Pseudocode for the automation of Taylor's Problem Specification

- Store all the Input Files inside the Input_Files folder which has to be created in the same folder as execution.py
- The name of the file has to be of the form YYYY_anyname.csv
- Execute the Program execution.py using python3
- The program reads the files as inputs one by one. The name of the file is retrieved.
- An instance of the workbook to which the output has to be stored is created and this instance along with the input files is parsed into the function().
- The data from the input file is read into a dictionary variable named as dictionary with respect to the ba values
  Eg : dictionary = {1:{nuclear:40.66}}
- Take in ba value from user
- Calculate gen_total[i] using dictionary[ba][i] where i is the for loop running through comparing data which is all the energy types
- Calculate gen_total_main using a value
- and dictionary. Here we sum up all the float values with respect to ba value
- Calculate percent_carbon,percent_change, gen_change_total, non_carbon_total using the formulas
- If energy type is carbon, then (dictionary)gen_change[i] = gen_total[i]*year_dict[int(year)](This is hardcoded)
- Some of the gen_change for certain energy types are hardcoded
- In order to calculate the gen_change for other energy types we need viability matrix
- Some of the viability dictionary for certain energy types are hardcoded
- We convert the given matrix in excel sheet and store it in row wise for each row ba values i:e 1:134.
- We calculate the viability matrix using the ba value given by the user and the hardcoded set_data values.
- We calculate the sum_off value by adding the values in viabilities matrix

- Calculate gen_change[i] using the formula gen_change_total*0.6/sum_off if viability[i]==1 else 0
- Calculate another dictionary new_total with the following conditions for each key in comparing_data list
  If i is carbon_type
  New_gen[i] = Gen_total[i]-gen_change[i]
  Else
  New_gen[i] = gen_total[i]+gen_change[i]
- Now having all the values, using the workbook instance write on each sheet corresponding to the ba values. The ba values range from 1 to 134 and we run a for loop initially before parsing individual files.
- Return the new_gen dictionary from the function() and using this dictionary, we constitute the output page as Taylor requested.
- The output files are stored in the same name format using the additional extension _output.xlsx and these output files are stored in the output_files/ folder.