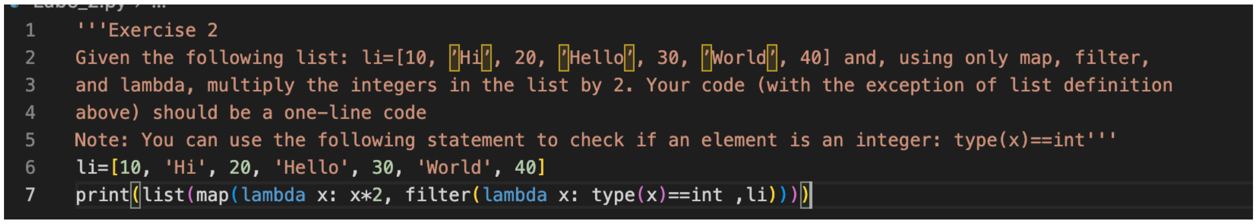
Raul Rodriguez
Python intro to data science



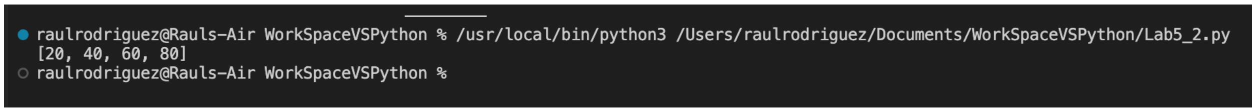
```python
'''Exercise 1
Create two tuples containing 3 numbers each. Using a map and a lambda, add the respective
elements of the tuples and print the result'''
t1 = (2,3,4)
t2 = (5,6,7)

numAdd = map(lambda x,y: x+y,t1,t2)
print(tuple(numAdd))
```



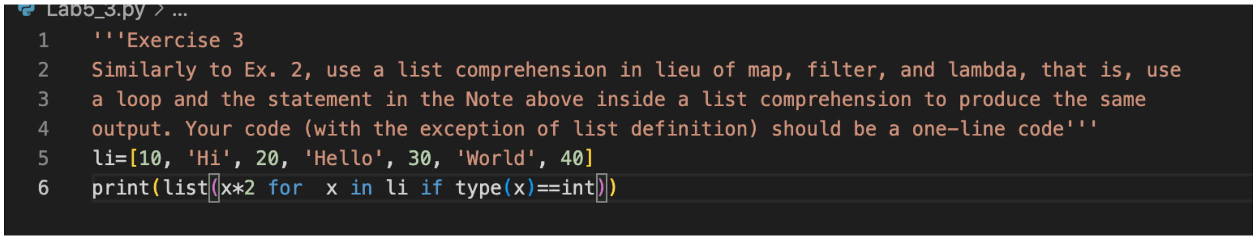
```
/usr/local/bin/python3 /Users/raulrodriguez/Documents/WorkSpaceVSPython/Lab5_1.py
● raulrodriguez@Rauls-Air WorkSpaceVSPython % /usr/local/bin/python3 /Users/raulrodriguez/Documents/WorkSpaceVSPython/Lab5_1.py
(7, 9, 11)
○ raulrodriguez@Rauls-Air WorkSpaceVSPython %
```



```python
'''Exercise 2
Given the following list: li=[10, ’Hi’, 20, ’Hello’, 30, ’World’, 40] and, using only map, filter,
and lambda, multiply the integers in the list by 2. Your code (with the exception of list definition
above) should be a one-line code
Note: You can use the following statement to check if an element is an integer: type(x)==int'''
li=[10, 'Hi', 20, 'Hello', 30, 'World', 40]
print(list(map(lambda x: x*2, filter(lambda x: type(x)==int ,li))))
```

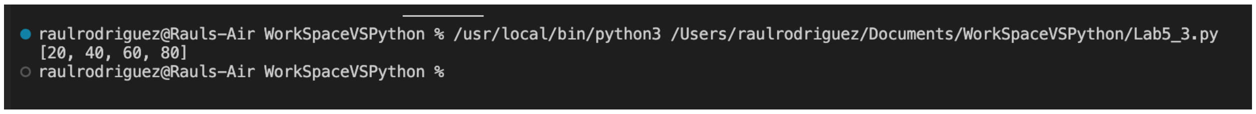


```
● raulrodriguez@Rauls-Air WorkSpaceVSPython % /usr/local/bin/python3 /Users/raulrodriguez/Documents/WorkSpaceVSPython/Lab5_2.py
[20, 40, 60, 80]
○ raulrodriguez@Rauls-Air WorkSpaceVSPython %
```

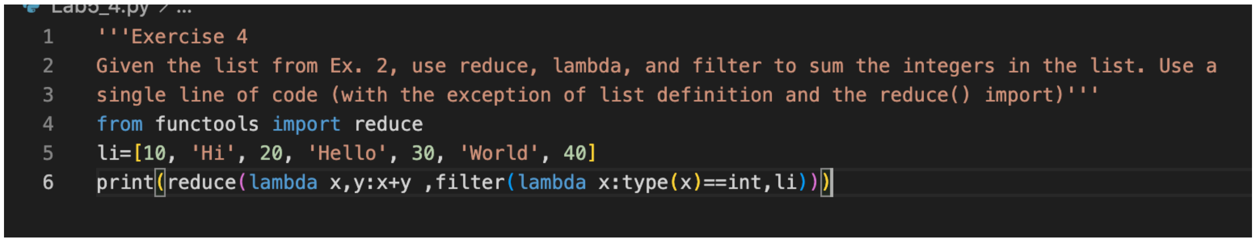


Lab5_3.py > ...

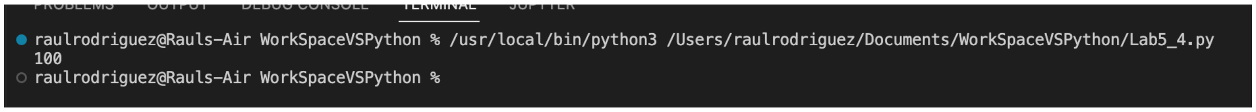
```python
'''Exercise 3
Similarly to Ex. 2, use a list comprehension in lieu of map, filter, and lambda, that is, use
a loop and the statement in the Note above inside a list comprehension to produce the same
output. Your code (with the exception of list definition) should be a one-line code'''
li=[10, 'Hi', 20, 'Hello', 30, 'World', 40]
print(list(x*2 for  x in li if type(x)==int))
```



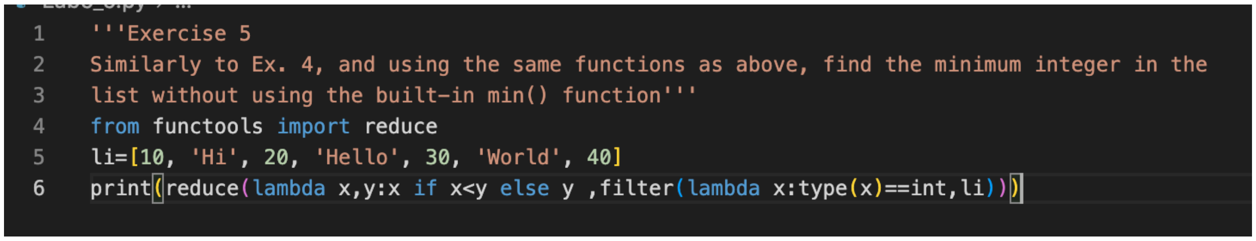
```
● raulrodriguez@Rauls-Air WorkSpaceVSPython % /usr/local/bin/python3 /Users/raulrodriguez/Documents/WorkSpaceVSPython/Lab5_3.py
[20, 40, 60, 80]
○ raulrodriguez@Rauls-Air WorkSpaceVSPython %
```



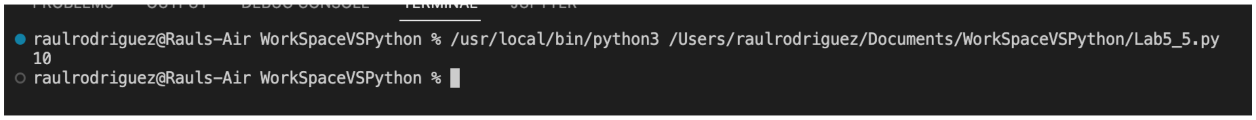
```python
'''Exercise 4
Given the list from Ex. 2, use reduce, lambda, and filter to sum the integers in the list. Use a
single line of code (with the exception of list definition and the reduce() import)'''
from functools import reduce
li=[10, 'Hi', 20, 'Hello', 30, 'World', 40]
print(reduce(lambda x,y:x+y ,filter(lambda x:type(x)==int,li)))
```



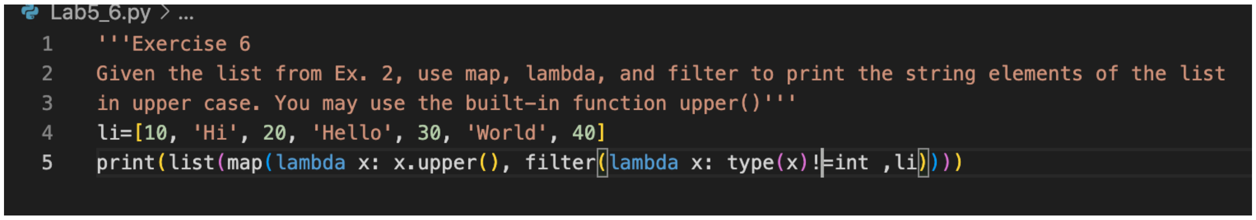
```
● raulrodriguez@Rauls-Air WorkSpaceVSPython % /usr/local/bin/python3 /Users/raulrodriguez/Documents/WorkSpaceVSPython/Lab5_4.py
100
○ raulrodriguez@Rauls-Air WorkSpaceVSPython %
```



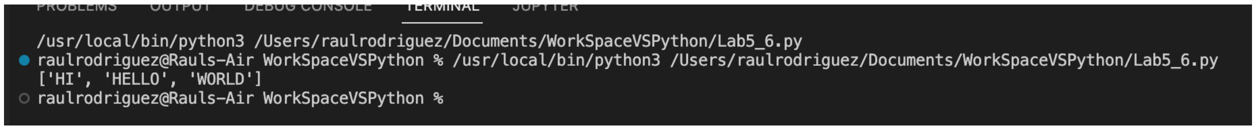
```python
'''Exercise 5
Similarly to Ex. 4, and using the same functions as above, find the minimum integer in the
list without using the built-in min() function'''
from functools import reduce
li=[10, 'Hi', 20, 'Hello', 30, 'World', 40]
print(reduce(lambda x,y:x if x<y else y ,filter(lambda x:type(x)==int,li)))
```



```
raulrodriguez@Rauls-Air WorkSpaceVSPython % /usr/local/bin/python3 /Users/raulrodriguez/Documents/WorkSpaceVSPython/Lab5_5.py
10
raulrodriguez@Rauls-Air WorkSpaceVSPython %
```



```python
'''Exercise 6
Given the list from Ex. 2, use map, lambda, and filter to print the string elements of the list
in upper case. You may use the built-in function upper()'''
li=[10, 'Hi', 20, 'Hello', 30, 'World', 40]
print(list(map(lambda x: x.upper(), filter(lambda x: type(x)!=int ,li))))
```



```
/usr/local/bin/python3 /Users/raulrodriguez/Documents/WorkSpaceVSPython/Lab5_6.py
raulrodriguez@Rauls-Air WorkSpaceVSPython % /usr/local/bin/python3 /Users/raulrodriguez/Documents/WorkSpaceVSPython/Lab5_6.py
['HI', 'HELLO', 'WORLD']
raulrodriguez@Rauls-Air WorkSpaceVSPython %
```