fullName:_____ andrewID:_____ section:___

15-112 S25
Quiz5 Bonus

Read these instructions carefully before starting:

1. Quiz versions are color-coded. You must have a different version (color) of this quiz than the students sitting to your left and right.
2. Stop writing and submit the entire quiz when instructed by the proctor.
    - Do not unstaple any pages.
    - You must submit the entire quiz with all pages intact.
3. Do not discuss the quiz with anyone else until FRIDAY 5pm.
    - This applies to everyone, including students in either lecture.
4. Do not use your own scrap paper.
    - You should not need scrap paper, there is plenty of room for you on the quiz.
    - However, if you absolutely must use scrap paper, raise your hand and we will provide some. Then, you must write your andrew id clearly on the scrap paper, and hand in the scrap paper with your paper quiz. We will not grade anything on your scrap paper.
5. You may not ask questions during the quiz.
    - The one exception is for English-language clarifications.
    - If you are unsure how to interpret a problem, just take your best guess.
6. Do not use any concepts (including built-in functions) not covered in the notes through week 5 or beyond unit 4.
    - Do not use dictionaries, sets, or recursion.
7. Do not hardcode your solutions.
    - We may test your code using additional test cases.
    - Hardcoding will receive zero points.
8. Assume almostEqual(x, y) and rounded(n) are both supplied for you.
    - You must write all other helper functions you wish to use, unless we specify otherwise.
9. Good luck!

BONUS QUIZ: Solving these problems correctly can earn you a *very* small number of bonus points, but cannot lower your score (so doing well can only help, not hurt, so don't stress and just do your best).

Bonus Fill In The Blank: smallestAlikeSum

Background: this problem only concerns lists that contain exactly 3 positive integers. Say we have two such lists, `L` and `M`. We will say that `L` and `M` are "alike" (a coined term) if they contain the same digits in the same order, though perhaps making up different integers.
For example, the list `[1, 23, 45]` is alike each of these lists:

```
[1, 2, 345]
[1, 23, 45]
[1, 234, 5]
[12, 3, 45]
[12, 34, 5]
[123, 4, 5]
```

With this in mind, complete the function `smallestAlikeSum(L)` by filling in the blank on the following page. This function takes a list `L` that contains exactly 3 positive integers, return the list that is alike to `L` with the smallest sum of its values. You can break ties in any way.

For example, if `L = [1, 23, 45]`, here again are the alike lists with their sums:

```
[1, 2, 345] # sum = 348
[1, 23, 45] # sum =  69
[1, 234, 5] # sum = 240
[12, 3, 45] # sum =  60
[12, 34, 5] # sum =  51
[123, 4, 5] # sum = 132
```

The smallest of these sums is 51. Thus:

```
assert(smallestAlikeSum([1, 23, 45]) == [12, 34, 5])
```

Fill in the blank on the next page.

Note: To earn points, you may only write a single line of code in the blank, and you may not change any other part of the code, or simply rewrite a different solution. Try to understand how this approach is solving the problem and insert the missing line.

```
def smallestAlikeSum(L):
    result = L # start with L itself as the best option

    # d is a string containing all the integers in L in order:
    d = ''.join([str(v) for v in L])

    # Now find all the locations (i and j) before which we can
    # add the two commas:
    for i in range(1, len(d)-1):
        for j in range(i+1, len(d)):

            # Complete the line below to complete the function
            L =
            if sum(L) < sum(result):
                result = L
    return result

def testSmallestAlikeSum():
    print('Testing smallestAlikeSum()...', end='')
    assert(smallestAlikeSum([1, 23, 45]) == [12, 34, 5])
    assert(smallestAlikeSum([543, 2, 1]) == [5, 43, 21])
    assert(smallestAlikeSum([1, 2, 34567]) == [123, 45, 67])
    print('Passed!')

testSmallestAlikeSum()
```

Bonus Code Tracing
For each bonusCt, indicate what the code prints
Place your answer (and nothing else) in the box below the code.

bonusCt1:

```
def bonusCt1(L):
    c = d = 0
    while L:
        c += 1
        for i in range(len(L)-1, -1, -1):
            L[i] -= 1
            if not L[i]:
                L.pop(i)
                if not d: d = c
    return c - d
print(bonusCt1([v**2 for v in [-6, -3, 2, 5]]))
```

bonusCt2:

```
import string
def bonusCt2(L):
    u = string.whitespace * len(L)
    M = ['' for v in L] * len(L)
    for v in L:
        for t in v:
            if t not in u and t not in M:
                M.insert(0, t)
    return ''.join(M)
print(bonusCt2(['a test', 'this is a test', 'yes']))
```