Lab 1- Instructions:

1. **Read the HW guidelines** posted in d2l.depaul.edu > admin.
2. Your solution file MUST be named **lab1.py**. In a comment, include the name(s) of any collaborators. Include implementations of the functions in the problems below.
3. To receive full credit, the **names** of files, functions and the **output** must be **exactly** as indicated here.
4. **Test your code** by downloading the file **lab1TEST.py** in the same working folder and add the following code to the bottom of your lab1.py module. Run and fix errors before you submit.

```
if __name__=='__main__':
    import doctest
    print( doctest.testfile( 'lab1TEST.py' ))
```

Problems:

1. Write a function `printTwoLargest()` that inputs an arbitrary number of positive numbers from the user. The input of numbers stops when the first negative or zero value is entered by the user. The function then prints the two largest values entered by the user. If no positive numbers are entered a message to that effect is printed instead of printing any numbers. If only one number is inputted, only the largest is printed out (see 2nd example below). Sample output:

```
>>> printTwoLargest()
Please enter a number: 12
Please enter a number: 99.9
Please enter a number: 4.5
Please enter a number: 77
Please enter a number: 0
The largest is 99.9
The second largest is 77
>>> printTwoLargest()
Please enter a number: 23.2
Please enter a number: -99
The largest is 23.2
>>> printTwoLargest()
Please enter a number: -9
No positive numbers were entered
>>>
```

2. Implement a function `printWordsLines()` that takes as a parameter the name of an input file. It opens the input file and then prints the number of lines and the number of words found in the file. Words are strings separated by spaces, and lines are strings terminated by the newline character. Don't forget to close the input file after you are done reading it. Sample runs:

```
>>> printWordsLines('test1.txt')
The file test1.txt contains 17 words and 3 lines
>>> printWordsLines('test2.txt')
The file test2.txt contains 38 words and 5 lines
```

3. Write a function `reverseDict()` that takes accepts one argument, a dictionary. The function constructs and returns the reversed dictionary. You may assume that all keys and values are unique. Sample runs:

```
>>> reverseDict( {'a':1,'b':2,'c':3})
{1: 'a', 2: 'b', 3: 'c'}
>>> reverseDict( {1:'apple', 2:'pear', 3:'orange'} )
{'orange': 3, 'apple': 1, 'pear': 2}
```