COSC 101, Exam #3
November 2018

Name: ______________________________

Please write your name above. Do not start the exam until instructed to do so.

You have 50 minutes to complete this exam.

There are 6 questions and a total of 41 points available for this exam. Don't spend too much time on any one question.

Since indentation is important in Python, please be sure that your use of indentation is obvious for any code you write.

If you want partial credit, show as much of your work and thought process as possible.

If you run out of space for answering a question, you can continue your answer on one of the blank pages at the end of the exam. If you do so, be sure to indicate this in two places: (1) below the question, indicate which blank page contains your answer, and (2) on the blank page, indicate which question you are answering.

Question	Points	Score
1	7	
2	5	
3	4	
4	3	
5	10	
6	12	
Total:	41	

1. (7 points) Assume that the following statements have already been executed:

```
s = 'A snowy thanksgiving?'
q = [5, 2, 7, 3]
d = {'k':7, 'n':4, 'w':2}
```

For each of the following expressions, evaluate the expression and write the resulting value, or identify the error in the code that would prevent it from running.

(a) `d[s[-3]]`

(b) `s[1:7:2]`

(c) `len(q) in d`

(d) `len(d) in q`

(e) `s.split()[2][-3:-9:-7] in s.split()[1]`

(f) `max(q) - min(d.values())`

(g) `d[s[sum(q[::2])]]`

2. (5 points) What are the contents of the file 'out.txt' after the following code is run?

   The file "q2.txt" contains the following:

```
h 18.2 44.3 23.1
y 13.1 22.3 19.6
h 16.8 42.5 21.0
h 16.9 42.4 22.3
y 12.9 23.5 19.9
```

```
def mystery(filename):
    d = {}
    with open(filename) as f:
        for i in f:
            if i[0] in d:
                d[i[0]].append(i[2:].split())
            else:
                d[i[0]] = [i[2:].split()]
    return d

def mystery2(f,d,a):
    with open(f, 'w') as out:
        for i in d:
            if i == a:
                for j in d[i]:
                    for k in range(len(j)):
                        out.write(j[k] + ' ')
                    out.write('\n')

mystery2("out.txt",mystery("q2.txt"),'h')
```

3. (4 points) Rewrite the function below to use the file function readline() instead of read(). Your solution must produce the same exact output as the original function. You may not use a for loop, readlines() or read() in your solution. (In your solution, you need only to rewrite the part below the comment.)

```
def readfile(filename):
    f = open(filename, 'r')

    # rewrite function from here down...
    p= []
    x = f.read()
    x = x.split('\n')
    for i in x:
        p.append(i.split())
    f.close()
    return(p)
```

4. Consider the following python function:

```
def my_func(d1,d2):
    result = -1
    for i in d1:
        if d1[i] in d2:
            return 1
        else:
            for j in d2:
                if i == d2[j]:
                    result = 0
    return result
```

   (a) (1 point) Consider the situation if `my_func` is called with `d1` equal to {`'b':2, 'c':3, 'd':4`}. Identify one value that, if passed as the last parameter, `d2`, would cause the value `0` to be returned from the function.

   (b) (1 point) Consider the situation if `my_func` is called with `d1` equal to {`'b':2, 'c':3, 'd':4`}. Identify one value that, if passed as the last parameter, `d2`, would cause the value 1 to be returned from the function.

   (c) (1 point) Consider the situation if `my_func` is called with `d1` equal to {`'b':2, 'c':3, 'd':4`}. Identify one value that, if passed as the second parameter, `d2`, would cause the value `-1` to be returned from the function.

5. (a) (5 points) Write a function called `add_data` that takes a dictionary `d` and a string `filename`. The function should read in data from the file and add the data to the dictionary `d`. The first line of the file contains the descriptions of data contained in the following lines separated by tabs ('
t'). The keys of the dictionary are tuples of the first two data items in each line and the values of the dictionary are dictionaries themselves, with the remaining data descriptions as keys and the data as values. Part of the function has been written for you.

For example, given a file containing:

```
season   No. in season   title   air date        viewers
2        1        "The North Remembers"   April 1, 2012   3.86
2        2        "The Night Lands"       April 8, 2012   3.76
2        3        "What Is Dead May Never Die"    April 15, 2012  3.77
```

The resulting dictionary would look like:

```
{('2', '1'): {'title': '"The North Remembers"',
              'air date': 'April 1, 2012', 'viewers': '3.86'},
('2', '2'): {'title': '"The Night Lands"',
             'air date': 'April 8, 2012', 'viewers': '3.76'},
('2', '3'): {'title': '"What Is Dead May Never Die"',
             'air date': 'April 15, 2012', 'viewers': '3.77'}}
```

```
def add_data(d,filename):
    # get the lines from the file
    with open(filename, 'r') as file:
        data = file.readlines()
    # get the categories of data contained on each line
    headers = data[0].strip().split('\t')
    # put each line's data in the dictionary
    for i in range(1,len(data)):
    # continue writing the function starting here
```

(b) (5 points) Write a function called `find_viewers_more_than` that given a dictionary of episode info (assume the dictionary resembles the example in question 5a) and an integer threshold will return a list of (season number, episode number) tuples designating episodes with at least threshold viewers.

6. (12 points) For this problem, select one line of code from each of the pairs of lines of code below and reorder them to solve the following problem:

   Write a main function that will prompt the user for a list of filenames containing TV episode data, followed by a threshold number of viewers and displays the title and air date of TV episodes from all files that had at least the threshold number of viewers.

   For example, for the file in question 5a, if the user entered 3.77 as the threshold, the output of the program (after the prompts) would be:

```
"The North Remembers" April 1, 2012
"What Is Dead May Never Die" April 15, 2012
```

```
A1              #print(i, + "not found.")
A2              print(i, "not found.")

B1          print(ep_info[i]['title'], ep_info[i]['air date'])
B2          #print(i['title'], i['air date'])

C1  #def main
C2  def main():

D1      #thresh = input("Minimum viewers: ")
D2      thresh = float(input("Minimum viewers: "))

E1              add_data(ep_info, i)
E2              #add_data(ep_info, filenames)

F1          #except:
F2          except FileNotFoundError:

G1      #for i in range(len(filenames)):
G2      for i in filenames:

H1      for i in qualifying_episodes:
H2      #for i in ep_info:

I1          try:
I2          #try()

J1      #ep_info = []
J2      ep_info = {}

K1      qualifying_episodes = find_viewers_more_than(ep_info,thresh)
K2      #qualifying_episodes = find_viewers_more_than(ep_info)

L1      filenames = input("Enter filenames: ").split()
L2      #filenames = input("Enter filenames: ")
```

Select only 12 lines of code from above, and only one line from each pair. You may fill in line identifiers (*e.g.*, E2) below, or write out the code.

(This page is intentionally blank. Label any work with the corresponding problem number.)

(This page is intentionally blank. Label any work with the corresponding problem number.)