Name:_____ Section:___ Andrew Id: _____

**15-112 Fall 2016 Quiz7a**

*** Up to 30 minutes. No calculators, no notes, no books, no computers. * Show your work! * No recursion**

1. **Big-Oh** [15 pts]:
   What is the worst-case big-oh runtime of each of the following, in terms of N? Assume L is a list of N integers, and x is a positive integer where N = math.log(x, 2). Place your answer (and nothing else) in the box next to the code.

```
def bigOh1(L):
    N = len(L)
    for value in sorted(L * N):
        L.append(value)
    return list(reversed(L[0: len(L): 3]))
```

```
def bigOh2(L):
    N = len(L)
    s, t = set(L), set()
    for i in range(len(L)):
        t.add(L[i]**2)
        for j in range(len(L)//4, i):
            for k in range(5):
                s.add(L[i] + L[j] - k)
    return t - s
```

```
def bigOh3(x):
    N = math.log(x, 2)
    c = 1
    while (x > 0): (x, c) = (x//42, c+1)
    x = 1
    while (x**2 < c): x += 1
    return x
```

2. **Short Answer** [20 pts]
   Be very brief.

   a. Given the list [5,2,7,3], what will the list be after exactly 2 swaps are made in bubbleSort?

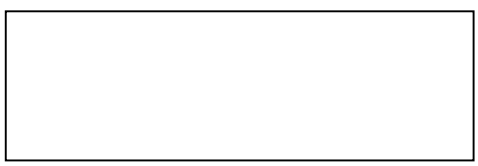
   b. Given the list [5,2,7,3], what will the list be after exactly 2 swaps are made in selectionSort, as it works in xSortLab?

c. State and prove the worst-case big-oh of mergesort. Note that a well-labeled picture can be sufficient proof.

d. Assume that s is a string of lowercase letters, and hash(s) is defined as such:
def hash(s): return sum([ord(c)-ord('a') for c in s])
Further, assume a hashtable H has 3 buckets, and that buckets for each value are chosen in the way we described in lecture. Write the resulting hashtable H after all the elements in the list ['ab', 'bc', 'e', 'cd', 'abc'] are added.

3. **Code Tracing** [10 pts]:Indicate what this prints. Place your answer (and nothing else) in the box below the code.

```
def ct1(n):
    s, t = set(), set()
    while (n > 0):
        (d, n) = (n%10, n//10)
        if (d in t): t.remove(d)
        elif (d in s): t.add(d)
        s.add(d)
    return sorted(t)
print(ct1(13051231))
```



Name:_____ Section:___ Andrew Id: _____

```
# Code Tracing (continued)

def ct2(d, key):
    while (key in d) and ((key+2) not in d):
        d[key+2] = key+1
        key = d[key]
    L = [ ]
    for key in sorted(d.keys()):
        L.append(10*key + d[key])
    return L
print(ct2({1:5, 0:2}, 0)) # prints 5 ints
```

4. **Reasoning Over Code** [5 pts]:
Find an argument for the following function that makes it return True. Place your answer (and nothing else) in the box below the code:

```
def rc1(d):
    i = j = k = m = 0
    for key in d:
        i += 1
        value = d[key]
        if (key == value):
            j = min(value, j)
            k = max(value, k)
        else:
            m += 1
    return ((i, j, k, m) == (4, -2, +2, 1))
```

d =

5. **Free Response: mostPopularFriend(d)** [50 pts]
Recall that friendsOfFriends(d) takes a dictionary d like this:

```
d = dict()
d["fred"]   = set(["wilma", "betty", "barney"])
d["wilma"]  = set(["fred", "betty", "dino"])
```

With this in mind, write the function mostPopularFriend(d) that takes a dictionary of that form, and returns the name that occurs the most number of times in all the sets of friends. In the example above, mostPopularFriend(d) would return "betty". You may assume that there is exactly one such name, so ignore ties.

6. **Bonus/Optional: Code Tracing** [5 pts; 2.5 pts each] What will these print? Place your answer in the boxes.

```
def bonusCt1(d, L):
    for v in L[:]: L *= v
    for v in L[:]: d[v] = 1 + d.get(v,42)
    return sum(d.values())
print(bonusCt1({1:2, 2:3, 3:1}, list(range(1,5))))
```

```
def bonusCt2(n, f=lambda v:''.join(sorted(v))):
    s = set(str(tuple([n*n]*n)))
    t = set(list(map(str, range(10))))
    return f(s-t), f(t-s)
print(bonusCt2(7))
```