

Assignment 1
DSC 450
Daniel O'Brien
01/10/21
Part 1

A.

For part A, I created a function that received a string of numbers. I then removed the commas from the string using the split function. I found the length of the string to use when solving for the average. I then added each item in the string to a list as an integer. Finally, I found the sum of the list and divided it by the length of the list to find the average. I returned the average as the variable, 'res'.

Python Code:

```
def average(lst):  
    newLst = lst.split(',')  
    length = len(newLst)  
    intLst = []  
    for item in newLst:  
        intLst.append(int(item))  
    res = sum(intLst)/length  
    return res
```

```
print(average('1, 2, 3, 4, 5'))
```

```
def average(lst):  
    newLst = lst.split(',')  
    length = len(newLst)  
    intLst = []  
    for item in newLst:  
        intLst.append(int(item))  
    res = sum(intLst)/length  
    return res
```

```
print(average('11, 25, 52, 71, 19'))
```

Output:

```
1 #Daniel OBrien
2 #OSC 458
3 #HW 1a
4
5 def average(lst):
6     newList = lst.split(',')
7     length = len(newList)
8     intList = []
9     for item in newList:
10         intList.append(int(item))
11     res = sum(intList)/length
12     return res
13
14 print(average('1, 2, 3, 4, 5'))
```

Usage

Variable explorer Help Plots Files

Console 1/A

```
In [64]: runfile('/Users/danielobrien/Desktop/HW1.py', wdir='/Users/danielobrien/Desktop')
3.0
In [65]:
```

```
1 #Daniel OBrien
2 #OSC 458
3 #HW 1a
4
5 def average(lst):
6     newList = lst.split(',')
7     length = len(newList)
8     intList = []
9     for item in newList:
10         intList.append(int(item))
11     res = sum(intList)/length
12     return res
13
14 print(average('11, 25, 52, 71, 19'))
```

Usage

Variable explorer Help Plots Files

Console 1/A

```
In [64]: runfile('/Users/danielobrien/Desktop/HW1.py', wdir='/Users/danielobrien/Desktop')
3.0
In [65]: runfile('/Users/danielobrien/Desktop/HW1.py', wdir='/Users/danielobrien/Desktop')
35.6
In [66]:
```

```
In [64]: runfile('/Users/danielobrien/Desktop/HW1.py', wdir='/Users/
danielobrien/Desktop')
3.0

In [65]: runfile('/Users/danielobrien/Desktop/HW1.py', wdir='/Users/
danielobrien/Desktop')
35.6

In [66]:
```

B.

For this code, I created a function that received two different variables, TName and Lst. The first variable indicates what the other items of the list will be inserted into. I then removed commas from the list to better format the return statement. I used brackets and the .format option to create a return statement as requested.

Python Code:

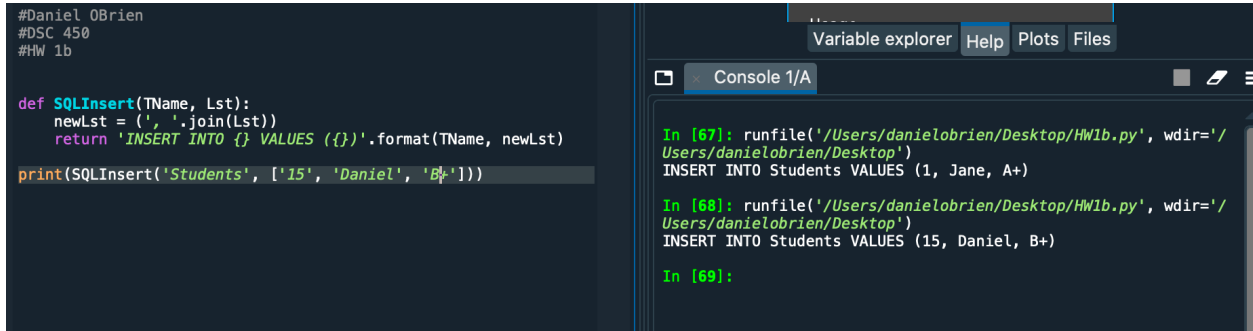
```
def SQLInsert(TName, Lst):
    newList = (' '.join(Lst))
    return 'INSERT INTO {} VALUES ({}).format(TName, newList)
```

```
print(SQLInsert('Students', ['1', 'Jane', 'A+']))
```

```
def SQLInsert(TName, Lst):
    newList = (' '.join(Lst))
    return 'INSERT INTO {} VALUES ({}).format(TName, newList)
```

```
print(SQLInsert('Students', ['15', 'Daniel', 'B+']))
```

Output:



The screenshot shows a Jupyter Notebook with two panels. The left panel contains a Python function `SQLInsert` and a `print` statement. The right panel shows the console output of the function being called twice.

```
#Daniel OBrien
#DSC 450
#HW 1b

def SQLInsert(TName, Lst):
    newLst = ('', '.join(Lst))
    return 'INSERT INTO {} VALUES {}'.format(TName, newLst)

print(SQLInsert('Students', ['15', 'Daniel', 'B+']))
```

Console 1/A

```
In [67]: runfile('/Users/danielobrien/Desktop/HW1b.py', wdir='/Users/danielobrien/Desktop')
INSERT INTO Students VALUES (1, Jane, A+)

In [68]: runfile('/Users/danielobrien/Desktop/HW1b.py', wdir='/Users/danielobrien/Desktop')
INSERT INTO Students VALUES (15, Daniel, B+)

In [69]:
```

```
In [58]: runfile('/Users/danielobrien/Desktop/HW1b.py', wdir='/Users/danielobrien/Desktop')
INSERT INTO Students VALUES (1, Jane, A+)

In [59]: runfile('/Users/danielobrien/Desktop/HW1b.py', wdir='/Users/danielobrien/Desktop')
INSERT INTO Students VALUES (15, Daniel, B+)

In [60]:
```