Name: _____ Andrew Id: _____

15-112 Spring 2022 Quiz 8

Up to 25 minutes (up to 20 minutes for 20% proficiency bonus) . No calculators, no notes, no books, no computers. Show your work!

Do not use try/except on this quiz.

1. (4 points) **Short Answer**: Consider the following code:

```
def f(a):
  t = 0
  for e in a:
    for t in a:
      t = t + 1
  return t
```

Big-O time efficiency of the function if:

(a) `a` is a `list` _____.

(b) `a` is a `set` _____.

(c) `a` is a `dict` _____.

(d) `a` is a 5-letter english word (`string`) _____.

2. (4 points) **Code Tracing**: Indicate what the following program prints. Place your answer (and nothing else) in the box next to the code.

```
def ct(lst):
    d1 = dict()
    for i in range(len(lst)):
        if lst[i] not in d1:
            d1[lst[i]] = 0
        d1[lst[i]] += 1
    print(d1)
    d2 = dict()
    for key in d1:
        d2[d1[key]] = key
    print(d2)
    for key in d2:
        d1[key] = d2[key]*2
    for key in d1:
        print("key:", key, "value:", d1[key])
    return d1.get(4, 1)
lst = [1, 5, 1, 1, 2, 5]
print(ct(lst))
```

3. (4 points) **Reasoning Over Code**: Find an argument, `d`, for the following function to cause it to return `True`. Place your answer (and nothing else) in the box below the code.

```
def rc(d):
    assert(len(d) == 4)
    for key in sorted(d.keys()):
        assert(len(key) == 2)
        assert(key[0] == 'a')
        if d[key] != int(key[1]):
            return False
    return True
```

4. (8 points) **Free Response** Write the function `dictDifference(d1, d2)` that takes two dictionaries whose values are sets and returns a new dictionary that results from taking `d1` and removing the values in `d2` for the common keys. If one the values becomes an empty set in the resulting dictionary, then the key should be removed. The values of non-common keys should remain the same. The function should be **non-destructive**.

   For instance,

   ```
   dictDifference({'cmu':{1,2,3}, '112':{4,5}}, {'112':{5}})
   ```

   should return

   ```
   {'cmu': {1, 2, 3}, '112': {4}}
   ```

   because the only common key is `'112'` and when we remove the values `d2['112']` from `d1['112']` we get the set `{4}`.

   ```
   dictDifference({'cmu':{1,2,3}, '112':{4,5}}, {'112':{4,5,6}})
   ```

   should return

   ```
   {'cmu': {1, 2, 3}}
   ```

   because the only common key is `'112'` and when we remove the values `d2['112']` from `d1['112']` we get an empty set and we must remove the key `'112'`.

   Finally,

   ```
   dictDifference({'cmu':{1,2,3}, '112':{4,5}},{ 'cmu': {3,4,5}, '112':{6,7,8}})
   ```

   should return

   ```
   {'cmu': {1, 2}, '112': {4, 5}}
   ```

   because when we remove the values `d2['cmu']` from `d1['cmu']` we get `{1,2}` (only 3 is common to both sets), and when we remove values `d2['112']` from `d1['112']` nothing changes (the sets are disjoint).

Free Response answers: