COSC 101, Exam #2
22 March 2017

Name: ______________________________ Section: 9:55 / 1:20 / 2:45

Instructions and advice:

- Do not open the exam until instructed to do so.
- Write your name and circle your section time.
- You have 60 minutes to complete this exam; use your time wisely.
- There are 5 questions and a total of 50 points available for this exam. Don't spend too much time on any one question.
- If you want partial credit, show as much of your work and thought process as possible.
- Since indentation is important in Python, please be sure that your use of indentation is obvious for any code you write.
- When defining functions, it is not necessary to write docstrings nor is it necessary to write comments.
- If you run out of space while answering a question, you can continue your answer on one of the scrap 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 scrap page contains your answer, and (2) on the scrap page, indicate which question you are answering.

Question	Points	Score
1	8	
2	12	
3	8	
4	10	
5	12	
Total:	50	

1. (8 points) Assume that the following statements have already been executed:

```
a = 'word'
b = 3.14
c = [1, 2, 3]
d = [a, b, c]
```

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) `len(d)`

(b) `a[round(b)]`

(c) `a[1:3]`

(d) `d[0][2]`

(e) `d[1][2]`

(f) `d[2][2]`

(g) `c + b`

(h) `d[2] + [b]`

2. (a) (4 points) What is the output of the following program?

```
count_up = 1
count_down = 10
done = False

while not done:
    count_up += 1
    count_down -= 2
    if count_down <= count_up:
        done = True
    print(count_down - count_up)
```

(b) (4 points) What is the output of the following program?

```
def adjust(the_list):
    for i in range(len(the_list)):
        the_list[i] = max(0, the_list[i])

a = [3, -2, 8, -4]
b = a
adjust(b)
print(a[1], b[1])
```

(c) (4 points) What is the output of the following program?

```
month = 'march'
tool = 'hammer'

counts = []
for c in month:
    count = 0
    for d in tool:
        if c == d:
            count += 1
    counts.append(count)
print(counts)
```

3. (8 points) Write a function called `jumps` that takes as a parameter a list of numbers. It should return the number of times that adjacent numbers in the list differ by more than one.

   For example:

   - `jumps([1, 2, 4, 5])` returns 1, because 2 and 4 differ by more than one, but $(1, 2)$ and $(4, 5)$ both differ by exactly one (so those pairs of adjacent items are not counted).
   - `jumps([3, 1.5, 4])` returns 2, because both adjacent pairs of numbers, namely $(3, 1.5)$ and $(1.5, 4)$, differ by more than one.
   - `jumps([1.1, 1.2, 1.0])` returns 0, because all adjacent pairs of numbers differ by no more than one.
   - `jumps([])` and `jumps([5])` both return 0, because there are fewer than two elements in each list, and so no adjacent elements for which to count jumps by more than one.

4. (10 points) Write a program that obtains a list of strings by repeatedly asking the user for input until the empty string is entered. It then prints out the number of strings whose length is greater than the average length of all strings entered.

   (The final empty string should not count towards computing the average string length, and you can assume that at least one nonempty string is entered.)

   Here is an example input/output for such a program:

```
Enter a string: spring
Enter a string: summer
Enter a string: fall
Enter a string: winter
Enter a string:
3 strings have length greater than average
```

   In this example, the average string length is $5.5 = (6 + 6 + 4 + 6)/4$, and three strings have length greater than 5.5 (namely, `'spring'`, `'summer'`, and `'winter'`).

5. This is a two-part question. The second part is on the next page and can be completed even if you have not finished part (a) correctly.

   (a) (5 points) Write a function called `separate_list_items` that takes a single list as a parameter and returns a new list. The returned list will contain two sublists: the first sublist will contain all of the items from the original list with even indexes, the second will contain all of the items with odd indexes. The original list should not be modified.

   For example: Suppose you have `a_list = [ 'a', 'b', 'c', 'd', 'e']`; if you call `separate_list_items(a_list)`, the returned value will be `[['a', 'c', 'e'], ['b', 'd']]`.

(b) (7 points) You have been tasked with dividing a group of attendees into two random teams for a competition at your company picnic. The picnic coordinator has provided you with a long string containing all of the names of the people attending separated by newline characters (`'\n'`). For example:

```
Madeline E Smith\nVijay Ramachandran\nElodie Fourquet\nVal Cucura
```

You need to write a function called `form_picnic_teams` that takes this string as a parameter returns a list where each name has been assigned to one of the two sublists (one sublist for each team). To make sure no team has a competitive advantage, you have been asked to "line people up" alphabetically by first name and assign them one-by-one to each team.

You cannot use `str.split()`. You are required to use the `separate_list_items` function from the previous part, and can assume the function works as described (regardless of whether your answer to part (a) is correct or not).

Note that the names are not provided in order, but recall that lists have a `sort()` method that mutates a list by rearranging its items in order.

(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.)