Name:_____ Section:___ Andrew Id: _____

**15-112 Spring 2016 Quiz 3xa**

*** Up to 20 minutes. No calculators, no notes, no books, no computers.**

*** No strings, lists, or recursion! * To receive credit (in Code Tracing), show your work.**

1. **Quick Answer** [15 pts]:
   Write a function f(n) so that it behaves *exactly* the same way as w(n), but using a for loop instead of a while loop.

```
def w(n):
    x = 0
    while (x**2 < n):
        print("x is", x)
        x += 2
```

2. **Code Tracing** [20 pts]:Indicate what this prints. Place your answer (and nothing else) in the box below the code.

```
def ct1(n):
    x = n
    k = 0
    while (x > 0):
        total = 0
        y = n
        for i in range(k):
            total += y%10
            y //= 10
        print(k, total)
        x //= 10
        k += 1

print(ct1(1234))
```

3. **Reasoning Over Code** [10 pts]: Find arguments for the following function that makes it return True. You only need one set of arguments, even if there are multiple correct answers.

```
def f(x, y):
    assert((type(x) == int) and (type(y) == int))
    if ((x <= 50) or (y > 25)): z = 3
    elif (x%10 + y%10 > 0): z = 42
    elif (x == y + 40): z = 10
    else: z = 5
    return (z == 2**5//3)
```

4. **Free Response**: **nthSteppingNumber(n)** [55 pts]
   A number n is a stepping number (a coined term) if it is a non-negative int and its digits are strictly increasing from left to right. For example, 1379 is a stepping number because 1 < 3 < 7 < 9, but 13379 is not because 3 is not strictly greater than 3. Note that a one-digit number is always a stepping number. With this in mind, write the function nthSteppingNumber(n) that takes a possibly-negative int n and returns the nth stepping number.

   Here are a couple test cases:

```
assert(nthSteppingNumber(0) == 0)
assert(nthSteppingNumber(10) == 12)
assert(nthSteppingNumber(433) == 145679)
```