fullName:____________________________andrewID:____________________________ recitationLetter:_____

15-112 S24

WS9+Quiz9 version B

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

- You may not unstaple any pages.
- 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 your own scrap paper. If you must use additional scrap paper, raise your hand and we will provide some. You must hand any scrap paper in with your paper quiz, and we will not grade it.
- You may not ask questions during the quiz, except for English-language clarifications. If you are unsure how to interpret a problem, take your best guess.
- You may not use any concepts (including builtin functions) which we have not covered in the notes in weeks 1-9.
- You may not use recursion.
- We may test your code using additional test cases.
- 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.

Writing Session 9 (10% of HW9)

WS1. isPermutation(L)

A permutation of a list L is a list which contains the same elements as L, but in any order. With this in mind, write the function isPermutation(L), which takes a list L, and returns True if L is a permutation of the list of numbers from 0 to (n - 1) inclusive, and False otherwise, where n is the length of L.

Note: you cannot create new lists or mutate existing lists. Instead, think of a clever way to use sets to solve the problem.

Quiz8

CT1: Code Tracing [10pts]

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

```
def ct1(L):
    s = set()
    t = set()
    for v in L:
        if v in s:
            s.remove(v)
            t.add(v)
        else:
            s.add(v)
    print(s)
    print(t)
    return t-s
print(ct1([1,2,2,3,3,3,4,4,4,4]))
```

CT2: Code Tracing [10pts]

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

```
def ct2(L):
    d = dict()
    for (s, n) in L:
        k = n%10
        d[k] = d.get(k, str(n)) + s
    return d
print(ct2([ ('A', 18), ('B', 37), ('C', 78), ('D', 17), ('E', 24) ]))
```

Efficiency (big-oh) [10pts total]

For each of the following functions, assuming the list L contains n values, what is the big-oh runtime? For each function, write the correct letter from the following options in the box below that function:

```
A) O(N**2)
B) O(NlogN)
C) O(N)
D) O(logN)
E) O(1)
F) None of these
```

```
def f1(L):
    M = [ ]
    for x in L:
        for y in L:
            M.append(x*y)
    return M
```

```
def f2(L):
    z = 0
    for x in L:
        y = 2*x
        if y in L:
            z += L.count(y)
    return z
```

For each of the following functions, assuming the list L contains n values, what is the big-oh runtime? For each function, write the correct letter from the following options in the box below that function:

```
A) O(N**2)
B) O(NlogN)
C) O(N)
D) O(logN)
E) O(1)
F) None of these
```

```
def f3(L):
    return len(L)
```

```
def f4(L):
    return [4] * len(L)
```

```
def f5(L):
    i = 1
    z = 0
    while i < len(L):
        z += L[i]
        i *= 2
    return z
```

Short Answers [20pts total]

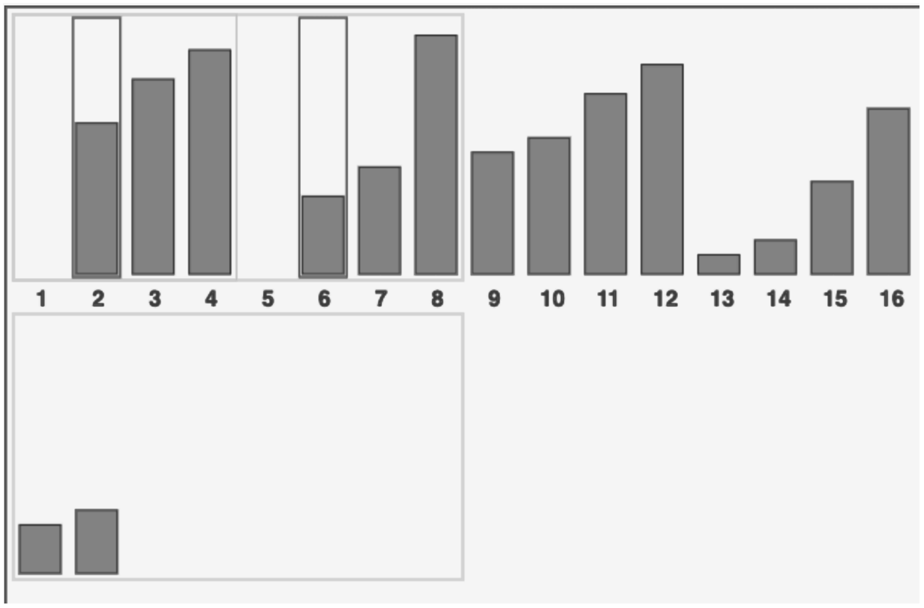
SA1. For a Python list of length N, write the big-oh runtime of each of the following (on the line provided):

```
A) binary search ______________________________

B) linear search ______________________________

C) selection sort _____________________________

D) merge sort _________________________________
```

SA2. Assuming L is a Python list of length N, what is the big-oh worst-case runtime of each of the following (on the line provided):

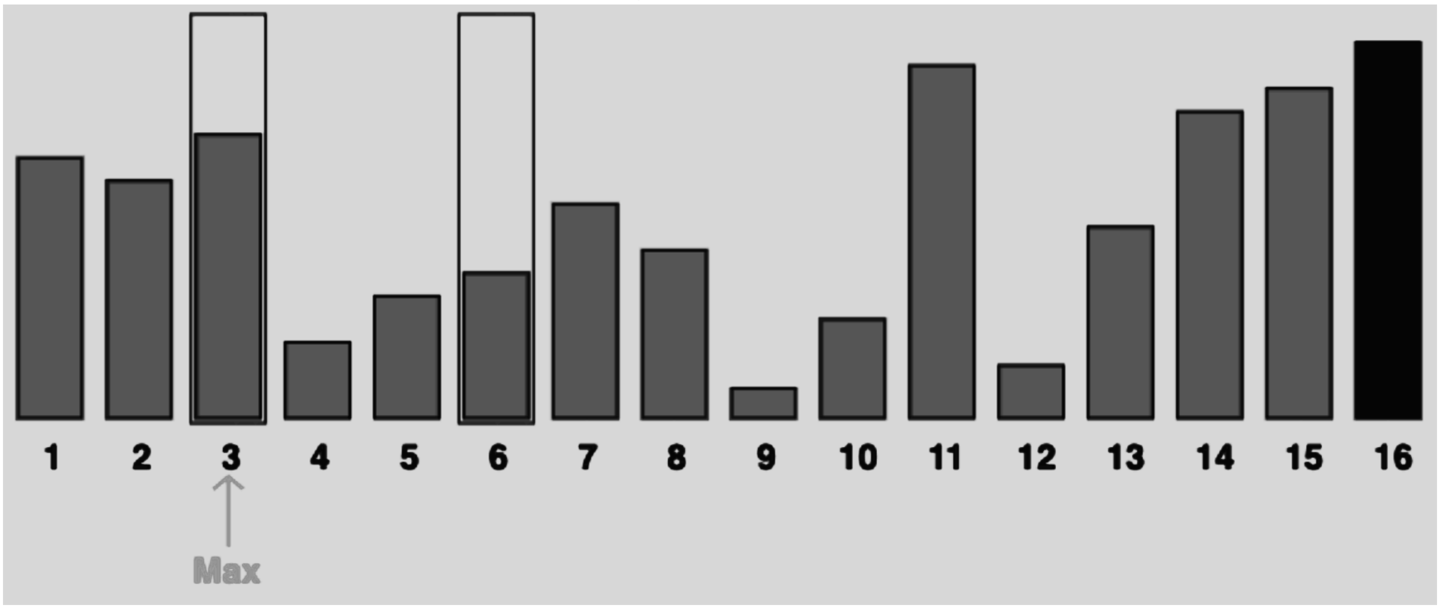
```
A) L.sort() __________________________________

B) L.append(42) ______________________________

C) L.insert(0, 42) ___________________________

D) S = set(L) ________________________________
```

SA3. Here is a screenshot of xSortLab running merge sort::



Given this state, what is the index of the next value to be copied to the "merged" list?

SA4. Here is a screenshot of xSortLab running selection sort:



Given this state, what are the indexes of the next two values to be swapped?

Free Response 1: sameXandY(L) [25pts]

Write the function `sameXandY(L)` that takes a list L that contains `n` (x, y) tuples, and returns True if every x value occurs as some y value and every y value also occurs as some x value, and False otherwise. If the list is empty, return False.
Note: To receive full credit, your function must run in O(n) where n is the length of L.

For example:
`assert(sameXandY([(1,2),(2,3),(3,2),(1,1)]) == True)`
Here, the x values are 1, 2, 3, 1, and the y values are 2, 3, 2, 1.
This meets the requirement, so the function returns True.
As another example:
`assert(sameXandY([(1,2),(2,3),(3,2),(4,1)]) == False)`
Here, the x values are 1, 2, 3, 4, and the y values are 2, 3, 2, 1. Since the x value 4 does not occur as a y value, the function returns False.

```
def testSameXandY():
    assert(sameXandY([ ]) == False)
    assert(sameXandY([(1,2)]) == False)
    assert(sameXandY([(2,2)]) == True)
    assert(sameXandY([(1,2),(2,1)]) == True)
    assert(sameXandY([(1,2),(2,3)]) == False)
    assert(sameXandY([(1,2),(2,3),(3,2),(1,1)]) == True)
    assert(sameXandY([(1,2),(2,3),(3,2),(4,1)]) == False)
```

Begin your FR1 answer here or on the following page

Begin or continue your FR1 answer here:

Free Response 2: makePlayerReport(gameReport) [25pts]

Background: This problem involves some friends playing games of connect4. Each game has one winner and one loser. The "gameReport" is a multiline string where each line is of the form "{name1} defeated {name2}", like so:

```
"""\
Cam defeated Bob
Ann defeated Cam
Ann defeated Bob
"""
```

Given such a gameReport, we can generate a "playerReport" like so:

```
{
    'Ann': { 'wins': 2, 'losses': 0 },
    'Bob': { 'wins': 0, 'losses': 2 },
    'Cam': { 'wins': 1, 'losses': 1 }
}
```

Note that this playerReport is a dictionary mapping player names to another dictionary that maps 'wins' to the number of wins for that player and 'losses' to the number of losses for that player.

With this in mind, write the function makePlayerReport(gameReport) that takes a gameReport and returns a playerReport (both of these as described above).

```
def testMakePlayerReport():
    print('Testing makePlayerReport()...', end='')
    gameReport = """\
Cam defeated Bob
Ann defeated Cam
Ann defeated Bob
"""
    assert(makePlayerReport(gameReport) == {
            'Ann': { 'wins': 2, 'losses': 0 },
            'Bob': { 'wins': 0, 'losses': 2 },
            'Cam': { 'wins': 1, 'losses': 1 }
        })
```

Begin your FR2 answer on the following page.

Begin your FR2 answer here:

Begin your FR2 answer here:

Continue your FR2 answer here:

The problems below are not required. Indicate what the following code prints. Place your answers (and nothing else) in the boxes below.

bonusCt1 [optional, 1pt]

```python
def bonusCt1(L, T):
    s = set(str(L))
    t = set(str(T))
    f = lambda s: int(''.join(sorted([c for c in s if c.isdigit()])))
    return f(t - s) + f(s - t)
print(bonusCt1([1, 23, 40], (71, 43)))
```

bonusCt2 [optional, 1pt]

```python
def bonusCt2(L):
    r, s, m = set(), set(), max(L)
    for v in range(2, m+1):
        if v not in s:
            r.add(v)
            for q in range(2*v, m+1, v): s.add(q)
    return [v for v in sorted(r) if v not in L]
print(bonusCt2(list(range(3, 13, 2)) + list(range(17, 27, 4))))
```