Name:__________________________________ Section:___ Andrew Id: ____________________ a

15-112 Spring 2020 Quiz 3
Up to 20 minutes. No calculators, no notes, no books, no other paper, no computers.
No lists, list indexing, or recursion
You may call almostEqual(x, y) and roundHalfUp(d) without writing them. Write everything else!

1. Code Tracing [20pts]

Indicate what the following code prints. Place your answer (and nothing else) in the box to the right.

```
def ct1():
    s = 'a'
    t = s
    s += 'b'
    t += 'c'
    print(1, s + '-' + t)
    s = 'a\nb'
    print(2, f'{s}{len(s)}{repr(s)}')
    s = 'This is amazing!'
    print(3, s[1:].find('is'))
    s = 'abcdba'
    t = ''
    v = chr(ord('b')-1)
    for w in s.split('b'):
        t += f'{ord(w[0]) - ord(v)}'
        v = w[0]
    print(4, t)
ct1()
```

2. Reasoning over Code [15pts]

Find an argument for the function rc1(n) that makes it return True. Place your answer (and nothing else) in the box to the right.

```
def rc1(s):
    assert(s.isdigit())
    for d in range(10):
        c = str(d)
        if (d < 3):
            assert((s.count(c) == d+1) and
                   (s.find(c) == d+1) and
                   (s.find(c+c) < 0))
        elif (d > 3):
            assert(c not in s)
    return ((s.count('3') == 2) and
            (int(s[-2:]) > 29))
```

s =

3. Free Response: `lowercasePal(s)` [25pts]

Write the function lowercasePal(s) that takes a string s and returns True if the string formed just by the lowercase letters in s form a palindrome, and False otherwise. Here are some test cases:

```
    assert(lowercasePal('1A 2b 3c 4b') == True)  # 'bcb'
    assert(lowercasePal('1a 2B 3c 4b') == False) # 'acb'
    assert(lowercasePal('I see sPOT!') == True)  # 'sees'
    assert(lowercasePal('YES!') == True)         # ''
```

The quiz continues on the next sheet

4. Free Response: `getBestAvg(scores)` [40pts]

Write the function getBestAvg(scores) that takes a non-empty multiline string of scores, where each line is a non-empty comma-separated list of possibly-negative integers, and returns the highest average from any line. Here are some test cases (which use almostEqual due to the results being floats and not ints):

```
    data1 = '''\
5,10,15
2,3,4'''
    assert(almostEqual(getBestAvg(data1), 10))
    data2 = '''\
5,10,15
11'''
    assert(almostEqual(getBestAvg(data2), 11))
    data3 = '''\
-123'''
    assert(almostEqual(getBestAvg(data2), -123))
```

5. Bonus/Optional CT [3pts; 1.5pts each]

```
def bonusCt1(s):
    for _ in range(4): s = repr(s)
    return s.count('\\')
print(bonusCt1('abc'))
```

```
def bonusCt2(s):
    def f(s,d):
        if (len(s) > 4): return s+str(d)
        return f((s+str(d))*d,d+1)
    return f(f(s,1),4)
print(bonusCt2('a'))
```