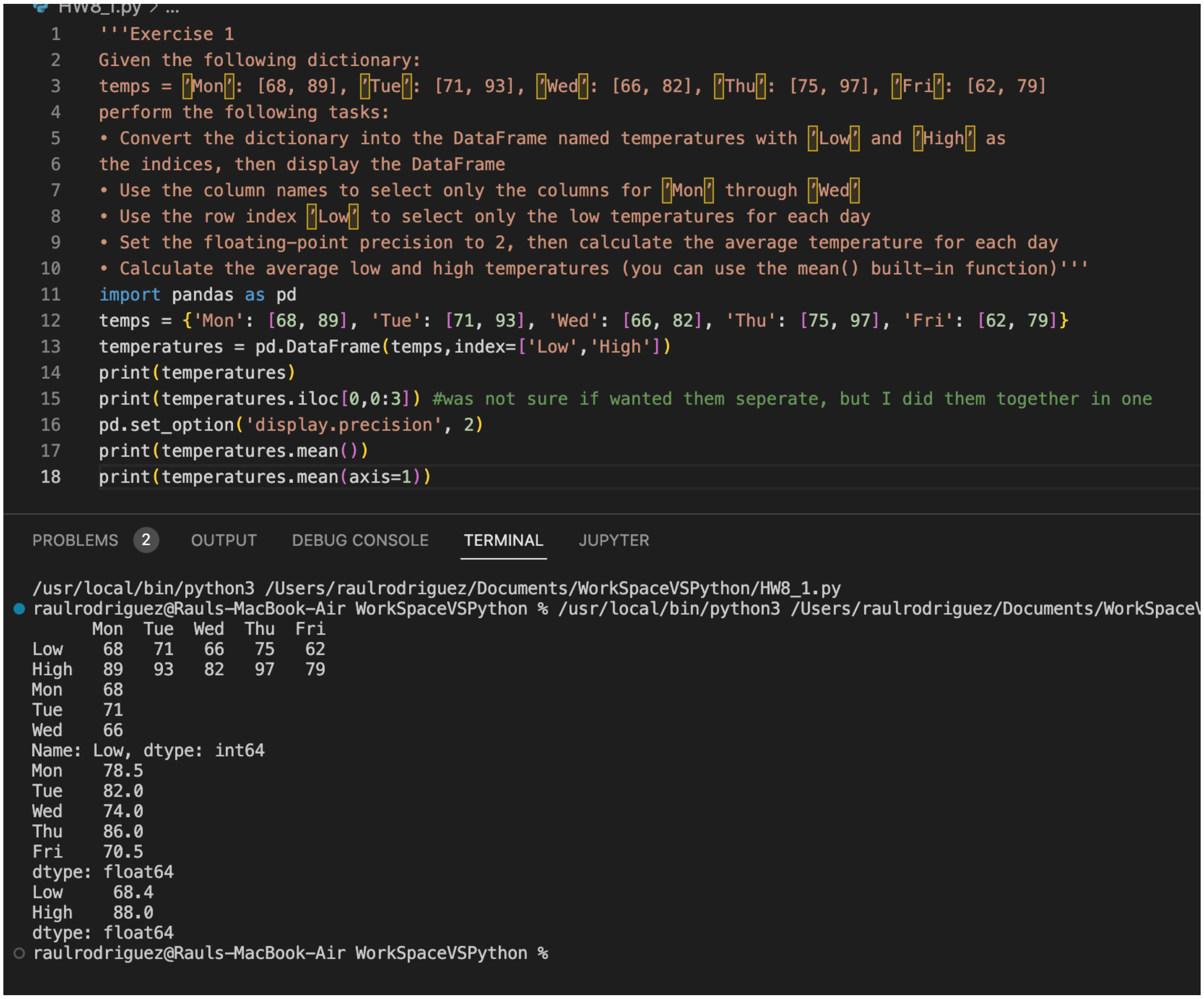
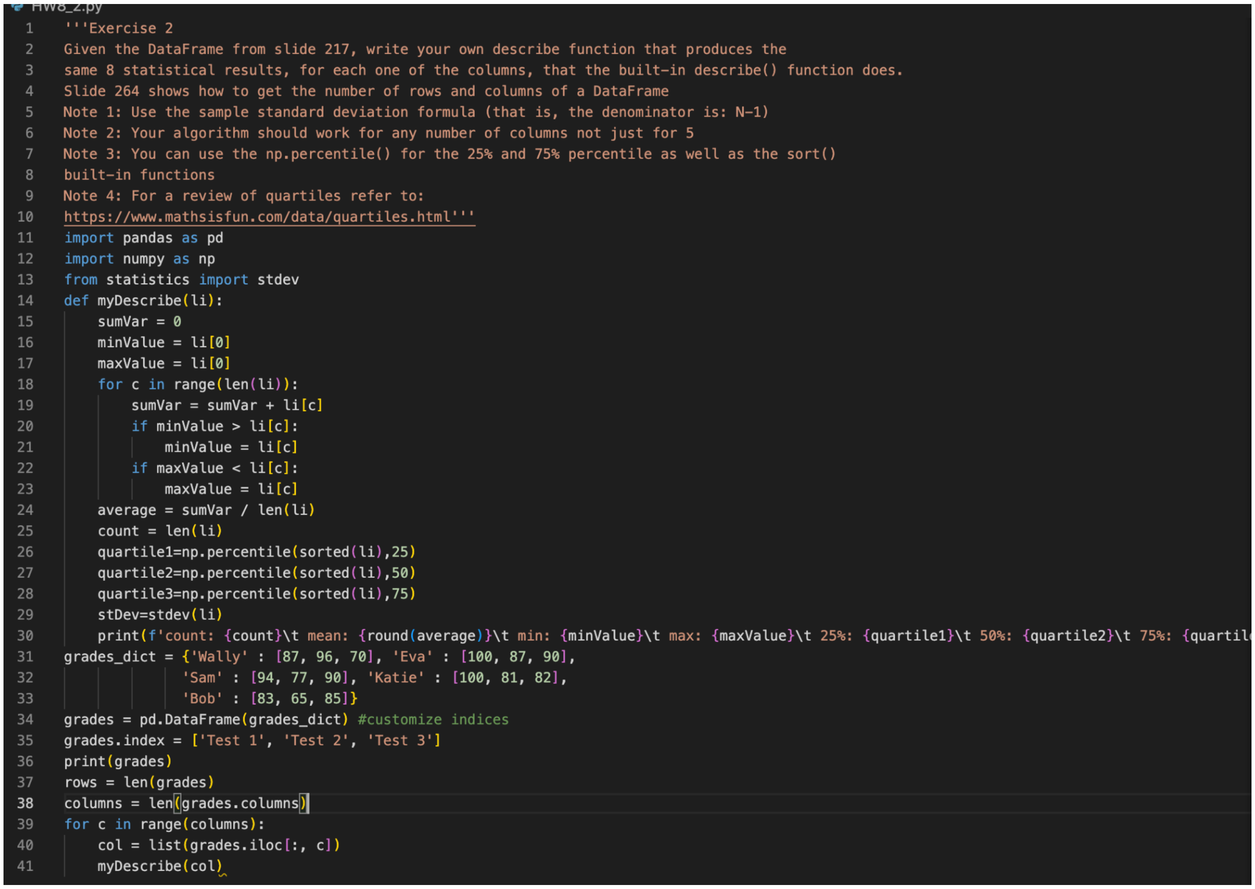
Raul Rodriguez

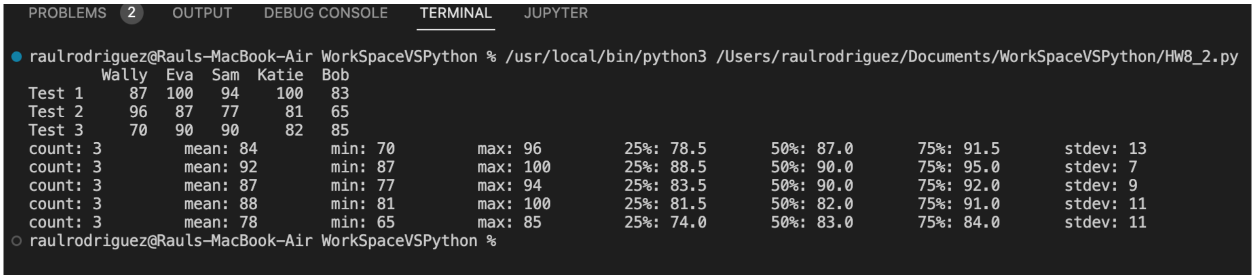


```python
'''Exercise 1
Given the following dictionary:
temps = ’Mon’: [68, 89], ’Tue’: [71, 93], ’Wed’: [66, 82], ’Thu’: [75, 97], ’Fri’: [62, 79]
perform the following tasks:
• Convert the dictionary into the DataFrame named temperatures with ’Low’ and ’High’ as
the indices, then display the DataFrame
• Use the column names to select only the columns for ’Mon’ through ’Wed’
• Use the row index ’Low’ to select only the low temperatures for each day
• Set the floating-point precision to 2, then calculate the average temperature for each day
• Calculate the average low and high temperatures (you can use the mean() built-in function)'''
import pandas as pd
temps = {'Mon': [68, 89], 'Tue': [71, 93], 'Wed': [66, 82], 'Thu': [75, 97], 'Fri': [62, 79]}
temperatures = pd.DataFrame(temps,index=['Low','High'])
print(temperatures)
print(temperatures.iloc[0,0:3]) #was not sure if wanted them seperate, but I did them together in one
pd.set_option('display.precision', 2)
print(temperatures.mean())
print(temperatures.mean(axis=1))
```

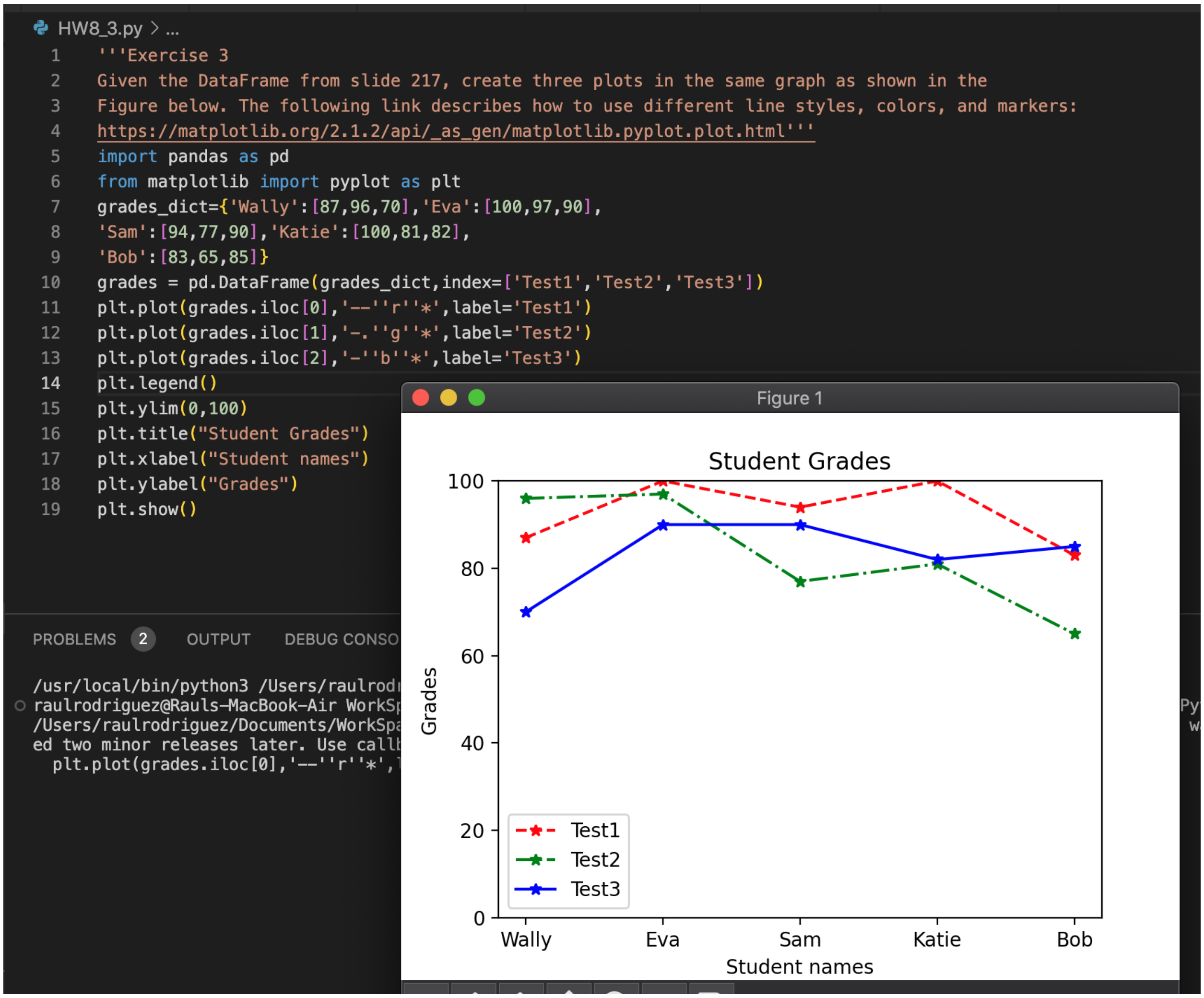
```
/usr/local/bin/python3 /Users/raulrodriguez/Documents/WorkSpaceVSPython/HW8_1.py
raulrodriguez@Rauls-MacBook-Air WorkSpaceVSPython % /usr/local/bin/python3 /Users/raulrodriguez/Documents/WorkSpaceV
      Mon  Tue  Wed  Thu  Fri
Low    68   71   66   75   62
High   89   93   82   97   79
Mon    68
Tue    71
Wed    66
Name: Low, dtype: int64
Mon    78.5
Tue    82.0
Wed    74.0
Thu    86.0
Fri    70.5
dtype: float64
Low     68.4
High    88.0
dtype: float64
raulrodriguez@Rauls-MacBook-Air WorkSpaceVSPython %
```



```python
'''Exercise 2
Given the DataFrame from slide 217, write your own describe function that produces the
same 8 statistical results, for each one of the columns, that the built-in describe() function does.
Slide 264 shows how to get the number of rows and columns of a DataFrame
Note 1: Use the sample standard deviation formula (that is, the denominator is: N-1)
Note 2: Your algorithm should work for any number of columns not just for 5
Note 3: You can use the np.percentile() for the 25% and 75% percentile as well as the sort()
built-in functions
Note 4: For a review of quartiles refer to:
https://www.mathsisfun.com/data/quartiles.html'''
import pandas as pd
import numpy as np
from statistics import stdev
def myDescribe(li):
    sumVar = 0
    minValue = li[0]
    maxValue = li[0]
    for c in range(len(li)):
        sumVar = sumVar + li[c]
        if minValue > li[c]:
            minValue = li[c]
        if maxValue < li[c]:
            maxValue = li[c]
    average = sumVar / len(li)
    count = len(li)
    quartile1=np.percentile(sorted(li),25)
    quartile2=np.percentile(sorted(li),50)
    quartile3=np.percentile(sorted(li),75)
    stDev=stdev(li)
    print(f'count: {count}\t mean: {round(average)}\t min: {minValue}\t max: {maxValue}\t 25%: {quartile1}\t 50%: {quartile2}\t 75%: {quartil
grades_dict = {'Wally' : [87, 96, 70], 'Eva' : [100, 87, 90],
               'Sam' : [94, 77, 90], 'Katie' : [100, 81, 82],
               'Bob' : [83, 65, 85]}
grades = pd.DataFrame(grades_dict) #customize indices
grades.index = ['Test 1', 'Test 2', 'Test 3']
print(grades)
rows = len(grades)
columns = len(grades.columns)
for c in range(columns):
    col = list(grades.iloc[:, c])
    myDescribe(col)
```



```
● raulrodriguez@Rauls-MacBook-Air WorkSpaceVSPython % /usr/local/bin/python3 /Users/raulrodriguez/Documents/WorkSpaceVSPython/HW8_2.py
          Wally  Eva  Sam  Katie  Bob
  Test 1     87  100   94    100   83
  Test 2     96   87   77     81   65
  Test 3     70   90   90     82   85
  count: 3        mean: 84        min: 70         max: 96         25%: 78.5       50%: 87.0       75%: 91.5       stdev: 13
  count: 3        mean: 92        min: 87         max: 100        25%: 88.5       50%: 90.0       75%: 95.0       stdev: 7
  count: 3        mean: 87        min: 77         max: 94         25%: 83.5       50%: 90.0       75%: 92.0       stdev: 9
  count: 3        mean: 88        min: 81         max: 100        25%: 81.5       50%: 82.0       75%: 91.0       stdev: 11
  count: 3        mean: 78        min: 65         max: 85         25%: 74.0       50%: 83.0       75%: 84.0       stdev: 11
○ raulrodriguez@Rauls-MacBook-Air WorkSpaceVSPython %
```



```python
'''Exercise 3
Given the DataFrame from slide 217, create three plots in the same graph as shown in the
Figure below. The following link describes how to use different line styles, colors, and markers:
https://matplotlib.org/2.1.2/api/_as_gen/matplotlib.pyplot.plot.html'''
import pandas as pd
from matplotlib import pyplot as plt
grades_dict={'Wally':[87,96,70],'Eva':[100,97,90],
'Sam':[94,77,90],'Katie':[100,81,82],
'Bob':[83,65,85]}
grades = pd.DataFrame(grades_dict,index=['Test1','Test2','Test3'])
plt.plot(grades.iloc[0],'--''r''*',label='Test1')
plt.plot(grades.iloc[1],'-.''g''*',label='Test2')
plt.plot(grades.iloc[2],'-''b''*',label='Test3')
plt.legend()
plt.ylim(0,100)
plt.title("Student Grades")
plt.xlabel("Student names")
plt.ylabel("Grades")
plt.show()
```

```
/usr/local/bin/python3 /Users/raulrodr
○ raulrodriguez@Rauls-MacBook-Air WorkSp
/Users/raulrodriguez/Documents/WorkSpa
ed two minor releases later. Use callb
  plt.plot(grades.iloc[0],'--''r''*',
```