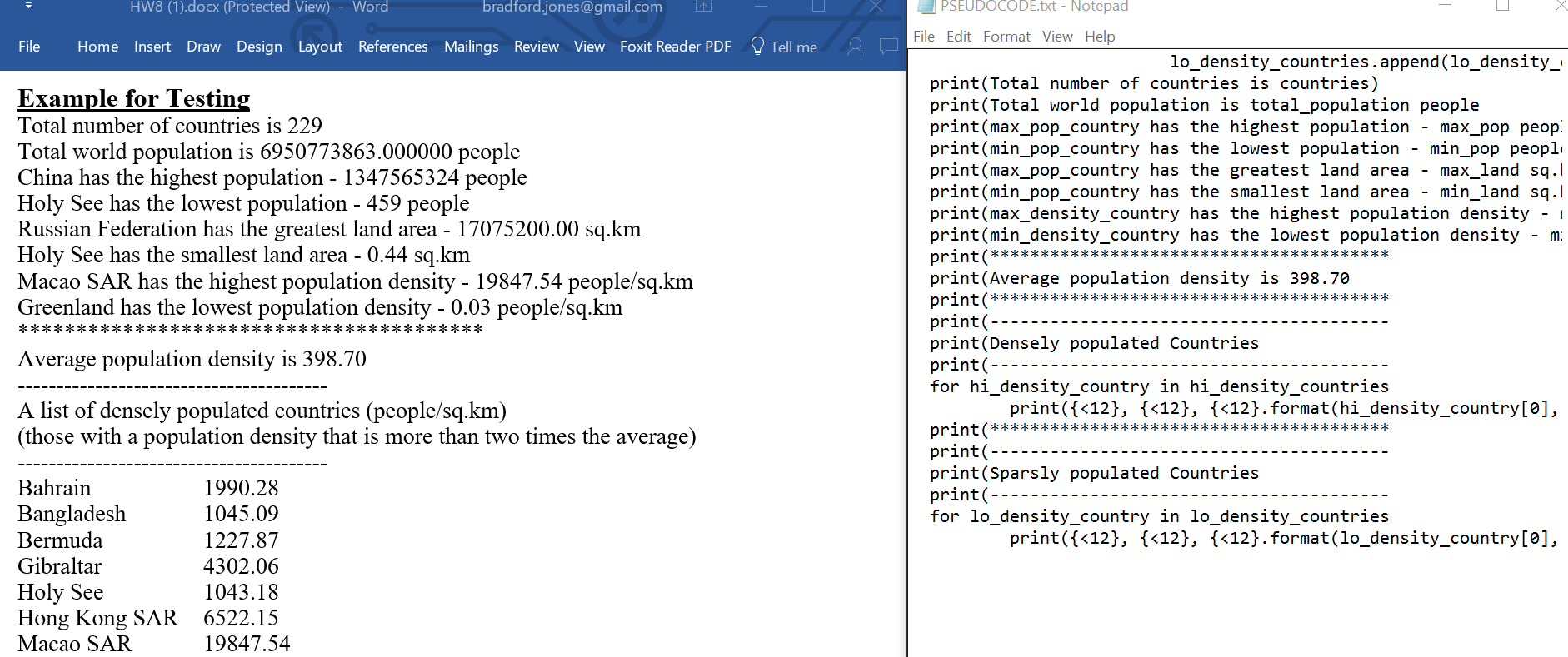
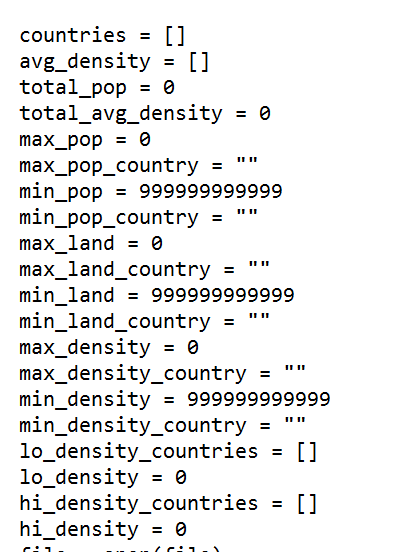
# World Population HW8

For the majority of the HW’s I start by viewing the assignment side by side with a blank notepad.

I like to start by going to the end of the assignment and copying the output section, then going over to notepad and pasting. Then I edit the output to remove numbers and start to insert variables.

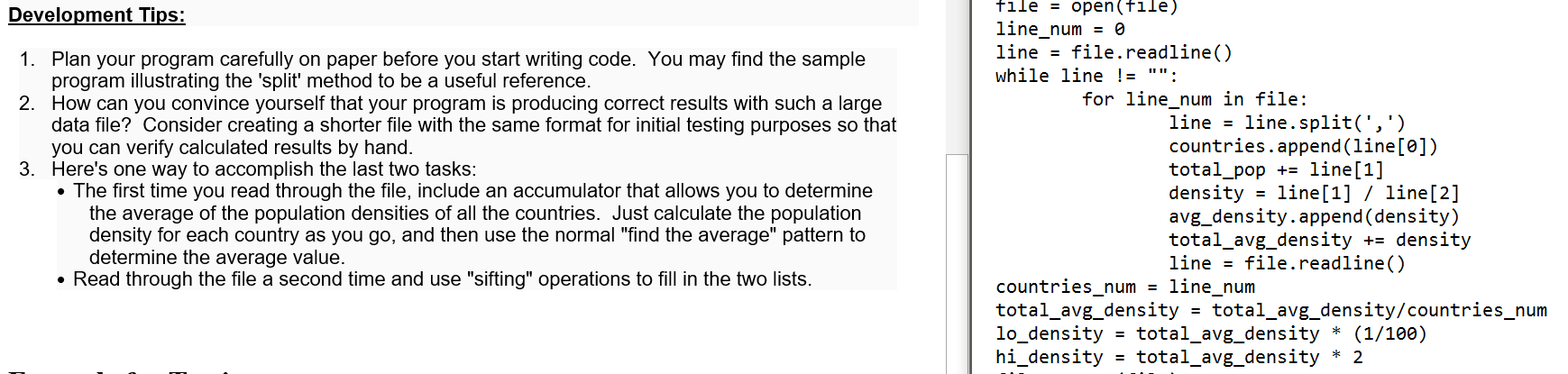


As I format the output section I begin to name variables and list them at the top. When I am through getting variables from the output section, I look through the assignment for other variables/accumulators/lists we will need for the rest of the program.

Min values must start very high (99999999999) in this case to count down from.

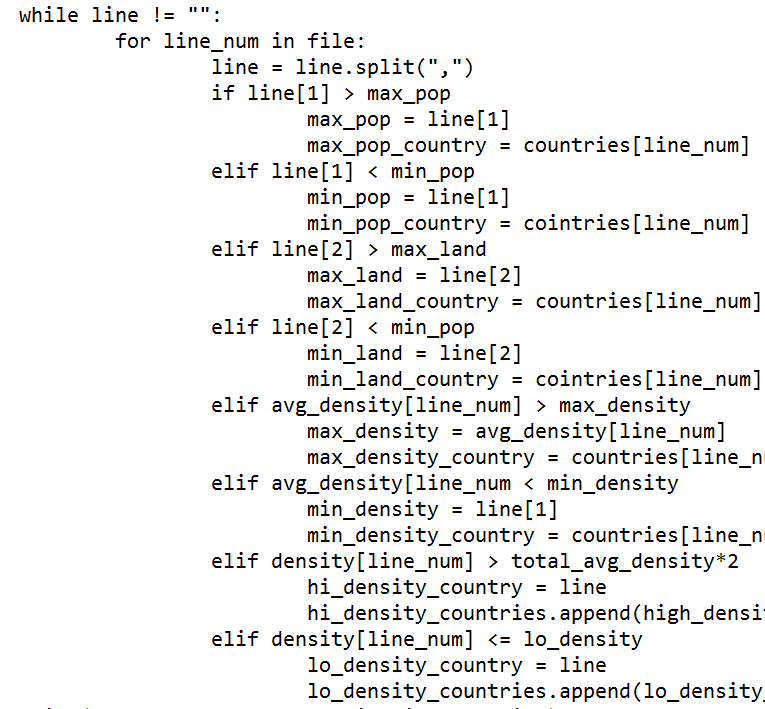
The assignment indicates that we should use an initial loop to count populations and density’s before looping through a second time to find the min/max’s. This is required because we will need to loop through the list and make comparisons against the average population density. You cant compare against something you don’t know so 2 loops!

Here is the construction of the first loop and calculating the density



Total\_avg\_density first adds all the densities together, then divides by the number of total countries. hi/lo\_density will comparisions for the hi/lo\_density lists.

Countries[] is a list of all country names. This is important because Ill recall them by index number later.

Here is the second loop:

Find max pop and country

Find min pop and country

Find max land and country

Find min land and country

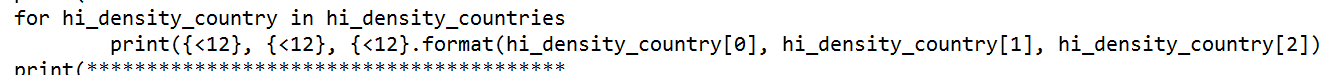
Find max density and country

Find min density and country

Check if low density

Check if hi density

The following is a part I’m improvising because I don’t know if this is how python works or not. I think its like a 2d array. Each list item in countries has another list item with 3 indices(name,pop,mass). I realized that I hadn’t set a mechanism for printing both the country name AND the density, so Im trying this.

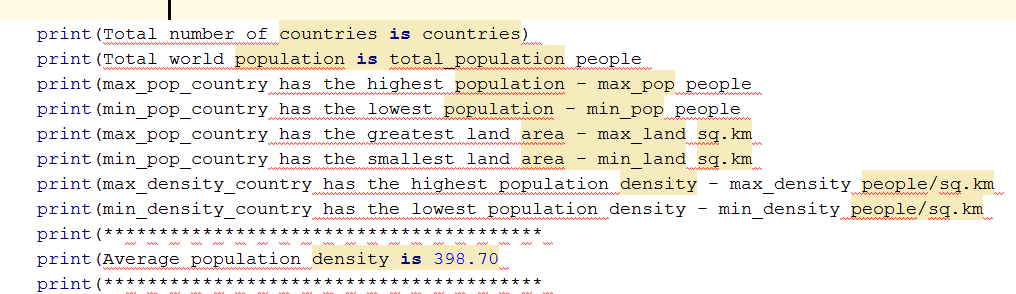


Still needs input validation/error checking/file closing but pseudocode is good to go.



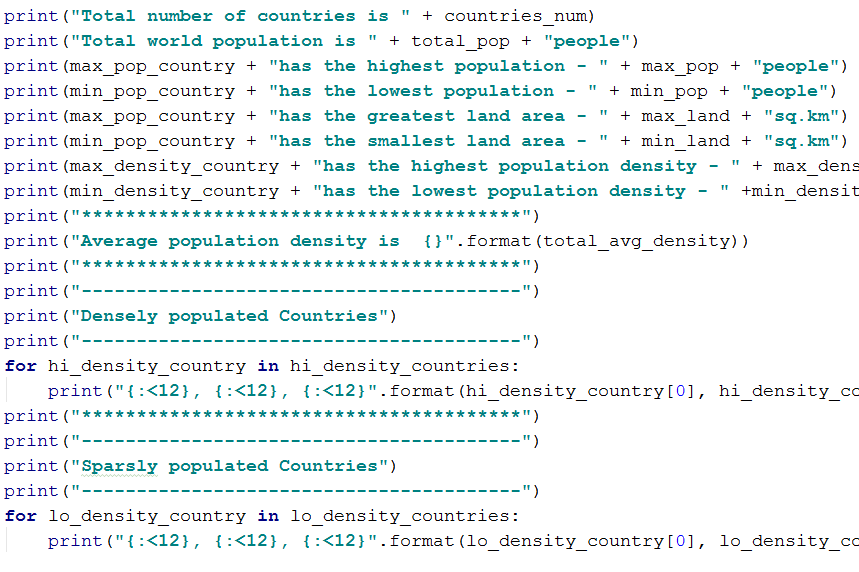
Copy and paste from PSUEDOCODE into PyCharm, begin formatting and commenting into python readable code:

The output section generally requires a lot of work to get functional.



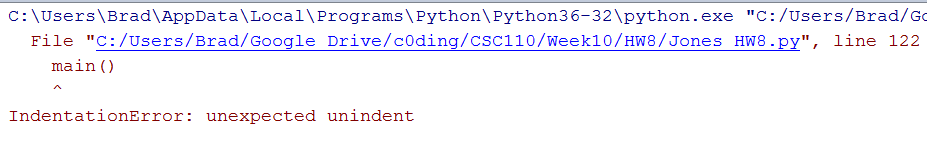
After Formatting:

Note the for loop for the 2d array are showing no errors. That’s a good sign

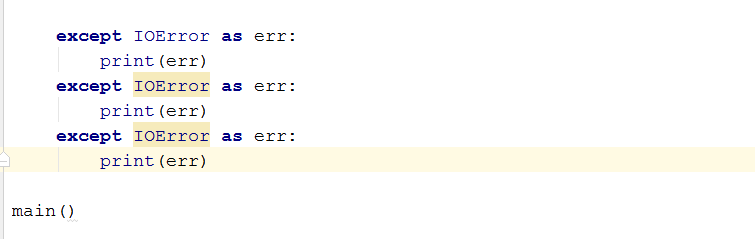
Time to run the program for the first time, using the test.txt file and start debugging

DEBUGGING:

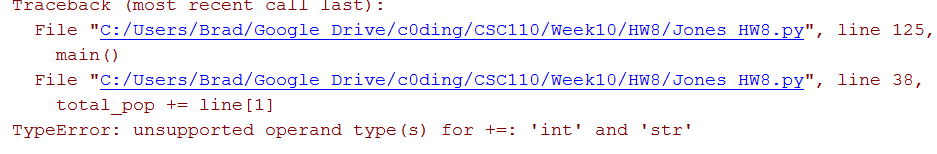
PROBLEM:



Solution

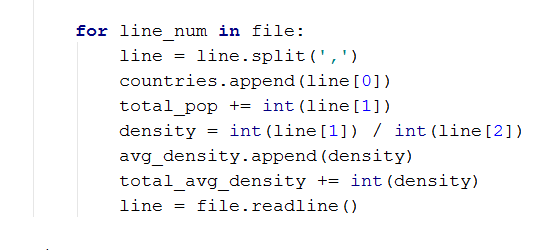


PROBLEM:

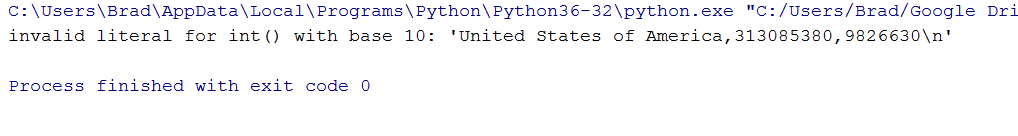


Solution:

Int() conversions



PROBLEM:



SOLUTION:

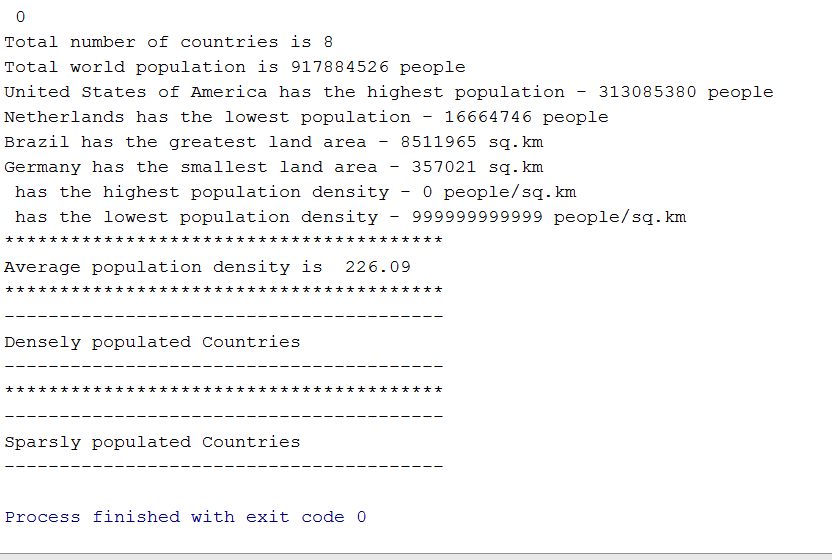
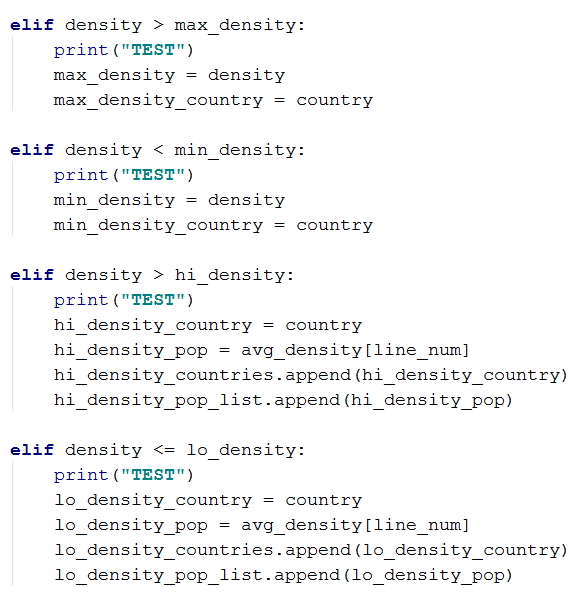
This is the same problem I was having with HW7. Strange that the error occurs at this line of the file, which is nether the beginning of the end, also that the” \n” is still there

Not a real solution, but works for now:



PROBLEM:

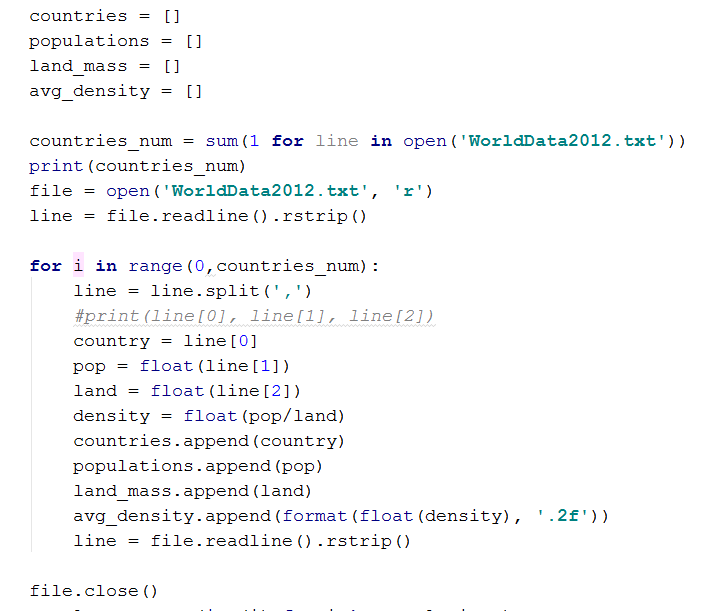
Ok for some reason my density counters and accumulators are not functioning. Added print statements to make sure they are being passed correct values:

Here you can see that the IF statements are never being activated for the density segments

I realized the elif statements were not working correctly because I hadn’t nested them properly.

# OK, starting over… I have to rewrite the majority of the program to get the if statements nested properly.

This time I’m going to read the file once and append the info into separate lists. Then I will use the index of the loops to store info on min/max’s

l

# Here is the only time I will access the file.

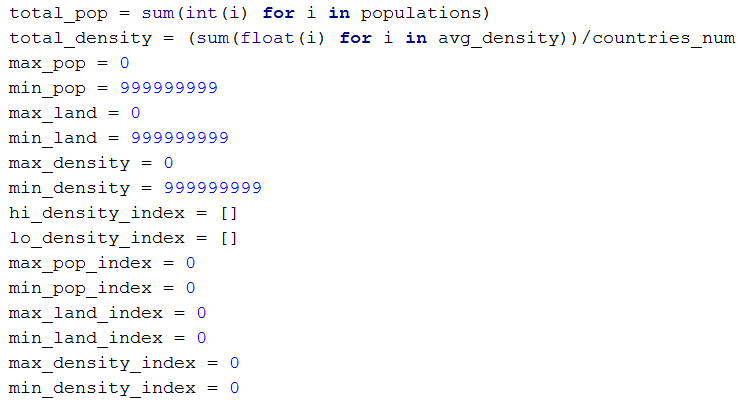
First I count the number of lines, to set the loop counts

Then open the file and strip the whitespace

For each loop I assign the values to the lists I initialized above.

Print statement for testing purposes

countries[i], populations[i], land\_mass[i], avg\_density[i] will show all info about country on line[i]

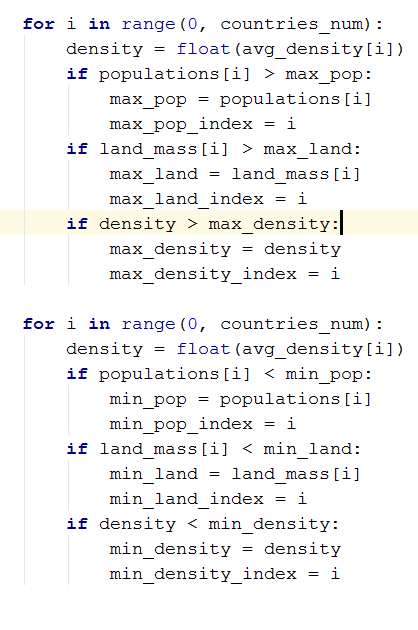
Then I set accumulators to find mins/maxs

The loops in total\_pop and total\_density convert the list to int() and float() so they can be used for math

The hi and lo density index lists hold the index value of countries that fit the parameters,

EXAMPLE: So if hi\_density\_index[0] = 3

Countries[3] is a hi density country, with a density of avg\_density[3]

Loops to track the max’x and min’s values.

You could eliminate these loops by nesting then further but you would need to either repeat a lot of code, or write functions to handle the repeated portions

If I was worried about big(O) I would have done something like:

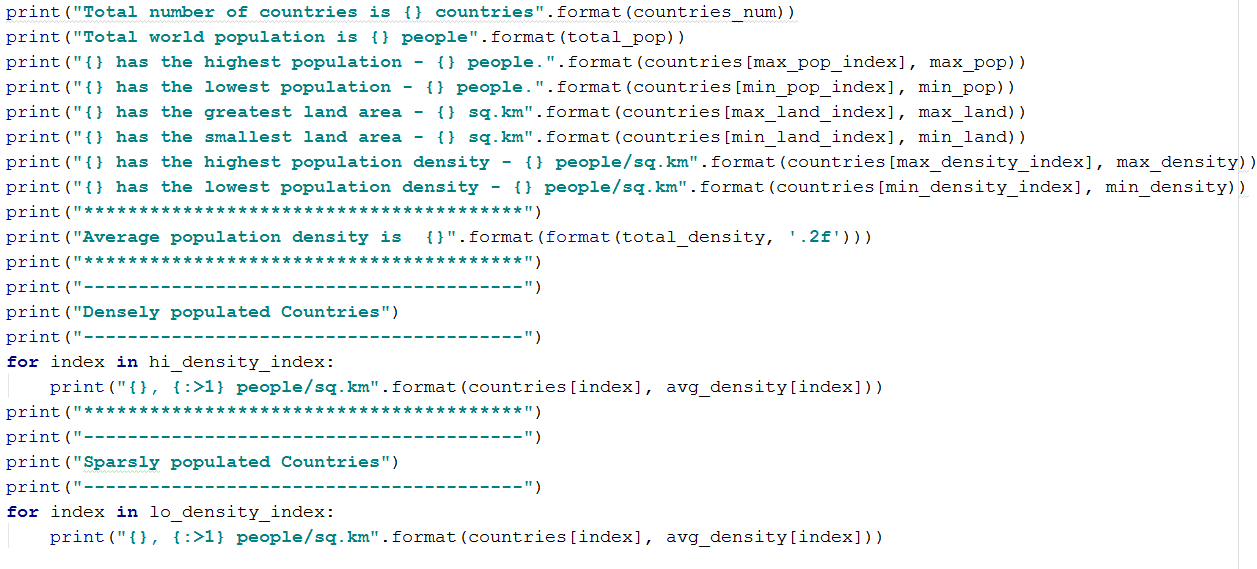
Def \_init\_(country,pop,land,density):

Country.pop = pop

Country.land = land

Country.density = density

For each line item. Then I could parse through the objects without have to run multiple loops, but we haven’t talked about any of that yet so I avoided it.



The print section is very similar to the last few HW assignments. The interesting part is the Hi and Lo density sections, where the loops reads the numbers in the lists one at a time, then prints the info from the INDEX VALUE from the list