```
# - ProblemSet4.py *- coding: utf-8 -*-
"""
Problem 4_1:
Write a function that will sort an alphabetic list (or list of words) into
alphabetical order. Make it sort independently of whether the letters are
capital or lowercase. First print out the wordlist, then sort and print out
the sorted list.
Here is my run on the list firstline below (note that the wrapping was added
when I pasted it into the file -- this is really two lines in the output).

problem4_1(firstline)
['Happy', 'families', 'are', 'all', 'alike;', 'every', 'unhappy', 'family',
 'is', 'unhappy', 'in', 'its', 'own', 'way.', 'Leo Tolstoy', 'Anna Karenina']
['alike;', 'all', 'Anna Karenina', 'are', 'every', 'families', 'family',
'Happy', 'in', 'is', 'its', 'Leo Tolstoy', 'own', 'unhappy', 'unhappy', 'way.']

"""
#%%
firstline = ["Happy", "families", "are", "all", "alike;", "every", \
              "unhappy", "family", "is", "unhappy", "in", "its", "own", \
              "way.", "Leo Tolstoy", "Anna Karenina"]
#%%
def problem4_1(wordlist):
    """ Takes a word list prints it, sorts it, and prints the sorted list """
    pass # replace this pass (a do-nothing) statement with your code

#%%
"""
Problem 4_2:
Write a function that will compute and print the mean and standard deviation
of a list of real numbers (like the following).  Of course, the length of the
list could be different. Don't forget to import any libraries that you might
need.
Here is my run on the list of 25 floats create below:

problem4_2(numList)
51.528
30.81215290541488

"""
#%%
import random
numList = []
random.seed(150)
for i in range(0,25):
    numList.append(round(100*random.random(),1))
#%%
def problem4_2(ran_list):
    """ Compute the mean and standard deviation of a list of floats """
    pass # replace this pass (a do-nothing) statement with your code
#%%
"""
Problem 4_3:
Write a function problem4_3(product, cost) so that you can enter the product
and its cost and it will print out nicely. Specifically, allow 25 characters
for the product name and left-justify it in that space; allow 6 characters for
the cost and right justify it in that space with 2 decimal places. Precede the
cost with a dollar-sign.  There should be no other spaces in the output.

Here is how one of my runs looks:
problem4_3("toothbrush",2.6)
toothbrush               $  2.60

"""
```

```
#%%
def problem4_3(product, cost):
    """ Prints the product name in a space of 25 characters, left-justified
        and the price in a space of 6 characters, right-justified"""
    pass # replace this pass (a do-nothing) statement with your code


#%%
"""
Problem 4_4:
This problem is to build on phones.py.  You add a new menu item
  r) Reorder
This will reorder the names/numbers in the phone list alphabetically by name.
This may sound difficult at first thought, but it really is straight forward.
You need to add two lines to the main_loop and one line to menu_choice to print
out the new menu option (and add 'r' to the list of acceptable choices).  In
addition you need to add a new function to do the reordering: I called mine
reorder_phones(). Here is a start for this very short function:

def reorder_phones():
    global phones       # this insures that we use the one at the top
    pass # replace this pass (a do-nothing) statement with your code


Note: The auto-grader will run your program, choose menu items s, r, s, and q
in that order.  It will provide an unsorted CSV file and see if your program
reorders it appropriately. The grader will provide a version of myphones.csv
that has a different set of names in it from the ones we used in the lesson.
This difference in data will, of course, not matter with a well coded program.
Below the result of this added function is shown using the names used in class.
Note that name is a single field.  Reorder by that field, don't try to separate
first and last name and reorder by one or the other --- just treat name as a
single field that you re-order by.  Also, in this case upper/lower case won't
matter.

TIP: phones[] is a list of lists (each sublist is a [name, phone].  It looks
complicated to sort.  Just pretend that each sublist is a single name item and
code it accordingly.  It will work.  This is a beginner course and this sort
function requires only one line and no fancy outside material to make it work.)
The main thrust of this problem is to add in the various pieces to make a new
menu entry.

Before:
Choice: s
     Name                     Phone Number
  1  Jerry Seinfeld          (212) 842-2527
  2  George Costanza         (212) 452-8145
  3  Elaine Benes            (212) 452-8723
After:
Choice: s
     Name                     Phone Number
  1  Elaine Benes            (212) 452-8723
  2  George Costanza         (212) 452-8145
  3  Jerry Seinfeld          (212) 842-2527

"""
```