

Name: \_\_\_\_\_ Section: \_\_\_\_ Andrew Id: \_\_\_\_\_

### 15-112 Fall 2014 Quiz 5

\* 17½ minutes. No calculators, no notes, no books, no computers.

\* You may not discuss any portion of this quiz with anyone until after 5pm today.

\* SHOW YOUR WORK, CIRCLE YOUR ANSWERS.

1. **Code Tracing** [30 pts]: Indicate what this will print:

```
1) a = range(2,10,7)
   (b, c) = (a, a[:2])
   b[0] += 3
   a += [3]
   a = a + [4]
   print c + [b[0]]
   print c.append(b[1])
   print a, b, c
```

```
2) import copy
   def f(a, b):
       a = copy.copy(a)
       a[0] = b[1]
       b[0] = a[1]
       return a + b
   a1 = a2 = range(5,7)
   b1 = b2 = range(2)
   a1 = f(a1, b1)
   print a1, a2, b1, b2
```

2. **Reasoning Over Code** [10 pts]

Find arguments for the following function that make it return True. You only need one set of arguments for the function, even if there are multiple correct answers.

```
def f(a):
    assert(type(a) == list)
    b = a + [ ]
    for i in xrange(len(a)):
        if (i % 2 == 1):
            b[i] -= i
    return (b == range(5,10))
```

3. **Short Free Response: median(a)** [20 pts]

Write the non-destructive function median(a) that takes a list of floats and returns the median value, which is the value of the middle element, or the average of the two middle elements. If the list is empty, return None.

4. **Short Free Response: prepend(a,b)** [20 pts]

Write the function `prepend(a, b)` which destructively modifies `a` (without modifying `b`) by adding all the values of `b` onto the front of `a`, and returns the usual value returned by destructive functions. So this code works:

```
a = [1, 2]
b = [3, 4]
prepend(a, b)
assert((a == [3,4,1,2]) and (b == [3,4]))
```

5. **Short Free Response: noDuplicateTypes(a)** [20 pts]

Write the function `noDuplicateTypes(a)` that takes a list of values and returns `True` if no two values in the list are the same Python type, and `False` otherwise.

6. **Bonus/Optional:** [3 pts each]

1) Indicate what this will print:

```
def f(a, n):
    try: a.append(2*n + a[0]); return n + f(a[3:], n+1)
    except: return 42
a = range(3); print f(a, 3), a
```

2) Find arguments for the following function that make it return `True`. For credit, be sure to show your work!

```
def f(a):
    assert(False not in [val>0 for val in a])
    a = [10**i*a[i] for i in xrange(len(a))]
    return sum(a) == int(string.digits[:6])
```