Name: _____ Andrew Id: _____

15-112 Spring 2024 Quiz 8

Up to 25 minutes. No calculators, no notes, no books, no computers. Show your work!
Do not use try/except or recursion on this quiz.

1. (6 points) **Code Tracing**: Indicate what the following program prints. Place your answers (and nothing else) in the box next to the code.

```
def CT(L):
    s = set(L)
    d = dict()

    for value in s:
        d[value[0].upper()] = d.get(value[0].upper(), []) + [value]

    return d

L = ["Amna", "Reem", "yomna", "reem", "Yaman", "aya", "Reem"]
d = CT(L)

for e in d:
    print(f"{len(d[e])} {e}s : {d[e]}")
```

2. **Big O**:

In this problem you will be calculating the efficiency of a provided piece of code. Consider the following example:

```
def bigOSample(n): # N is n
    print("Simple")     # 1 step
    for i in range(n):  # 1 step for range and loop runs N times
        print(i)        # 1 step
                        # 1 step: Update i at end of loop

# Total Steps Count: 2 + 2N
# BigO Time Efficiency: O(N)
```

(a) (4 points) Write line-by-line step count (In the comment space next to each line).

```
def bigO(L): # L is an NxN list

    length = len(L)    # _____________________________________________

    for i in range(0, length, length//4): # ____________________________

        L[i].sort() # ______________________________________________

        sortedRow = L[i] #_________________________________________

        s = set(sortedRow) # ______________________________________

        if len(sortedRow) == len(s): #____________________________

            print("A unique Row") #_____________________________

        else: #____________________________________________________

            print("Row is not unique") # ____________________________

            L[i] = list(s) # ________________________________________

        # ____________________________________________________________

    return s # ____________________________________________________
```

(b) (0.5 points) Write the total number of steps.

```
Total Steps Count = ____________________________________________
```

(c) (1.5 points) Write the simplified BigO complexity (i.e. not including lower order terms or coefficients).

```
BigO Time Efficiency = _____________________________________________
```

3. (8 points) **Free Response**: Fortunate Tuples

   A *fortunate tuple* (coined term) is a tuple that has a frequency in the list equal to its length.

   Write function `fortunateTuples(L)` which takes a list (L) of tuples and returns a set containing all the fortunate tuples in the list. Your solution should run in O(N) time.

   Consider the following example:

```
assert(fortunateTuples([(1, 2), (3, 4), (1, 2), (5,), (3, 4, 5)])== {(1, 2), (5,)})
assert(fortunateTuples([("A", "B"), ("C", "D")] == set())
assert(fortunateTuples([]) == set())
```