Name:_________________________________ Section:___ Andrew Id: ____________________

15-112 Spring 2017 Quiz 2a

*** Up to 25 minutes. No calculators, no notes, no books, no computers. * Show your work!**

*** No strings, lists, or recursion**

1. **Code Tracing** [20 pts]:Indicate what these print. Place your answers (and nothing else) in the boxes below the code.

```
def ct1(x, y, z):
  count = 0
  while (x > y):
    x //= 2
    y -= z
    z += 1
    print(x, end=' ')
    print(y, end=' ')
  return z
print(ct1(20, 10, 2)) # prints 5 values
```

```
def ct2(x):
  y = 3
  while True:
    print(y, end=' ')
    for z in range(1, x*y, 2):
      if (z % 3 == 1):
        print(z, end=' ');
        continue
      y += x
      if (y % 5 > 0):
        print(y, end=' ')
        break
      print(y, end=' ')
    if (y > 10): return x
    y += 1
print(ct2(7)) # prints 5 values
```

2. **Reasoning Over Code** [10 pts]:
   Find an argument for the following function that makes it return True. Place your answers (and nothing else) in the boxes below the code:

```
def rc1(n):
  if (n == 0): return False
  count = 0
  for x in range(0, 100, n):
    count += 1
  # hint: here, x equals the last value in the range
  return ((count == 5) and (x//10 == x%10 + 4))
```

n =

Quiz continues on next page

Name:___________________________________ Section:___ Andrew Id: _____________________

3. **Free Response 1: nthDecreasingOddsNumber(n)** [35 pts]
Background: we will say that a positive integer is a "decreasing odds number" (a coined term) if all the digits are odd and each digit is smaller than the one before it, moving left to right. So 973 and 91 are decreasing odd numbers, but 977, 9073, 379, 963 and 972 all are not. With this in mind, write the function nthDecreasingOddsNumber(n), which you can abbreviate as nd(n), which takes a non-negative int n and returns the nth decreasing odds number. Note that nd(0) should return 1, and the first several decreasing odds numbers are: 1, 3, 5, 7, 9, 31, 51, 53, 71, 73, 75, 91, 93, 95, 97, 531, 731, 751, 753, 931,...

Quiz continues on next page

4. **Free Response 2: latticePointCount(f, x1, x2)** [35 pts]
Background: a lattice point is a point (x,y) where x and y are both integers. With this in mind, write the function latticePointCount(f, x1, x2) that takes a function f and two floats x1 and x2 where x1 < x2, and returns the number of lattice points f(x) passes through when x1 <= x <= x2. Since we are using floats here, consider a number an integer if it is almost equal to the nearest integer. An example may help:

```
def f(x): return x/2 + 0.5
print(latticePointCount(f, 0.8, 4.7))
```

We are checking for lattice points for 0.8 <= x <= 4.7. But we only need to consider integer values for x (right?), so we consider 1 <= x <= 4:

```
f(1) = 1.0  # so (1,1) is a lattice point!
f(2) = 1.5
f(3) = 2.0  # so (3,2) is a lattice point!
f(4) = 2.5
```

So there are 2 lattice points in the range, and latticePointCount(f, 0.8, 4.7) returns 2. Note: you may not assume that almostEqual is already written for you.

5. **Bonus/Optional: Code Tracing** [5 pts] Indicate what this prints. Circle your answer below each function.

```
def bonusCt1(x, y=0):
  def f(x):
    for y in range(x): x += 2*y
    return x
  for x in range(f(x), f(f(x)), x):
    if (x%10 + x//10 > 13): y = 100*y + x
  return y
print(bonusCt1(3))
```

```
def bonusCt2(n):
  (a,b,c) = (0,1000, 100)
  while (c < 1000):
    for x in range(a, b, c):
      (a,b,c) = (a+1, b-1, c+50)
  return a-n
print(bonusCt2(2))
```