| fullName: | _andrewID: |
|-------------------|------------|
| recitationLetter: | |

15-112 N22

Quiz4 [30 min.]

You **MUST** stop writing and hand in this **entire** quiz when instructed in lecture.

- You may not unstaple any pages.
- You may not use your own scrap paper. If you must use additional scrap paper, raise your hand and we will provide some. You must hand this in with your paper quiz, and we will not grade it.
- Failure to hand in an intact quiz will be considered cheating. Discussing the quiz with anyone in any way, even briefly, is cheating. (You may discuss it only once the quiz has been posted to the course website.)
- You may not use any concepts (including builtin functions) we have not covered in the notes this semester.
- You may not use recursion.
- We may test your code using additional test cases. Do not hardcode.
- We do not deduct points for bad style on quizzes
- Assume almostEqual(x, y) and roundHalfUp(n) are both supplied for you. You must write all other helper functions you wish to use.

CT1: Code Tracing [9pts]

Indicate what the following code prints. Place your answers (and nothing else) in the box below.

```
#Hint: Draw a box and arrow diagram!
# Note: this prints 3 lines
import copy
def ct1(L, A, n):
    A[-1][0] += n
    A[n%2] += [10*n]
    L.append(n)
    print(A)

L = [[2], [3]]
C = copy.copy(L)
D = copy.deepcopy(L)
ct1(L, C, 1)
ct1(L, D, 2)
print(L)
```

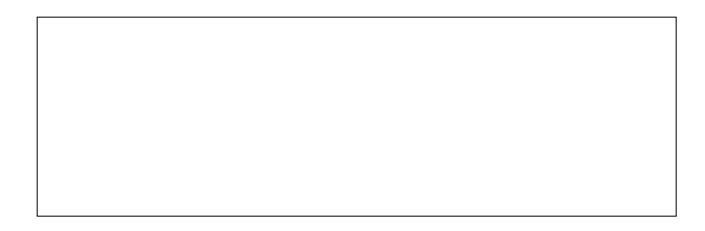


CT2: Code Tracing [9pts]

Indicate what the following code prints. Place your answers (and nothing else) in the box below.

```
#Hint: Draw a box and arrow diagram!
# Note: this prints 3 lines

def ct2(L):
    A = [set()]
    for item in L:
        if item in A[0]:
            print(A[0])
            A.insert(0, {item})
        else:
            A[0].add(item)
        return A
```



Free Response 1: makeAuthorDict(books) [32pts]

Write the function makeAuthorDict(books) which takes a dictionary that maps book titles to their authors, and returns a new dictionary mapping each author to a set of their books, like so:

You may assume each book only has one author.

You may continue your FR1 answer here, if you wish

Free Response 2: makeTable(n) [32 points]

Write the function makeTable(n) that takes a positive integer n and returns a multiplication table in the form of a 2D list with n rows and n columns. Each cell should be the product of the integer at the beginning of its row and the top of its column. Look at the test cases to identify this pattern.

```
def testMakeTable():
   print("Testing makeTable(n)...", end="")
   assert(makeTable(1) == [[1]])
   assert(makeTable(2) == [[1, 2],
                           [2, 4]])
   assert(makeTable(3) == [[1, 2, 3],
                           [2, 4, 6],
                           [3, 6, 9]])
   assert(makeTable(4) == [[1, 2, 3, 4],
                           [2, 4, 6, 8],
                           [3, 6, 9, 12],
                           [4, 8, 12, 16]])
   assert(makeTable(5) == [[1, 2, 3, 4, 5],
                           [2, 4, 6, 8, 10],
                           [3, 6, 9, 12, 15],
                           [4, 8, 12, 16, 20],
                           [5, 10, 15, 20, 25]])
   print('Passed!')
```

You may continue your FR2 answer here, if you wish

True or False [2pts ea]

Write only the whole word "True" or "False" (and not just T or F).

| Also, assume we are always talking at trick you. | pout very large values of n, and we are not trying to |
|--|---|
| TF1: The Big-O of merge so | ort is O(n*logn) |
| TF2: The Big-O of selection | sort is O(n*logn) |
| TF3: Selection sort is gener | ally slower than merge sort |
| TF4: Sets cannot contain in | nmutable values |
| TF5: Sets cannot contain do | uplicate values |
| TF6: Dictionary keys must b | pe immutable |
| TF7: Dictionary values mus | t be immutable |
| TF8: Linear search can only | be used when a list is known to be sorted |
| TF9: O(100N) is slower than | n O(N**100) |

bonusCT: Code Tracing [2pts]

This question is optional. Indicate what the following code prints. Place your answers (and nothing else) in the box below.

```
def bonusCt(L, s):
    L = [chr(ord('a')+L[i]+i) for i in range(len(L))]
    s = sorted(set(s) - set(L)) * len(set(s))**len(L)
    return ''.join(s).count('rb')
print(bonusCt([2, -1], 'abracadabra'))
```