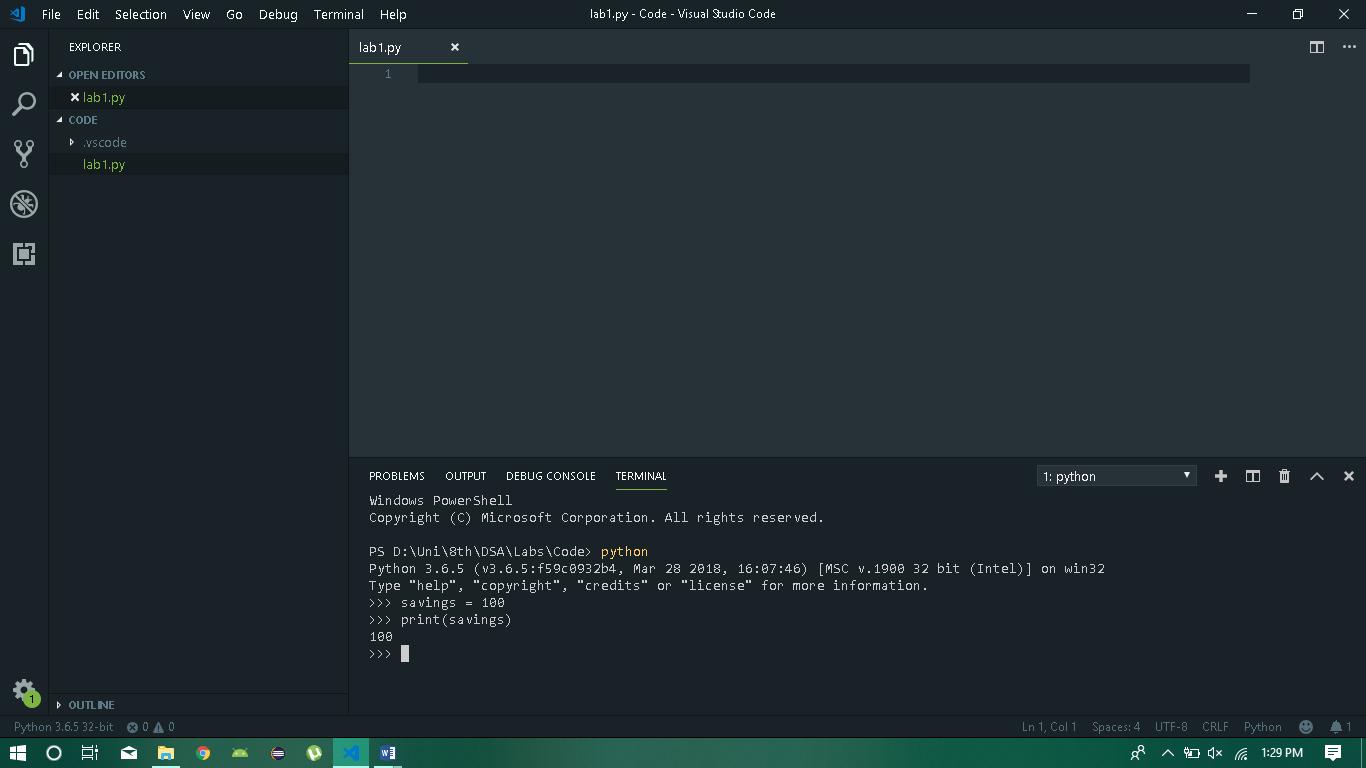
# Practical session 1: Introduction to Python for Data Sciences

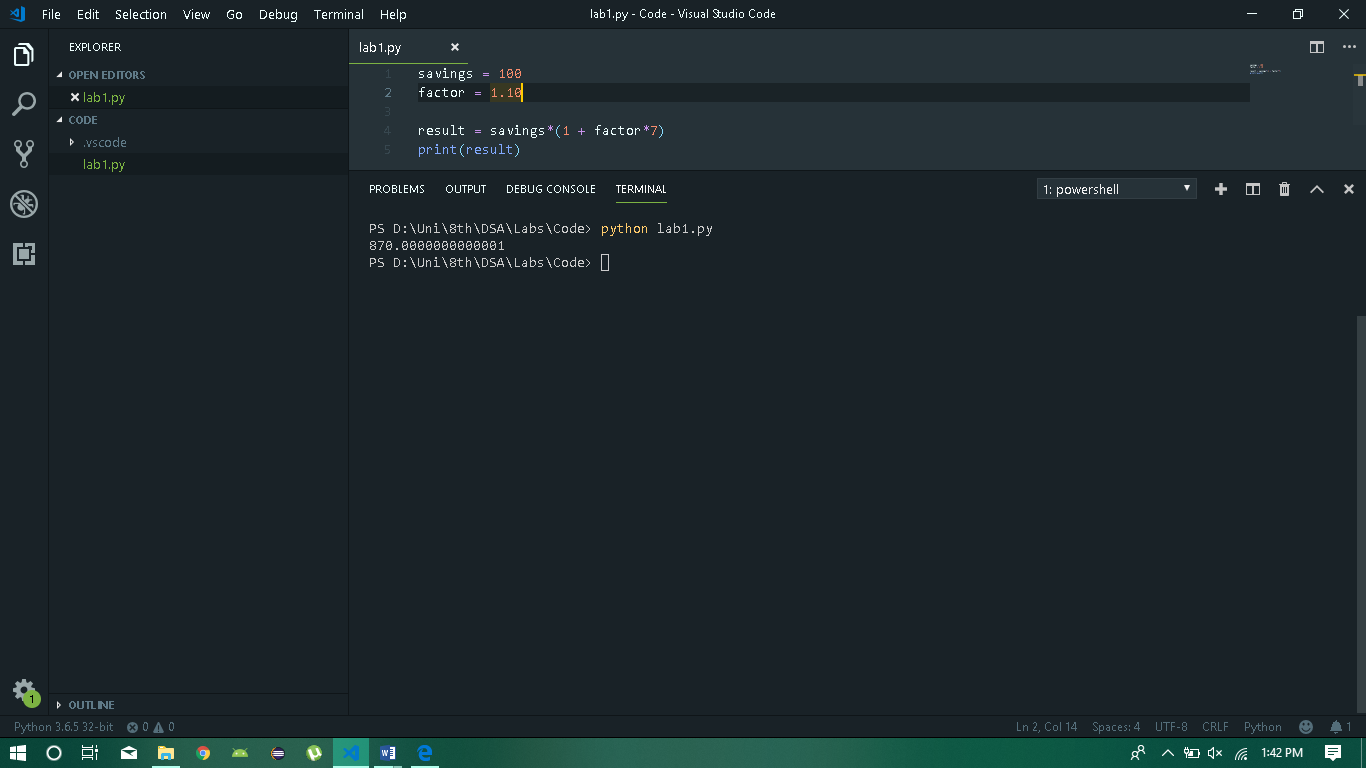
Question 1:

* Create a variable savings with the value 100.
* Check out this variable by typing print(savings) in the script.



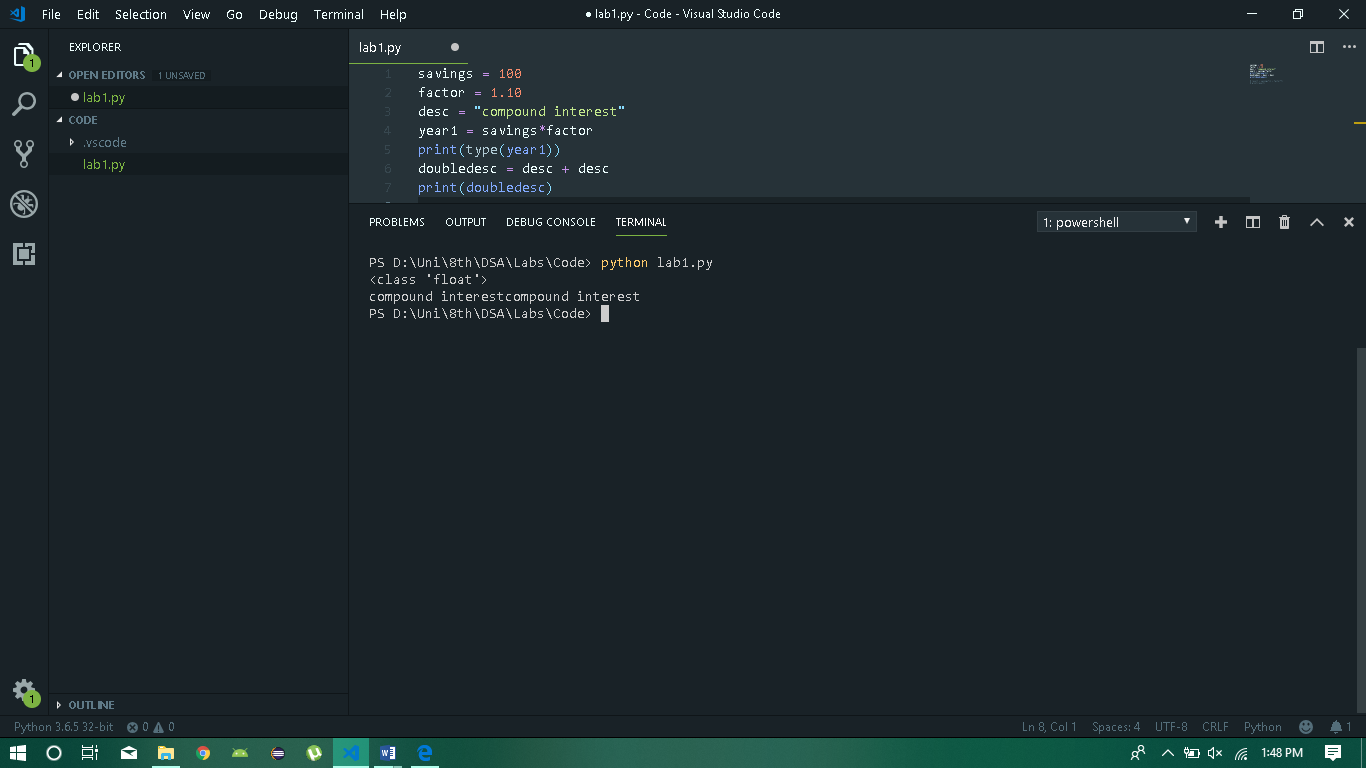
Question 2:

* Create a variable factor, equal to 1.10.
* Use savings and factor to calculate the amount of money you end up with after 7 years. Store the result in a new variable, result.
* Print out the value of result.



Question 3:

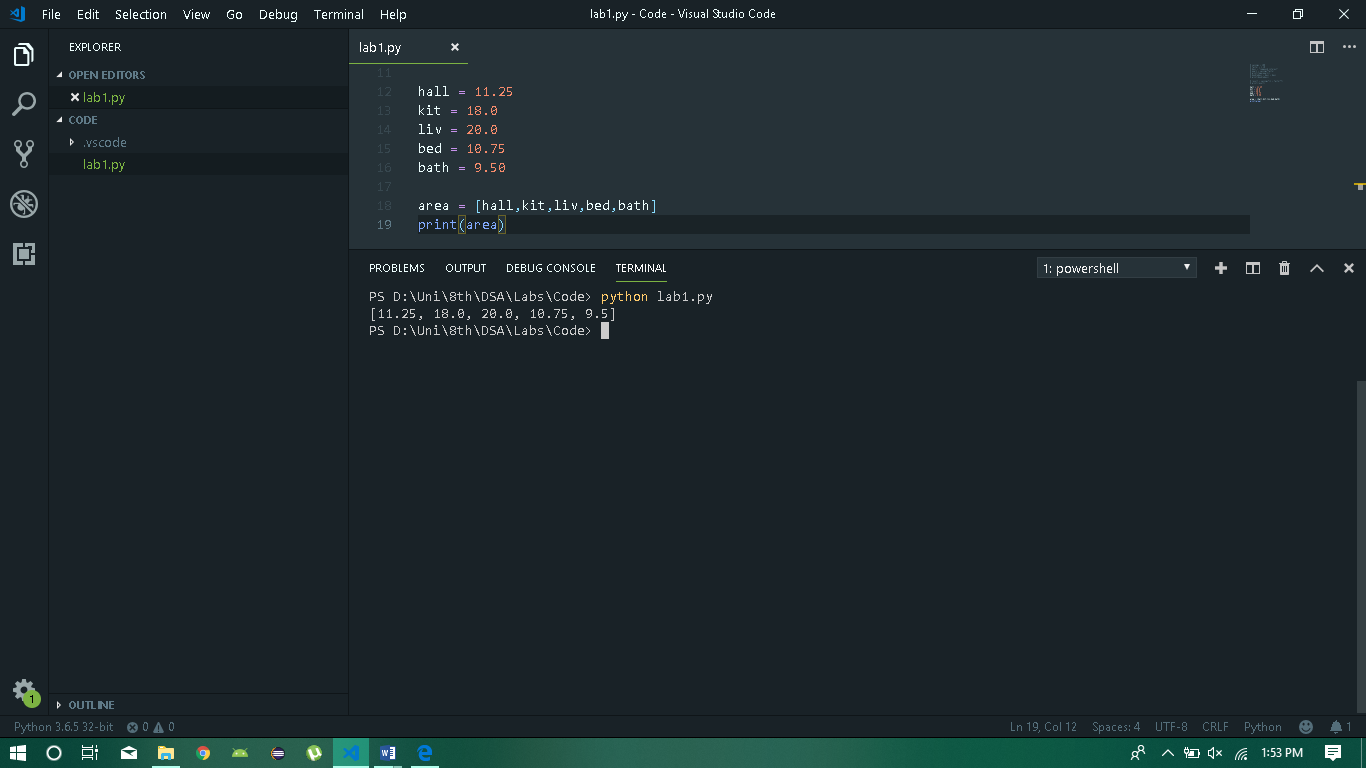
* Calculate the product of savings and factor. Store the result in year1.
* What do you think the resulting type will be? Find out by printing out the type of year1.
* Calculate the sum of desc(string) and desc and store the result in a new variable doubledesc.
* Print out doubledesc. Did you expect this?
* savings = 100, factor = 1.1, desc = "compound interest"



### Python Lists

Question 4:

* Create a list, areas, that contains the area of the hallway (hall), kitchen (kit), living room (liv), bedroom (bed) and bathroom (bath), in this order. Use the predefined variables.
* Print areas with the [print()](https://docs.python.org/3/library/functions.html#print) function.



Question 5:

* Finish the line of code that creates the areas list such that the list first contains the name of each room as a string, and then its area. More specifically, add the strings "hallway", "kitchen" and "bedroom" at the appropriate locations.
* Print areas again; is the printout more informative this time?

**# area variables (in square meters)**

hall = 11.25

kit = 18.0

liv = 20.0

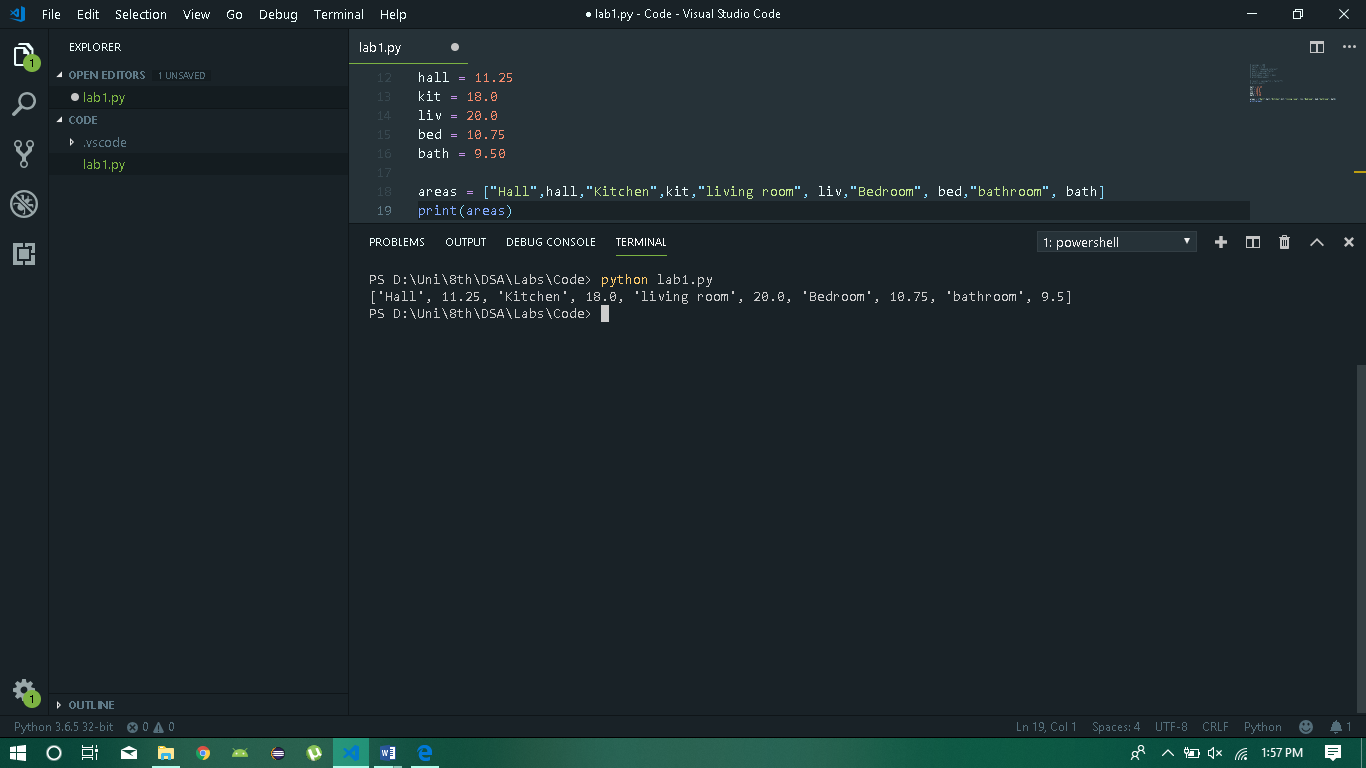
bed = 10.75

bath = 9.50

**# Adapt list areas**

areas = [hall, kit, "living room", liv, bed, "bathroom", bath]

**# Print areas**



Question 6:

As a data scientist, you'll often be dealing with a lot of data, and it will make sense to group some of this data.

Instead of creating a flat list containing strings and floats, representing the names and areas of the rooms in your house, you can create a list of lists. The script on the right can already give you an idea.

* Finish the list of lists so that it also contains the bedroom and bathroom data. Make sure you enter these in order!
* Print out house; does this way of structuring your data make more sense?
* Print out the type of house. Are you still dealing with a list?

**# area variables (in square meters)**

hall = 11.25

kit = 18.0

liv = 20.0

bed = 10.75

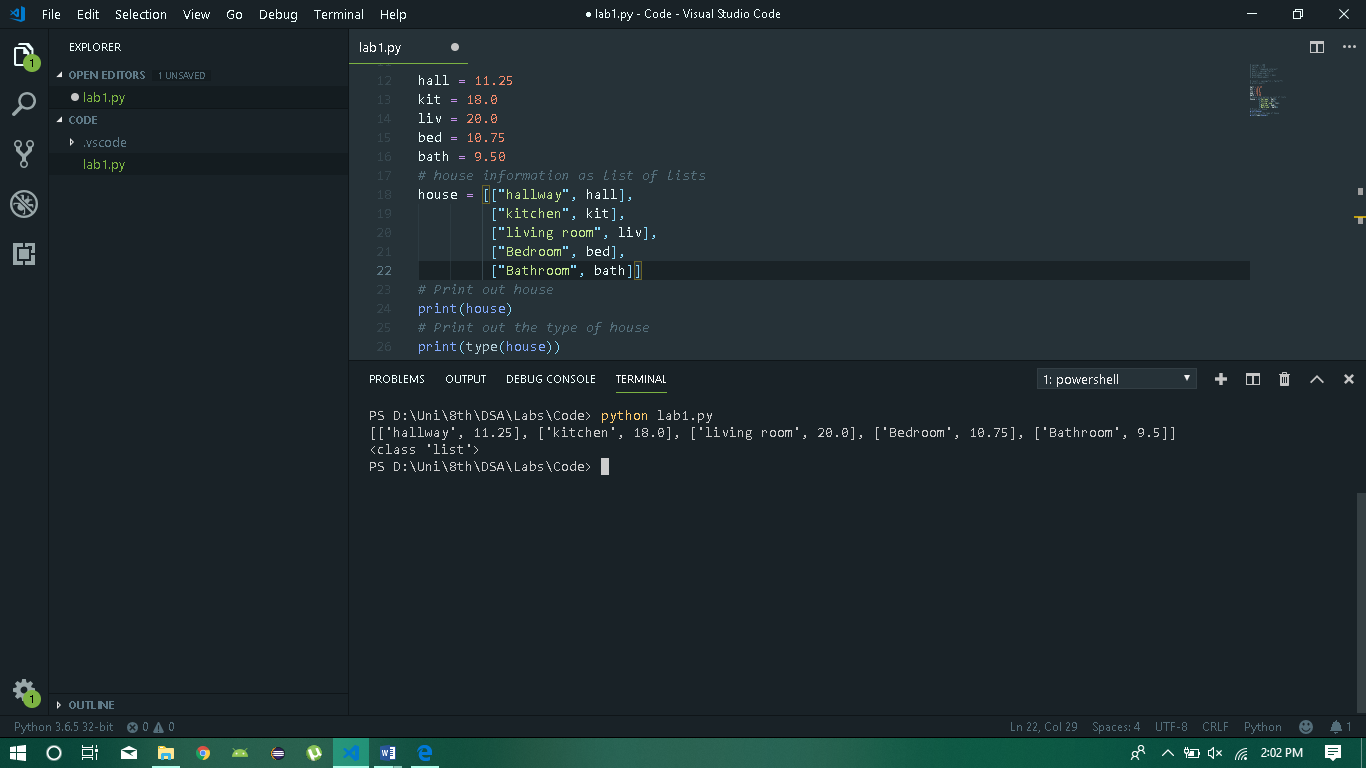
bath = 9.50

**# house information as list of lists**

house = [["hallway", hall],

["kitchen", kit],

["living room", liv]]



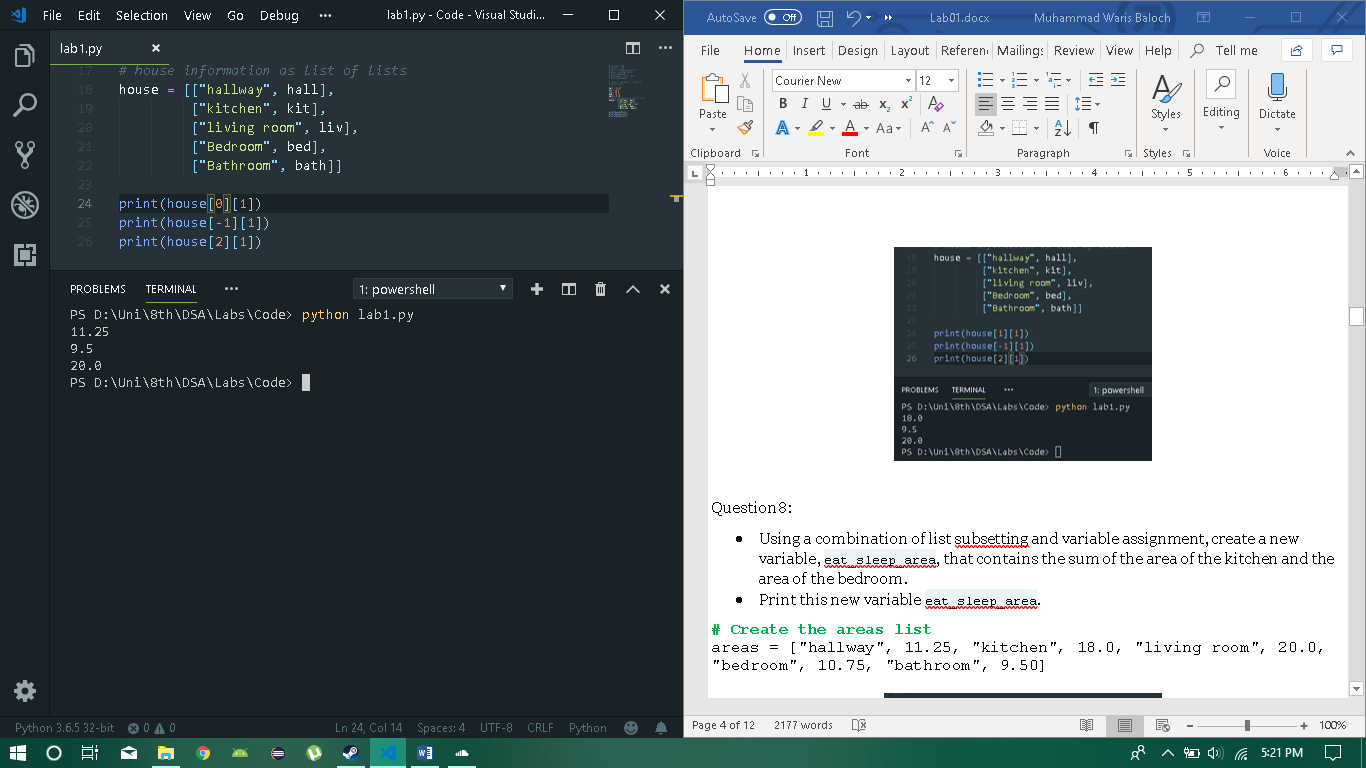
### List sub-setting

Question 7:

* Print out the second element from the areas list, so 11.25.
* Subset and print out the last element of areas, being 9.50. Using a negative index makes sense here!
* Select the number representing the area of the living room and print it out.

**# Create the areas list**

areas = ["hallway", 11.25, "kitchen", 18.0, "living room", 20.0, "bedroom", 10.75, "bathroom", 9.50]

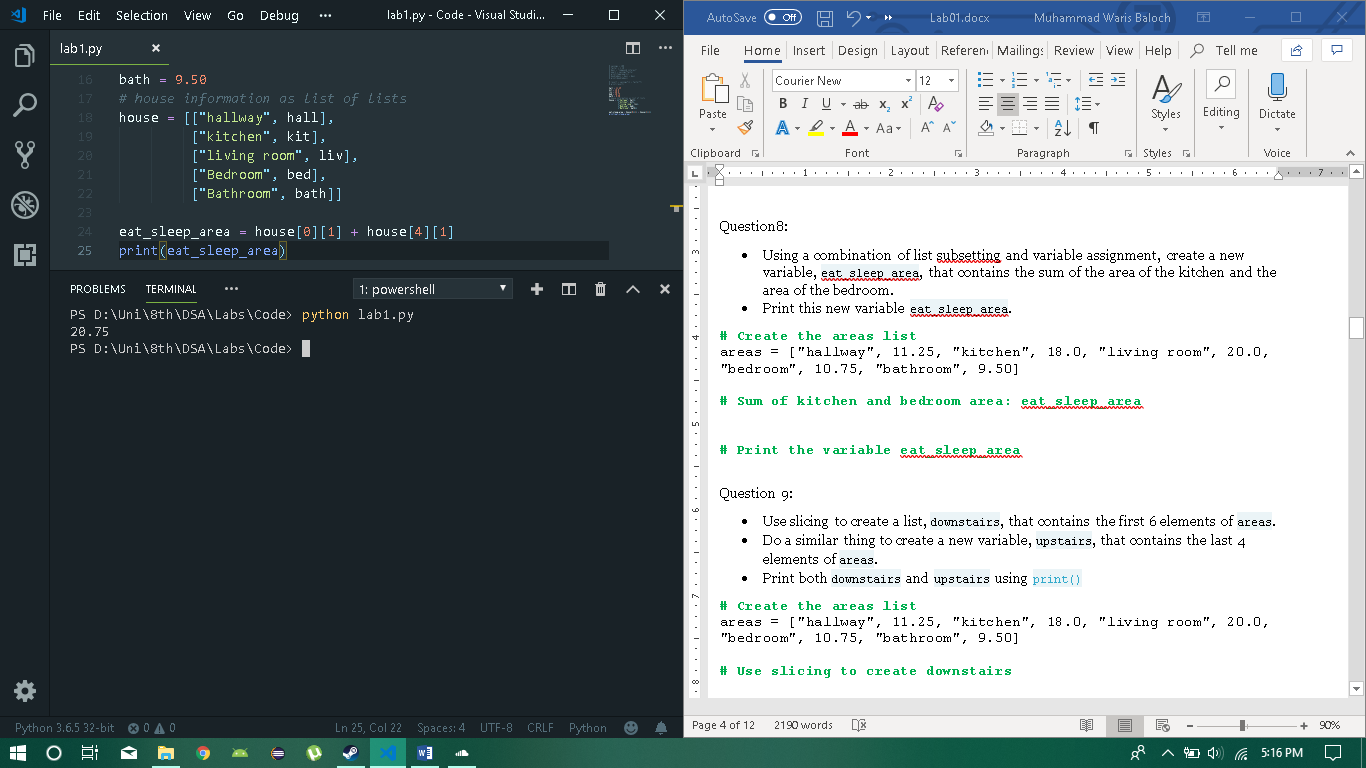


Question8:

* Using a combination of list subsetting and variable assignment, create a new variable, eat\_sleep\_area, that contains the sum of the area of the kitchen and the area of the bedroom.
* Print this new variable eat\_sleep\_area.

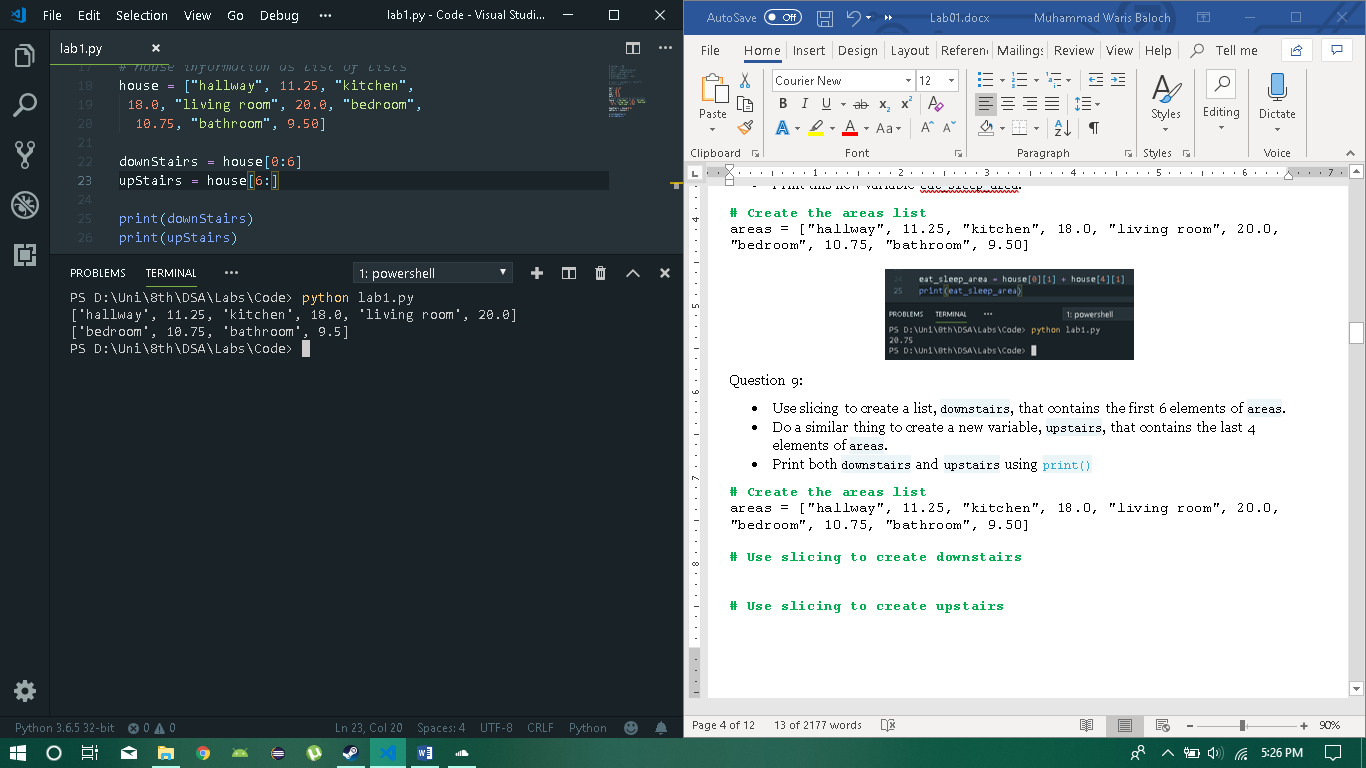
**# Create the areas list**

areas = ["hallway", 11.25, "kitchen", 18.0, "living room", 20.0, "bedroom", 10.75, "bathroom", 9.50]



Question 9:

* Use slicing to create a list, downstairs, that contains the first 6 elements of areas.
* Do a similar thing to create a new variable, upstairs, that contains the last 4 elements of areas.
* Print both downstairs and upstairs using [print()](https://docs.python.org/3/library/functions.html#print)



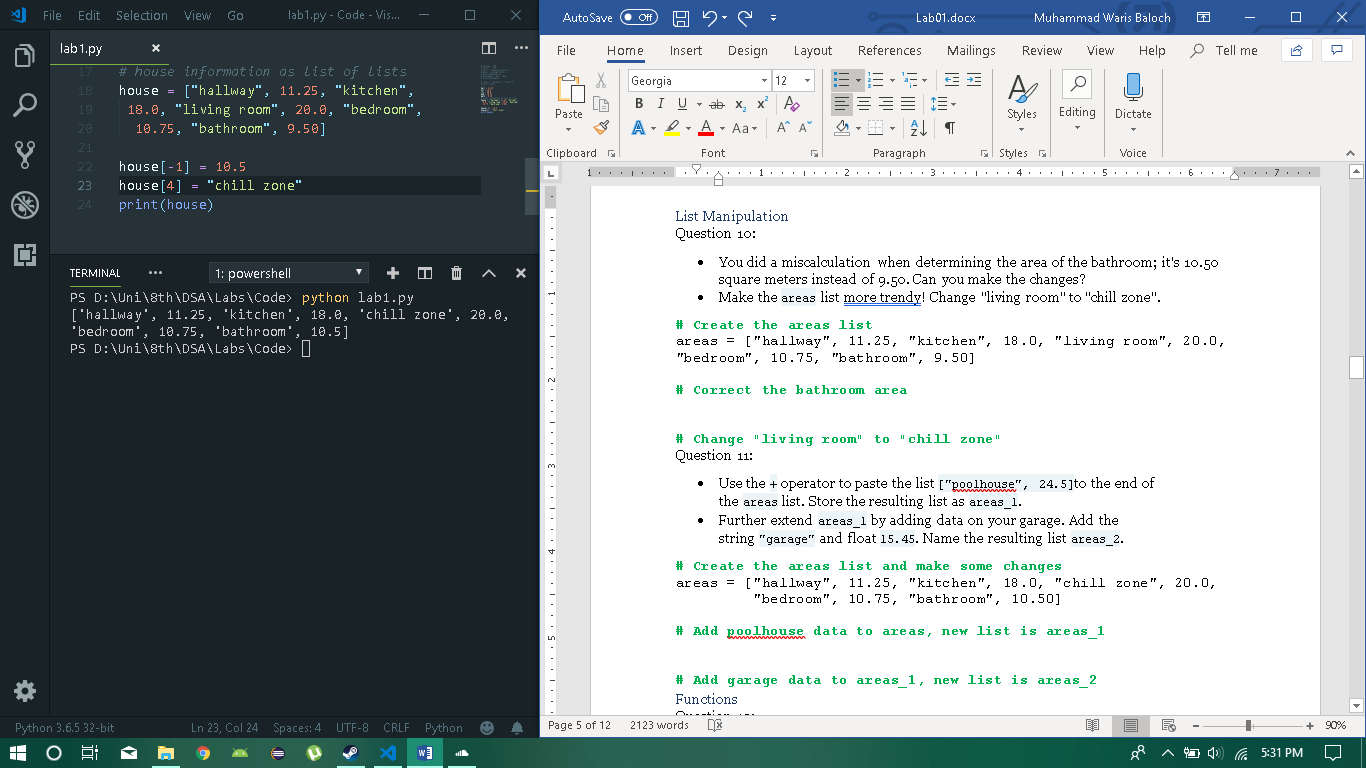
### List Manipulation

Question 10:

* You did a miscalculation when determining the area of the bathroom; it's 10.50 square meters instead of 9.50. Can you make the changes?
* Make the areas list more trendy! Change "living room" to "chill zone".

**# Create the areas list**

areas = ["hallway", 11.25, "kitchen", 18.0, "living room", 20.0, "bedroom", 10.75, "bathroom", 9.50]



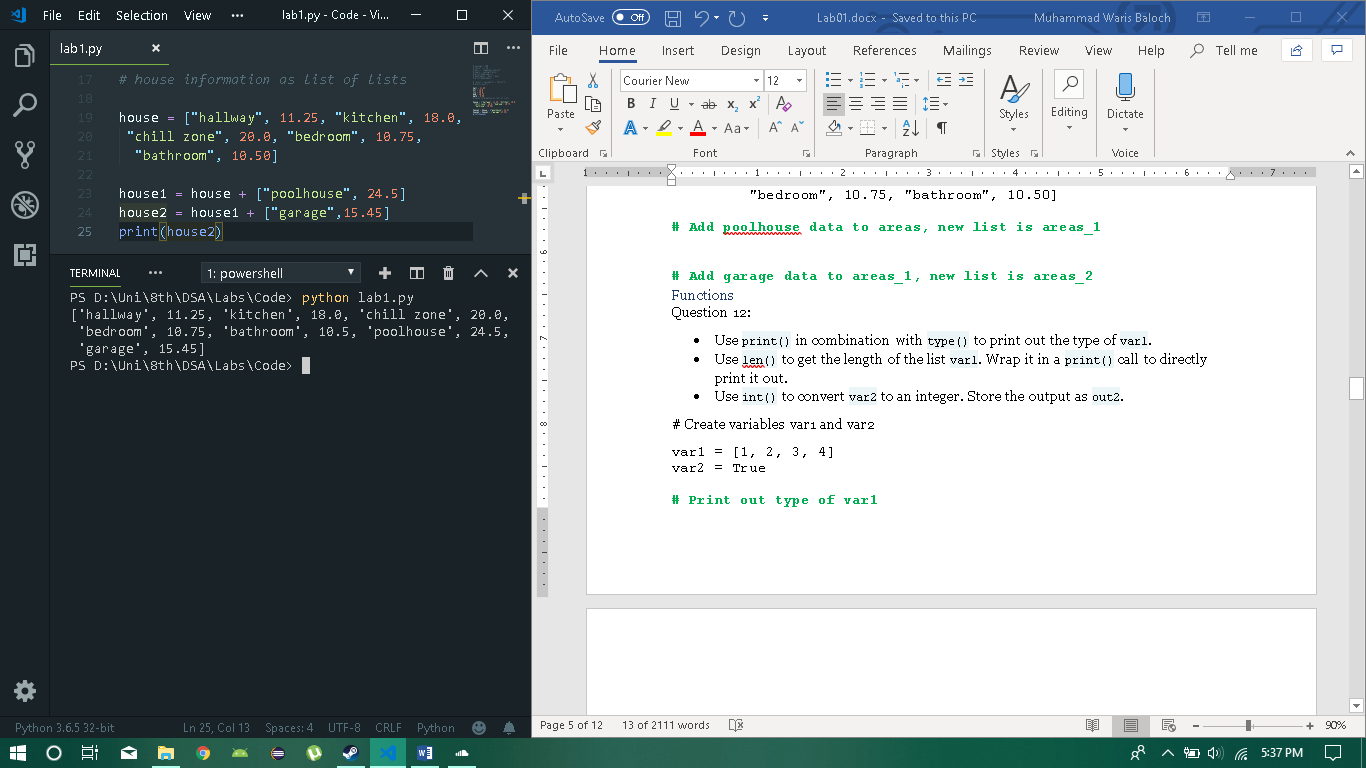
Question 11:

* Use the + operator to paste the list ["poolhouse", 24.5]to the end of the areas list. Store the resulting list as areas\_1.
* Further extend areas\_1 by adding data on your garage. Add the string "garage" and float 15.45. Name the resulting list areas\_2.

**# Create the areas list and make some changes**

areas = ["hallway", 11.25, "kitchen", 18.0, "chill zone", 20.0,

"bedroom", 10.75, "bathroom", 10.50]



### Functions

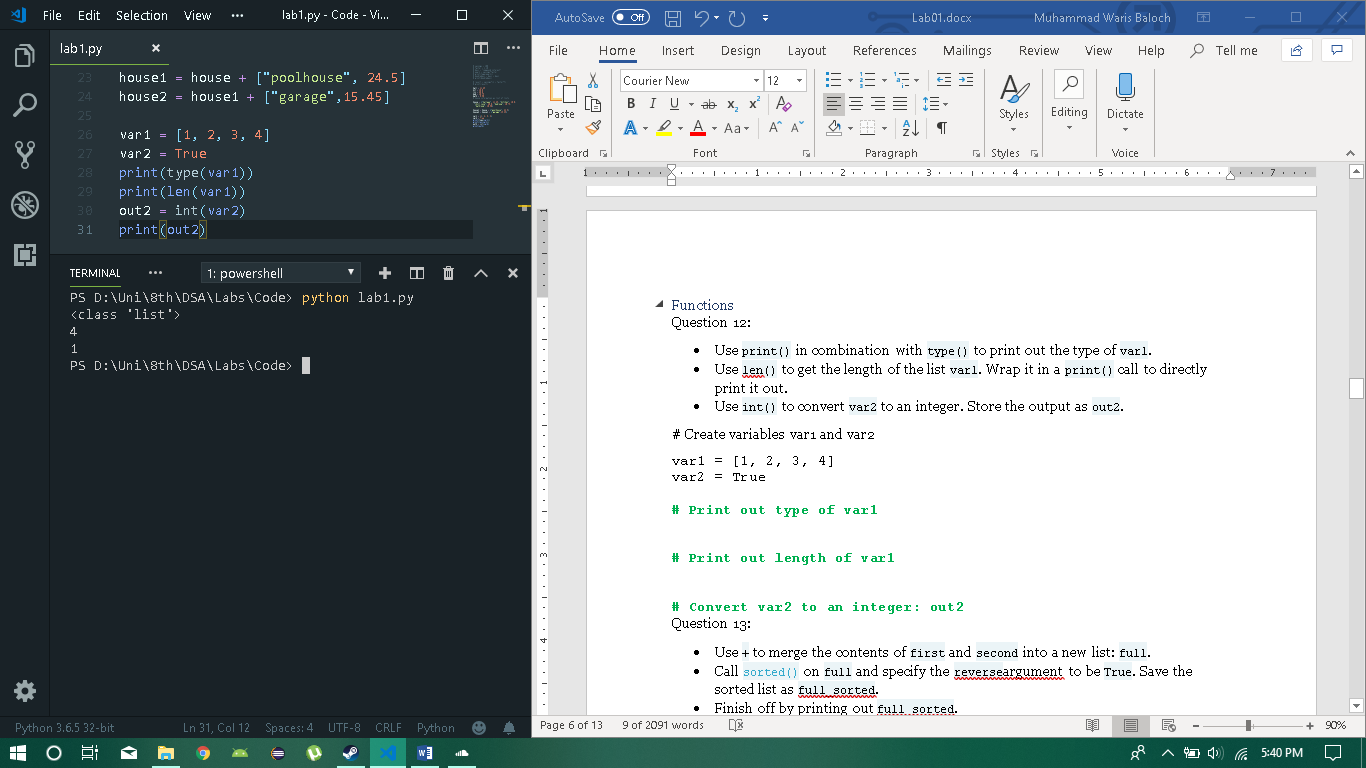
Question 12:

* Use [print()](https://docs.python.org/3/library/functions.html#print) in combination with [type()](https://docs.python.org/3/library/functions.html#type) to print out the type of var1.
* Use [len()](https://docs.python.org/3/library/functions.html" \l "len" \t "_blank) to get the length of the list var1. Wrap it in a [print()](https://docs.python.org/3/library/functions.html#print) call to directly print it out.
* Use [int()](https://docs.python.org/3/library/functions.html#int) to convert var2 to an integer. Store the output as out2.

# Create variables var1 and var2

var1 = [1, 2, 3, 4]

var2 = True



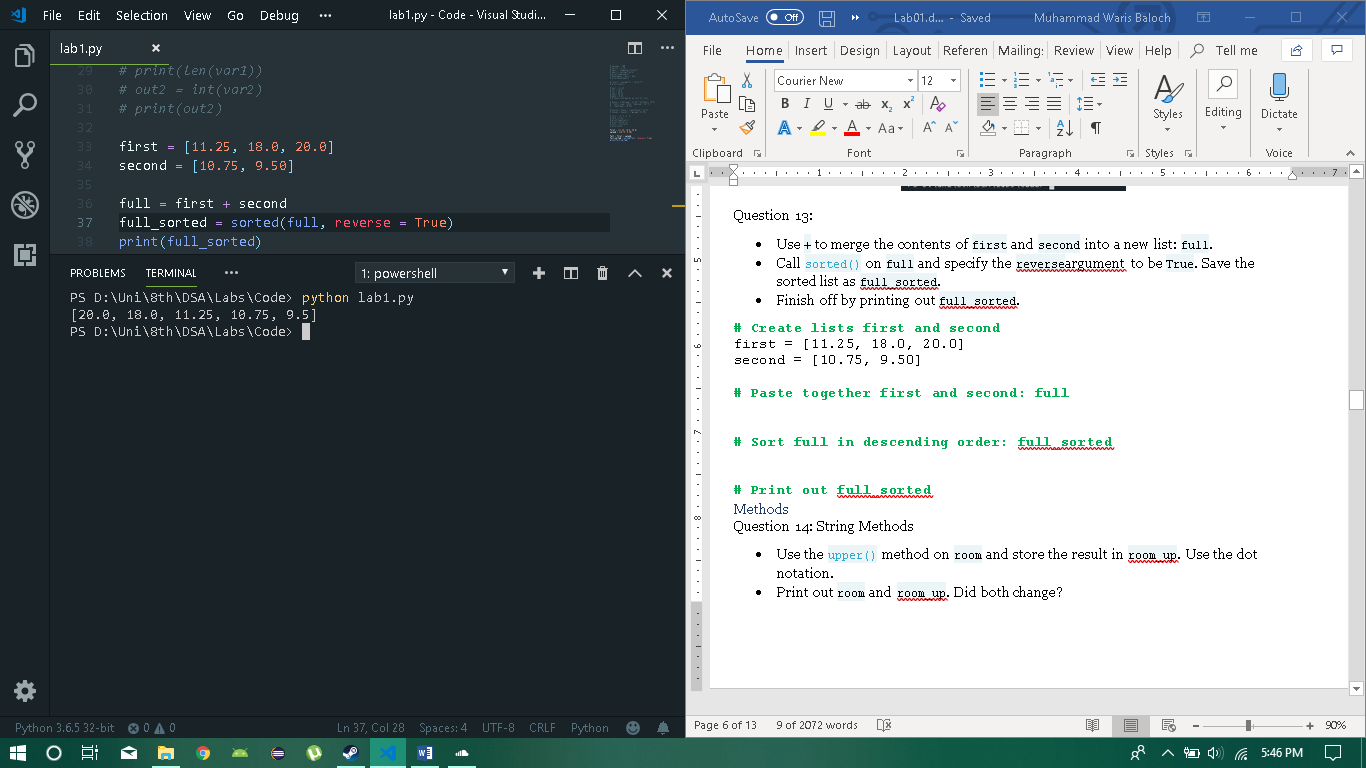
Question 13:

* Use + to merge the contents of first and second into a new list: full.
* Call [sorted()](https://docs.python.org/3/library/functions.html#sorted) on full and specify the reverseargument to be True. Save the sorted list as full\_sorted.
* Finish off by printing out full\_sorted.

**# Create lists first and second**

first = [11.25, 18.0, 20.0]

second = [10.75, 9.50]



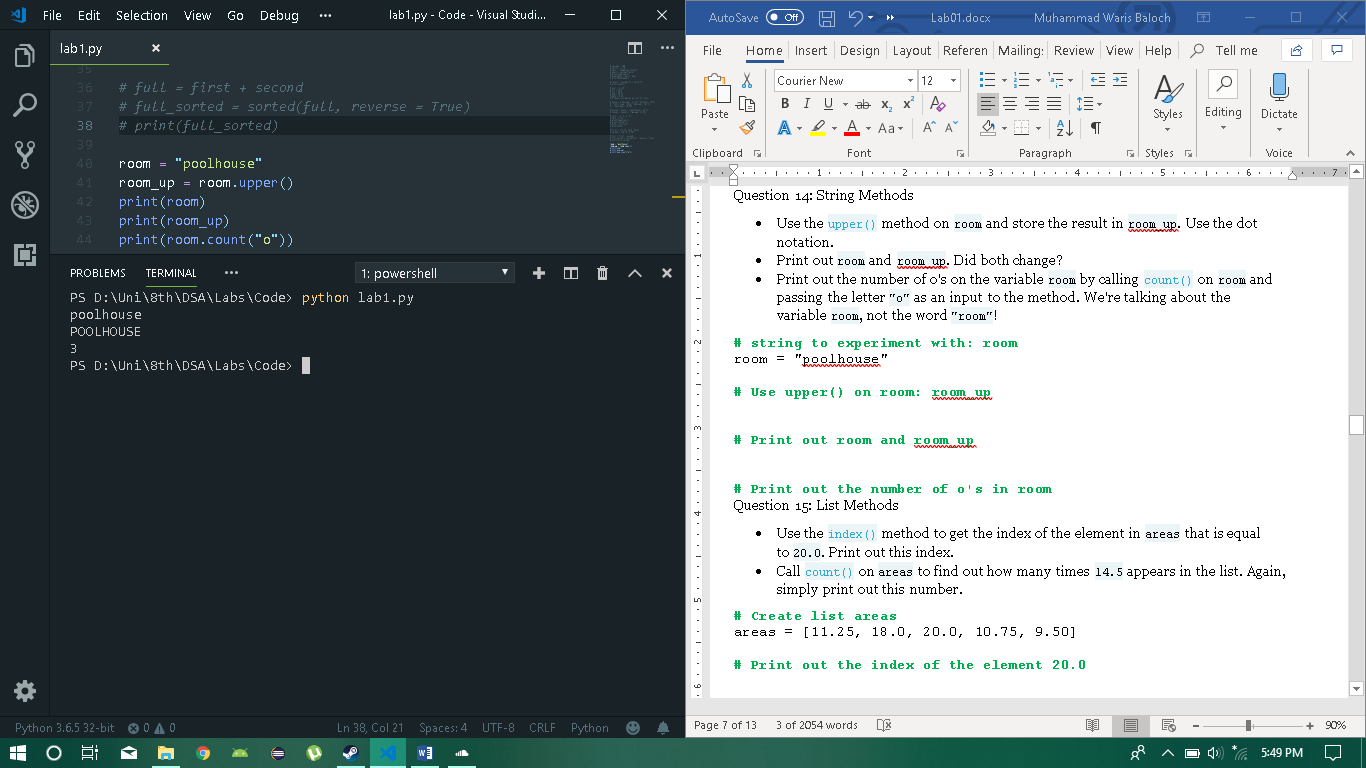
### Methods

Question 14: String Methods

* Use the [upper()](https://docs.python.org/3/library/stdtypes.html#str.upper) method on room and store the result in room\_up. Use the dot notation.
* Print out room and room\_up. Did both change?
* Print out the number of o's on the variable room by calling [count()](https://docs.python.org/3/library/stdtypes.html#str.count) on room and passing the letter "o" as an input to the method. We're talking about the variable room, not the word "room"!

**# string to experiment with: room**

room = "poolhouse"

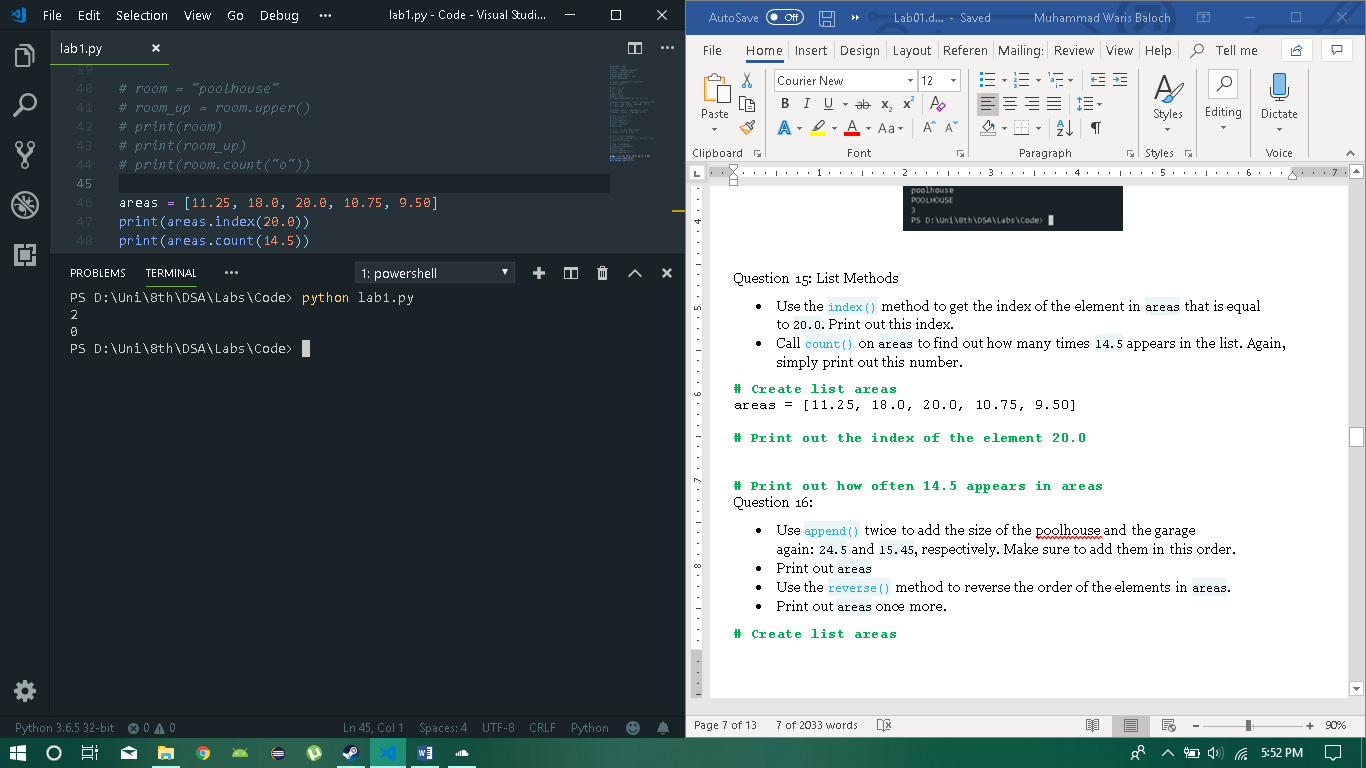


Question 15: List Methods

* Use the [index()](https://docs.python.org/3/library/stdtypes.html#str.index) method to get the index of the element in areas that is equal to 20.0. Print out this index.
* Call [count()](https://docs.python.org/3/library/stdtypes.html#str.count) on areas to find out how many times 14.5 appears in the list. Again, simply print out this number.

**# Create list areas**

areas = [11.25, 18.0, 20.0, 10.75, 9.50]

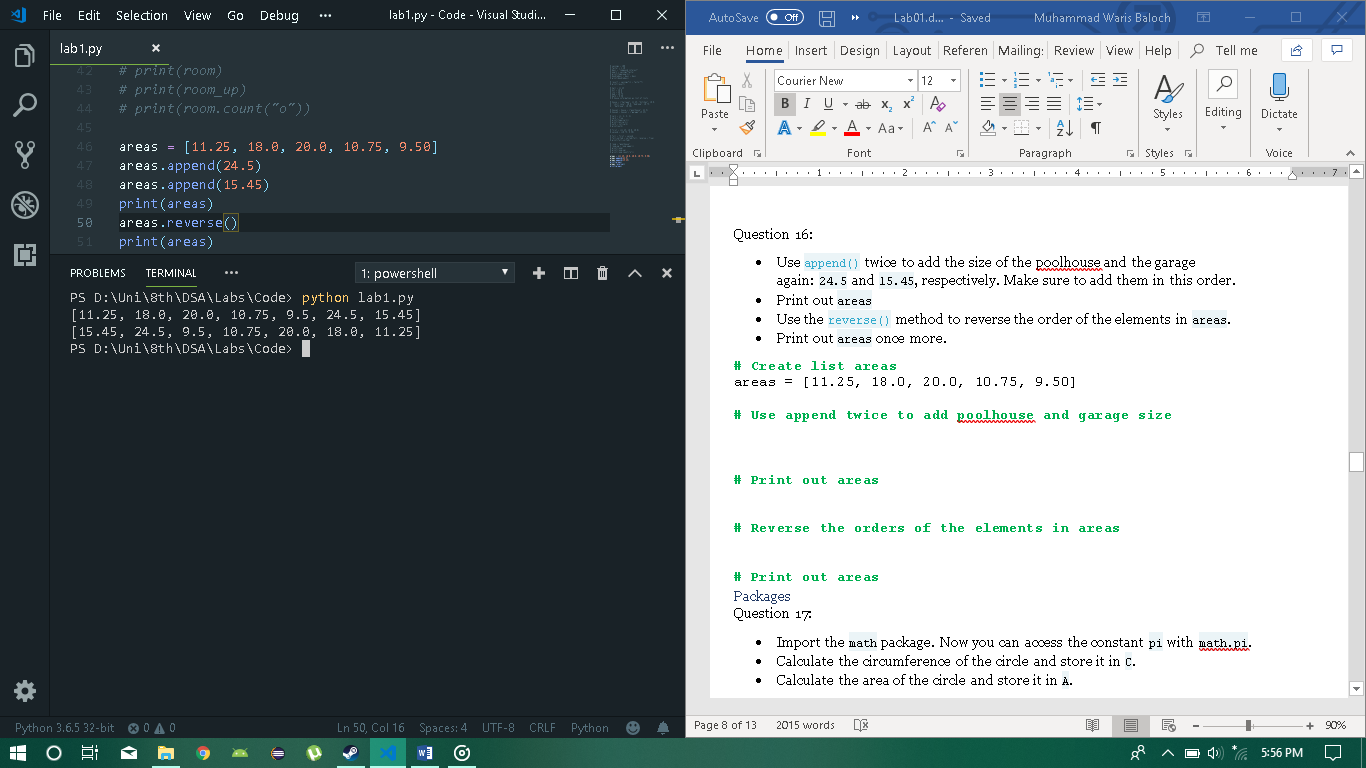


Question 16:

* Use [append()](https://docs.python.org/3/library/stdtypes.html#typesseq-mutable) twice to add the size of the poolhouse and the garage again: 24.5 and 15.45, respectively. Make sure to add them in this order.
* Print out areas
* Use the [reverse()](https://docs.python.org/3/library/stdtypes.html#typesseq-mutable) method to reverse the order of the elements in areas.
* Print out areas once more.

**# Create list areas**

areas = [11.25, 18.0, 20.0, 10.75, 9.50]



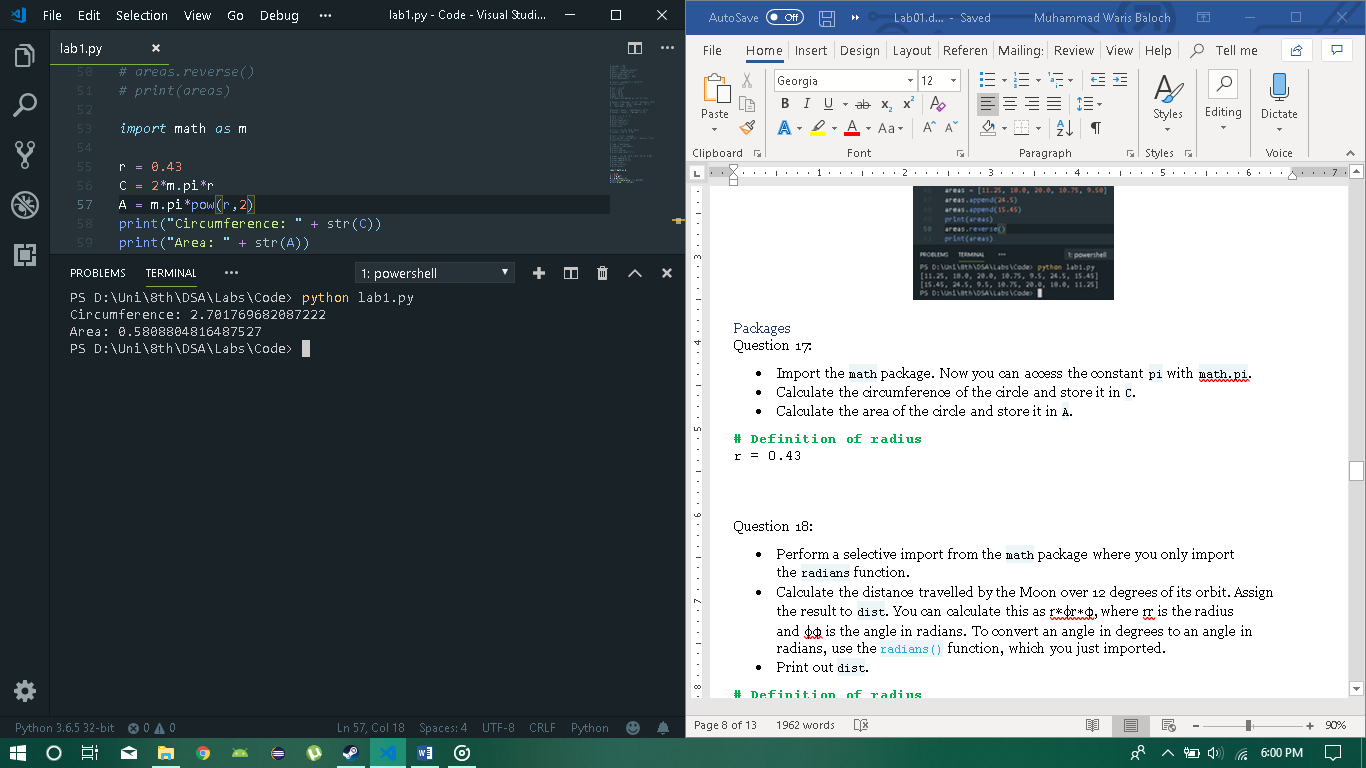
### Packages

Question 17:

* Import the math package. Now you can access the constant pi with math.pi.
* Calculate the circumference of the circle and store it in C.
* Calculate the area of the circle and store it in A.

**# Definition of radius**

r = 0.43

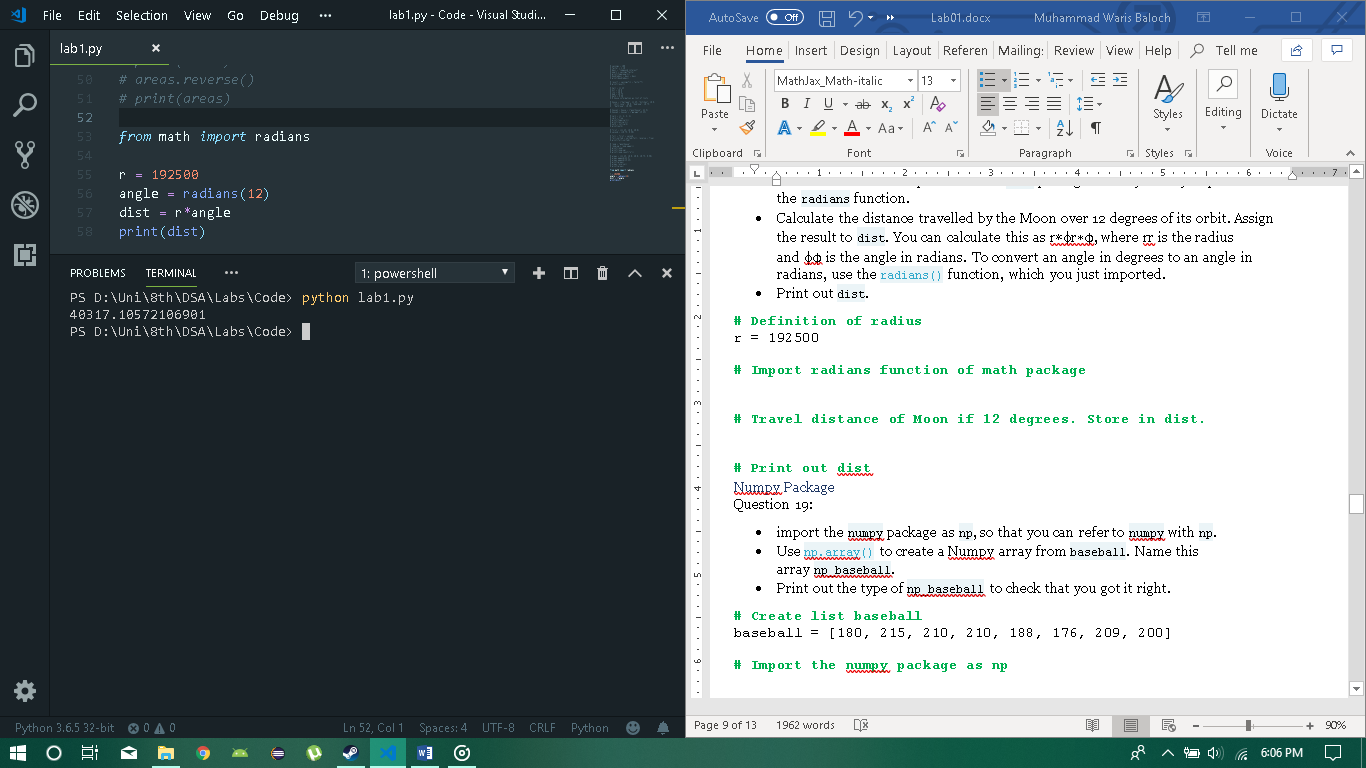


Question 18:

* Perform a selective import from the math package where you only import the radians function.
* Calculate the distance travelled by the Moon over 12 degrees of its orbit. Assign the result to dist. You can calculate this as r∗ϕr∗ϕ, where rr is the radius and ϕϕ is the angle in radians. To convert an angle in degrees to an angle in radians, use the [radians()](https://docs.python.org/3/library/math.html#math.radians) function, which you just imported.
* Print out dist.

**# Definition of radius**

r = 192500



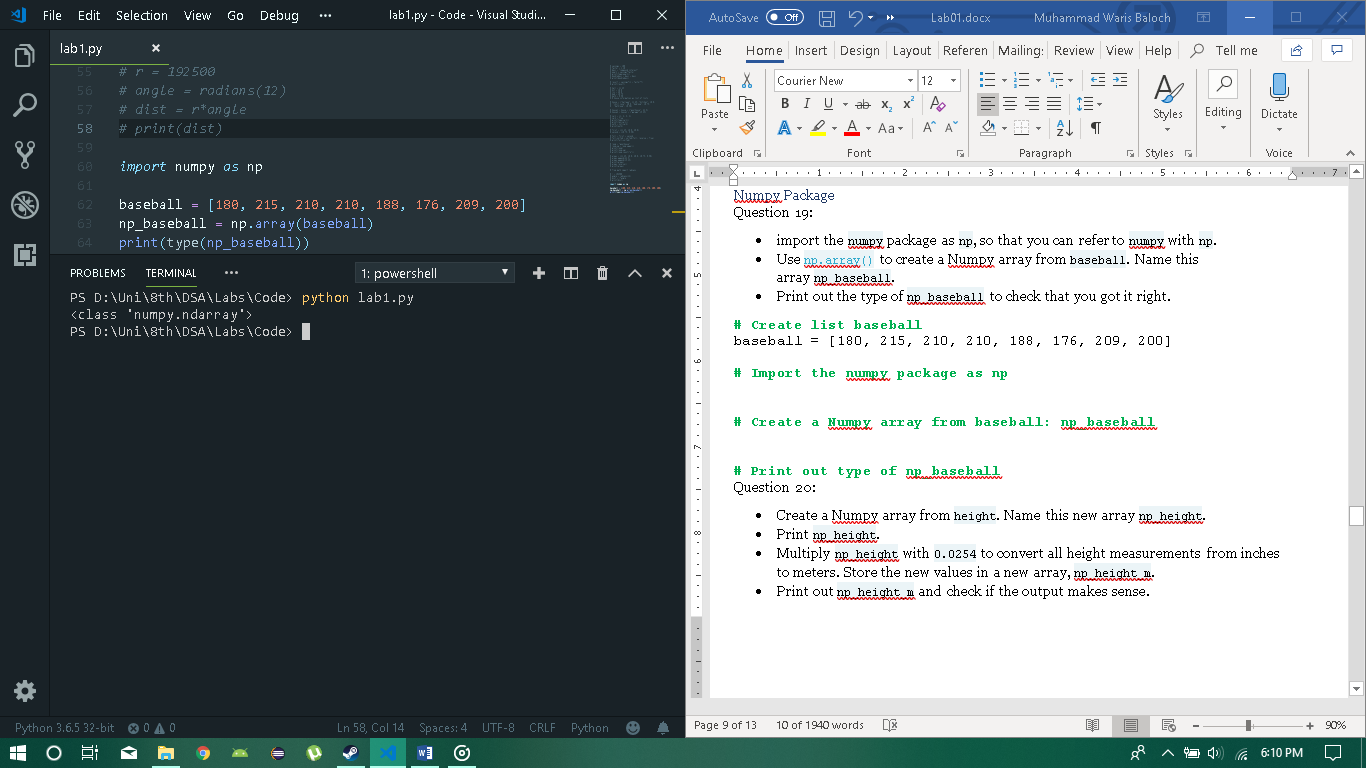
### Numpy Package

Question 19:

* import the numpy package as np, so that you can refer to numpy with np.
* Use [np.array()](http://docs.scipy.org/doc/numpy-1.10.0/glossary.html" \l "term-array" \t "_blank) to create a Numpy array from baseball. Name this array np\_baseball.
* Print out the type of np\_baseball to check that you got it right.

**# Create list baseball**

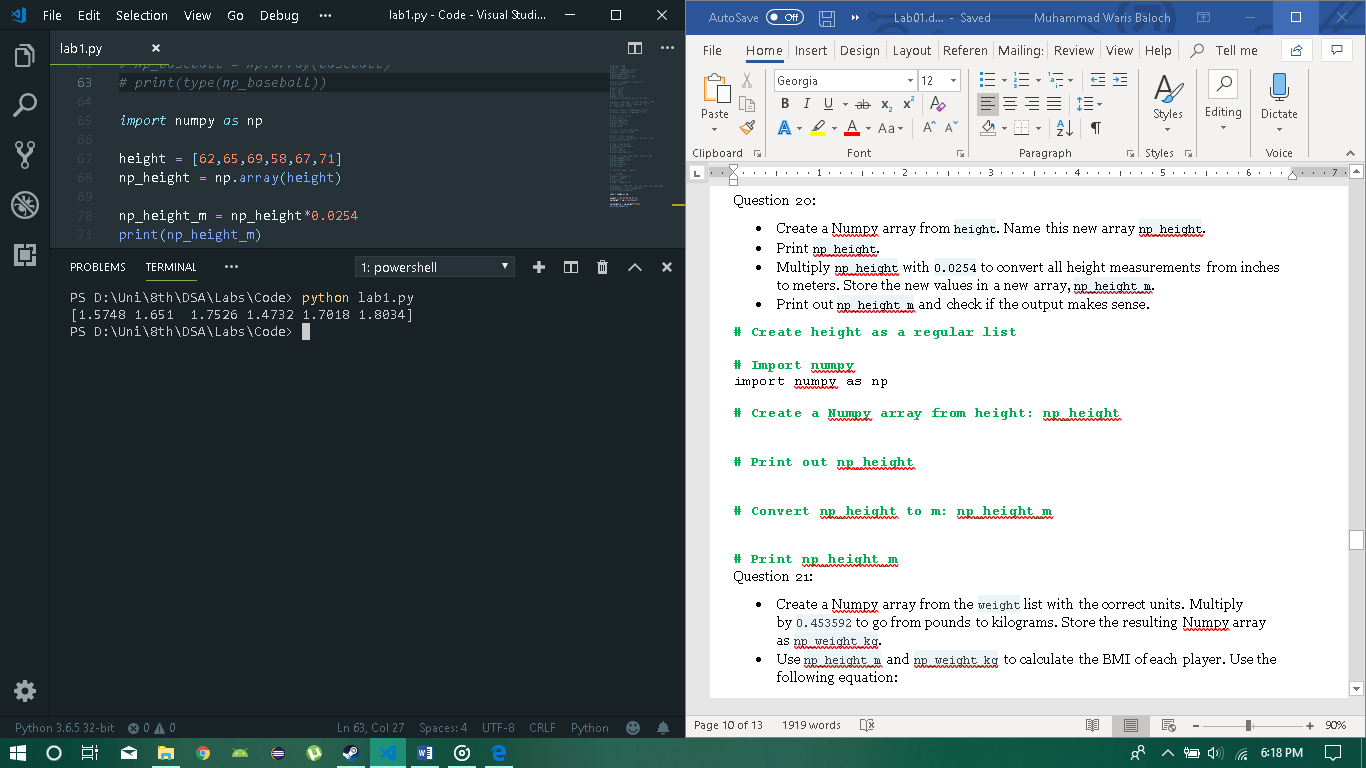
baseball = [180, 215, 210, 210, 188, 176, 209, 200]



Question 20:

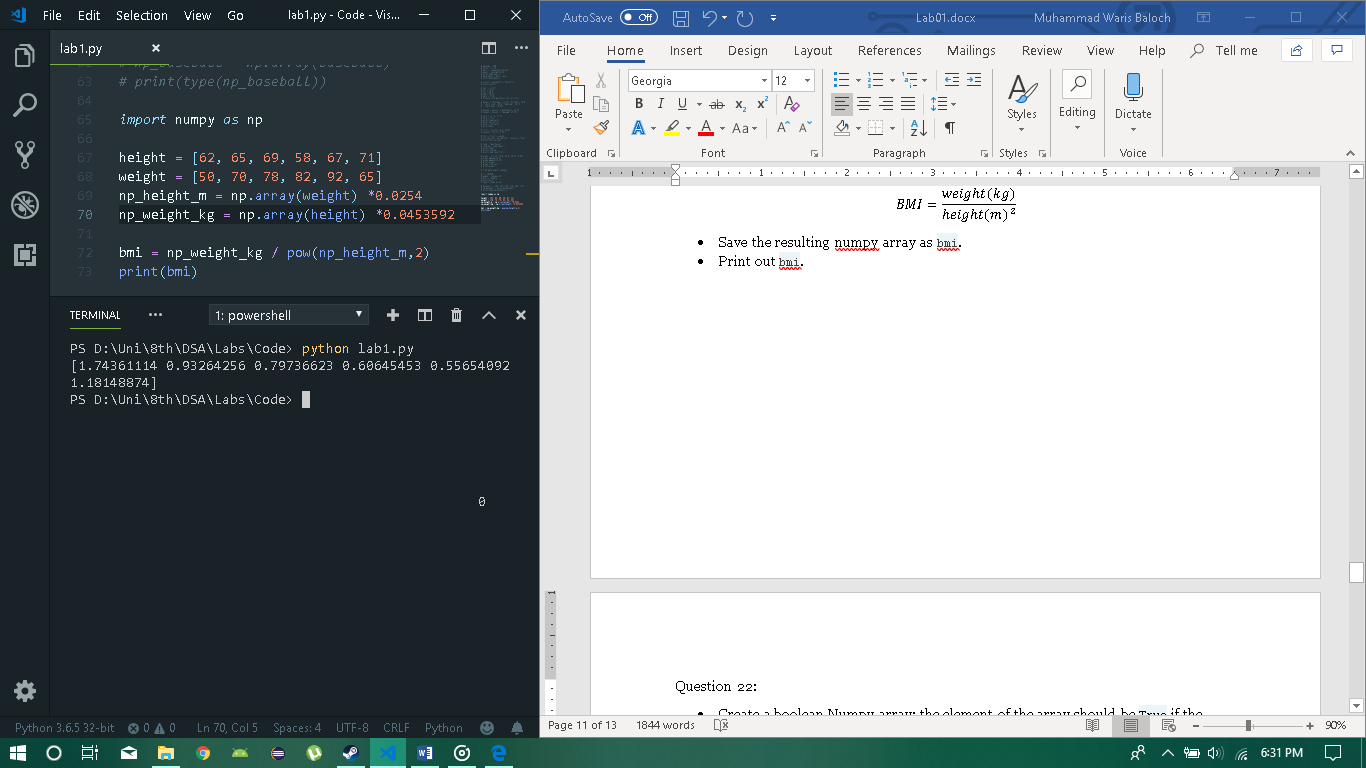
* Create a Numpy array from height. Name this new array np\_height.
* Print np\_height.
* Multiply np\_height with 0.0254 to convert all height measurements from inches to meters. Store the new values in a new array, np\_height\_m.
* Print out np\_height\_m and check if the output makes sense.

**# Create height as a regular list**



Question 21:

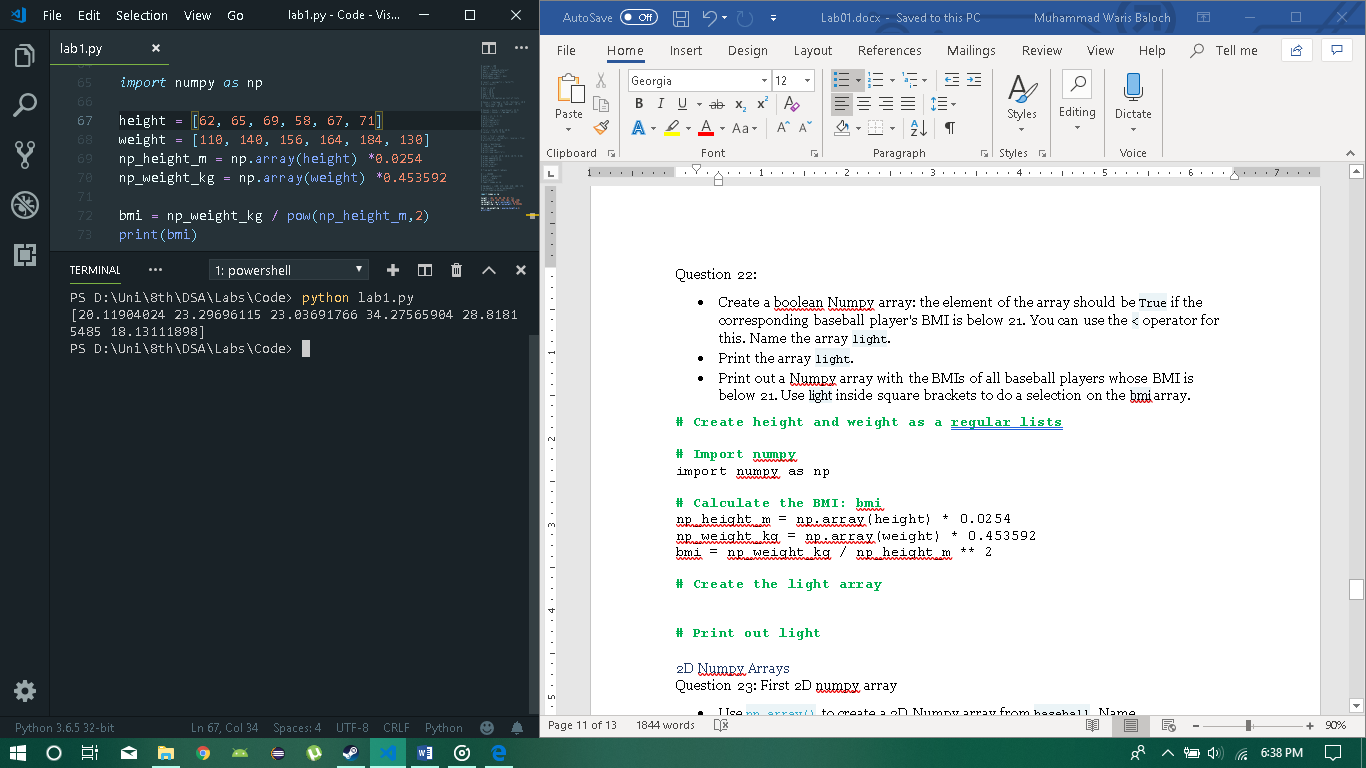
* Create a Numpy array from the weight list with the correct units. Multiply by 0.453592 to go from pounds to kilograms. Store the resulting Numpy array as np\_weight\_kg.
* Use np\_height\_m and np\_weight\_kg to calculate the BMI of each player. Use the following equation:
* Save the resulting numpy array as bmi.
* Print out bmi.



Question 22:

* Create a boolean Numpy array: the element of the array should be True if the corresponding baseball player's BMI is below 21. You can use the < operator for this. Name the array light.
* Print the array light.
* Print out a Numpy array with the BMIs of all baseball players whose BMI is below 21. Use light inside square brackets to do a selection on the bmi array.

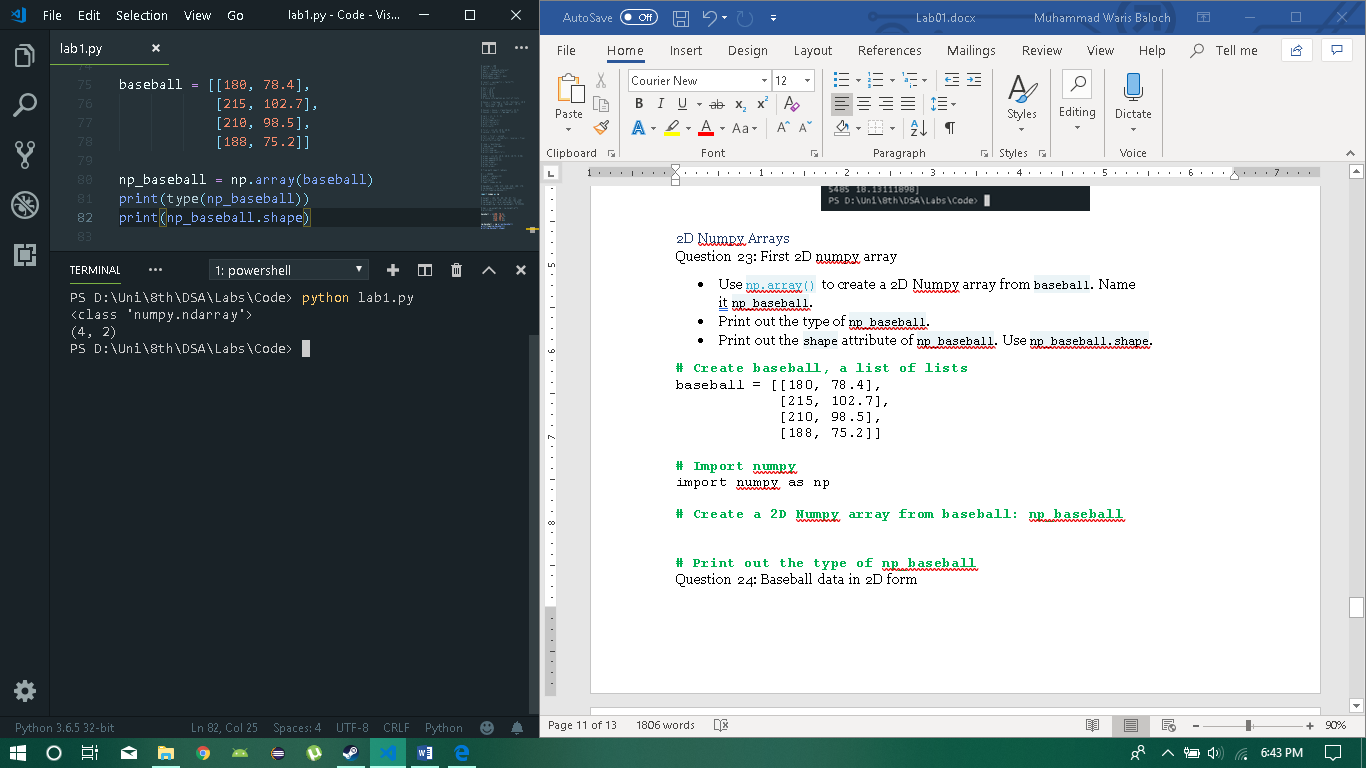
**# Create height and weight as a regular lists**



### 2D Numpy Arrays

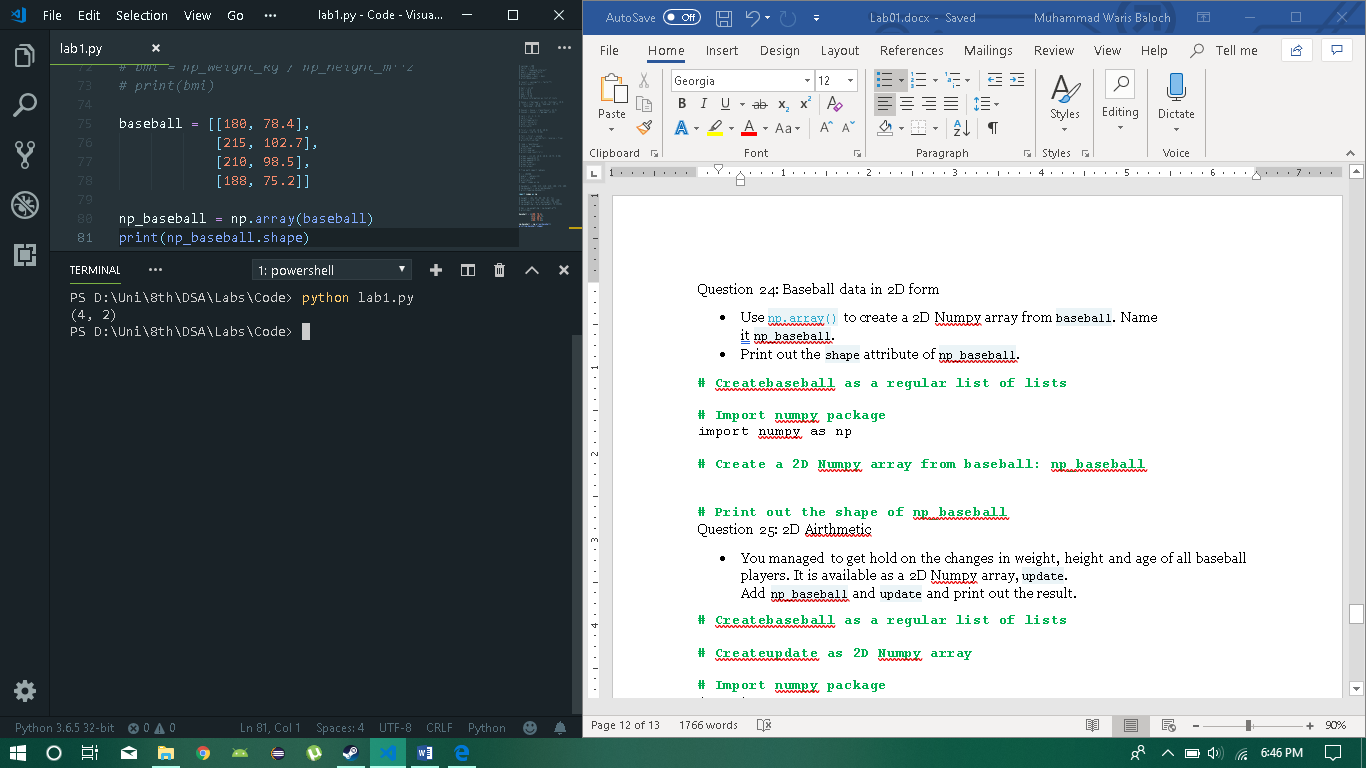
Question 23: First 2D numpy array

* Use [np.array()](http://docs.scipy.org/doc/numpy-1.10.0/glossary.html" \l "term-array" \t "_blank) to create a 2D Numpy array from baseball. Name it np\_baseball.
* Print out the type of np\_baseball.
* Print out the shape attribute of np\_baseball. Use np\_baseball.shape.



Question 24: Baseball data in 2D form

* Use [np.array()](http://docs.scipy.org/doc/numpy-1.10.0/glossary.html" \l "term-array" \t "_blank) to create a 2D Numpy array from baseball. Name it np\_baseball.
* Print out the shape attribute of np\_baseball.



Question 25: 2D Airthmetic

* You managed to get hold on the changes in weight, height and age of all baseball players. It is available as a 2D Numpy array, update.
* Add np\_baseball and update and print out the result.

**# Createbaseball as a regular list of lists**

**# Createupdate as 2D Numpy array**

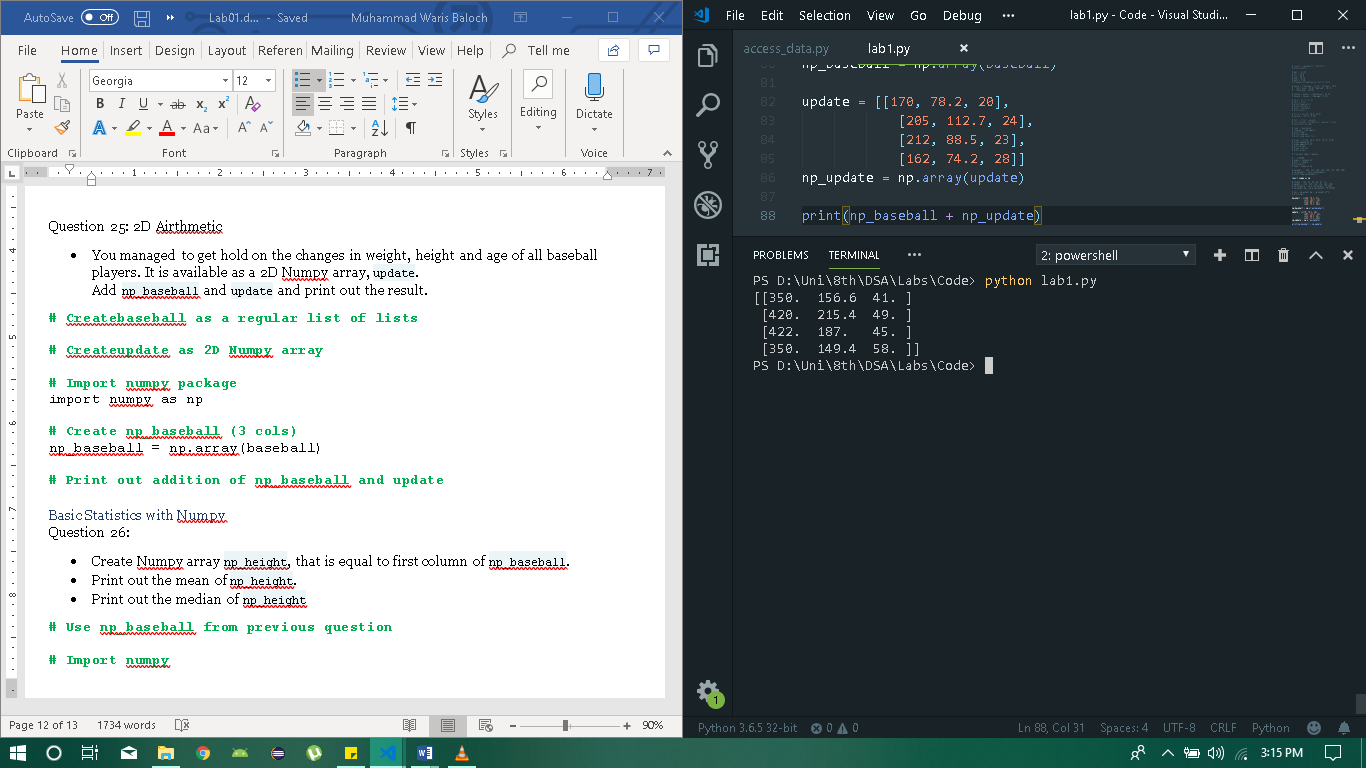
**# Import numpy package**

import numpy as np

**# Create np\_baseball (3 cols)**

np\_baseball = np.array(baseball)

**# Print out addition of np\_baseball and update**



### Basic Statistics with Numpy

Question 26:

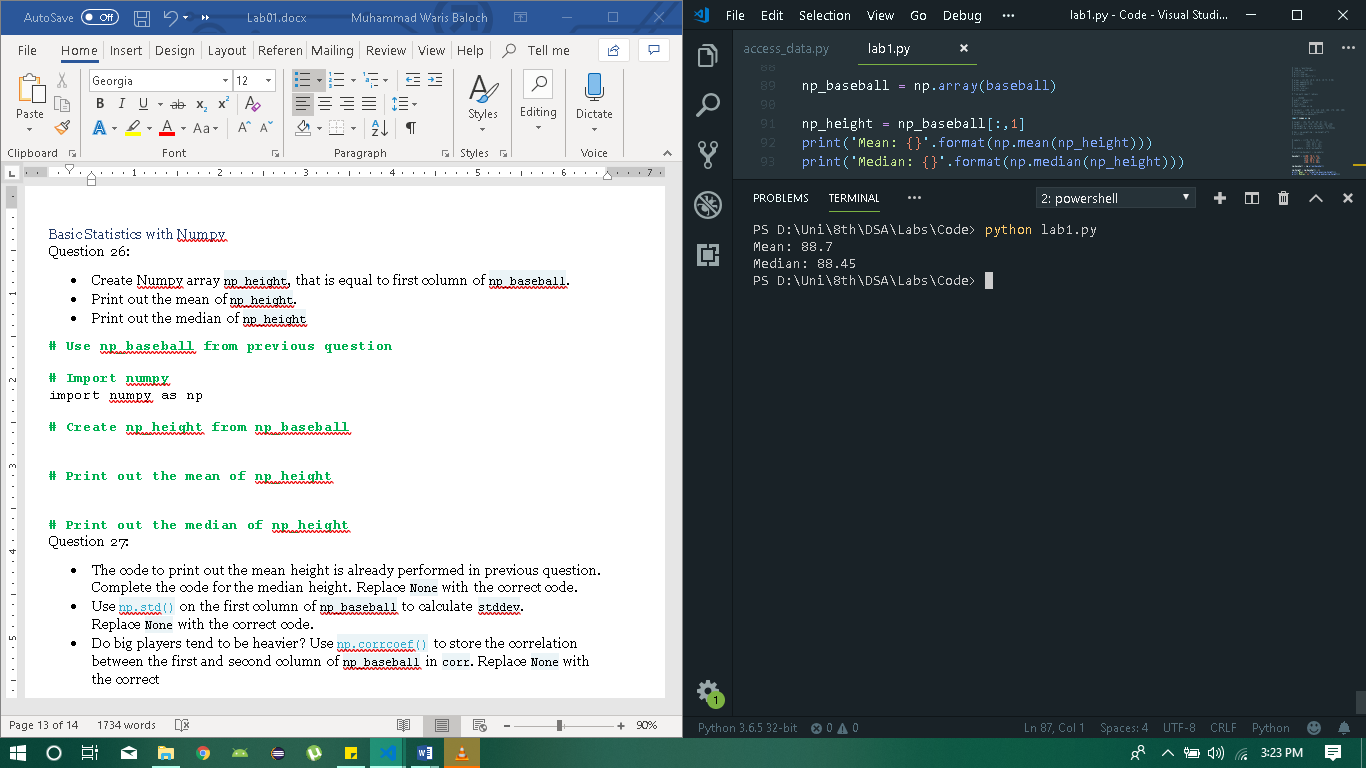
* Create Numpy array np\_height, that is equal to first column of np\_baseball.
* Print out the mean of np\_height.
* Print out the median of np\_height

**# Use np\_baseball from previous question**

**# Create np\_height from np\_baseball**

**# Print out the mean of np\_height**

**# Print out the median of np\_height**



Question 27:

* The code to print out the mean height is already performed in previous question. Complete the code for the median height. Replace None with the correct code.
* Use [np.std()](http://docs.scipy.org/doc/numpy-1.10.0/reference/generated/numpy.median.html" \t "_blank) on the first column of np\_baseball to calculate stddev. Replace None with the correct code.
* Do big players tend to be heavier? Use [np.corrcoef()](http://docs.scipy.org/doc/numpy-1.10.0/reference/generated/numpy.corrcoef.html" \t "_blank) to store the correlation between the first and second column of np\_baseball in corr. Replace None with the correct

**# Use np\_baseball from previous question 25**

**# Import numpy**

import numpy as np

**# Print mean height (first column)**

avg = np.mean(np\_baseball[:,0])

print("Average: " + str(avg))

**# Print median height. Replace 'None'**

med = None

print("Median: " + str(med))

**# Print out the standard deviation on height. Replace 'None'**

stddev = None

print("Standard Deviation: " + str(stddev))

**# Print out correlation between first and second column. Replace 'None'**

corr = None

print("Correlation: " + str(corr))

