New York University
School of Continuing and Professional Studies
Division of Programs in Information Technology

Introduction to Python
Homework, Session 5

5.1 Reading through student_db.txt, create a dictionary where the key is the student id (first field) and the value is a list of line values from split().

When loaded, the dict will look like this:

```
students = { 'jk43':  ['23 Marfield Lane', 'Plainview' ,'NY', '10023'],
             'axe99': ['315 W. 115th Street, Apt. 11B', 'New York', 'NY', '10027'],
             ...[etc]...                                                         ]
```

After the dict has been loaded, take user input for an id like jk43. Print that user's addres
in a readable format.

This assignment encourages you to think about how to start thinking about multidimensional structures. (The dictionary key is a string, and the dictionary value is a list, a slice of the split of the line.) You will need to split the string in order to retrieve the i
as well as to slice the list that will become the value.

As far as building the dict goes, the split() of the line returns a list. Simply slice the list to exclude the id and use this slice as the value.

Taking user input: using an endless (while True) loop, allow for 3 conditions: the user types a valid id (program prints the address and continues to ask for another id); the user types an invalid id (program prints error message and continues to ask for another id); the
user types q (program breaks out).

If a valid id is entered, simply use the list associated with that id in the dict to build a readable address block as shown.

Sample program run(s):

```
7 records loaded.
Please enter an id ('q' for quit):  xx95
sorry, that id does not exist

Please enter an id ('q' for quit):  jk43

address for jk43:
23 Marfield Lane
Plainview, NY  10023

Please enter an id ('q' for quit):  axe99

address for axe99:
315 W. 115th Street, Apt. 11B
New York, NY  10027

Please enter an id ('q' for quit):  q
```

5.2 Reading through F-F_Research_Data_Factors_daily.txt, create a dictionary that sums all the Mkt-RF values for each year in the file. The script should query the user for a number of results and the value "highest" or "lowest", indicating whether the results should be the highest or lowest years. (For example, if the user enters 5 and lowest the script will return the 5 years with the lowest sums.)

You must use a dictionary to sum up the values, and you should loop through the file data only once. Please don't add selected values to a list and then sum up the list in order to add to a dictionary -- the purpose of this exercise is to use a dictionary itself to sum up th values. Please also don't loop through the data more than once -- some students resort to this to solve the KeyError problem (this is discussed in the discussion document), but you should always think of looping through the data once. A "summing dictionary" is the point of this exercise.

As you loop line by line, the operative code line is this:

```
days[year] = days[year] + mkt_rf   # here, mkt_rf is the
                                   # current Mkt-RF value
```

However, as you loop, you will need to check the dict ahead of time to see if the year key is already there. If not, set the key and value in the dict for that line.

Please avoid looping through the file more than once. Please store all summable year values in a dictionary only. A dictionary is all you need to compile a sum for each year. This exercise helps you see how a dict works and the kind of work it can do.

Validate input: for number of results, reject a non-integer or an integer greater than the available number of years; for 'highest' or 'lowest', reject any value that is not one of thos (hint: you can use a test like if user_input not in ['highest', 'lowest']).

Sample program run(s):

```
$ python myprog.py
please enter number of results desired:  ten
select "highest" or "lowest" results:  lowest
num results must be all digits

please enter number of results desired:  195
select "highest" or "lowest" results:  lowest
num results "195" greater than max "87"

please enter number of results desired:  10
select "highest" or "lowest" results:  middle
top or bottom results must be "highest" or "lowest"

please enter number of results desired:  5
select "top" or "bottom" results:  lowest
1931:  -52.44
2008:  -39.32
1937:  -38.96
1974:  -38.09
1930:  -33.61
```