Name:_____

1. In lecture 1 we considered an algorithm to exit the building if the fire-alarm sounded. For walking, we decided that we needed to repeat two operations many times:

- move left foot forward
- move right foot forward

If we knew how far we needed to walk and how big each step was, then we might know how many times to repeat these operations, called a "count-controlled" loop. Alternatively, we might repeat these operations until some condition (e.g., until we reach the door) is satisfied, called conditional iteration.

Most languages have a `for` loop to aid in writing "count-controlled" loops where the number of iterations is known in advance. In Python, the `for` loop iterates once for each item in some sequence type (i.e, a list, tuple, string). A simple example printing each number in the list [1, 3, 9, 7] is:

```
for value in [1, 3, 9, 7]:
    print value
```

```
1
3
9
7
```

a) What do you guess would be printed by the following program?

```
for character in 'house':
    print character
print 'done'
```

2. Often the `for` loop iterates over a list generated by the built-in `range` function which has the syntax of: `range([start,] end, [, step])`, where [ ] are used to denote optional parameters. Some examples:

- `range(5)` generates the list `[0, 1, 2, 3, 4]`
- `range(2,7)` generates the list `[2, 3, 4, 5, 6]`
- `range(10,2)` generates the empty list `[]`
- `range(10,2,-1)` generates the list `[10, 9, 8, 7, 6, 5, 4, 3]`

A `for` loop iterates over a list generated by the built-in would look like:

```
for count in range(1,6):
    print count, "  " ,
print "\nDone"
```

```
1  2  3  4  5
Done
```

Write range function calls to generate the following lists:

a) odd numbers from 3 to 13 (inclusive)

b) multiples of 10 from 0 to 100 (inclusive)

c) the numbers from 0 to 100 (inclusive)

3. Since the list generated by the `range` function needs to be stored in memory, a more efficient `xrange` function is typically using in `for` loops to generate each value one at a time for each iteration of the loop. For example:

```
for count in xrange(1,6):
    print count, "  " ,
print "\nDone"
```

```
1  2  3  4  5
Done
```

Thus, the `xrange` version does the same thing as `range` without generating the list.

a) Write a for-loop that counts down from 10 and then prints "BLAST OFF!".

b) Write a program to print even numbers from 0 up to some user specified amount.

c) Complete the following program to average a collection of numeric values:

```
numberOfValues = input(“Enter the number of values to average: ")
total = 0.0
for counter in
```

4. A common operation within a program is to update a variable as:

```
myVariable = myVariable + 1
```

Augmented assignment operations provide a short-hand way to update a variable:

```
myVariable += 1
```

The assignment symbol can be combined with most arithmetic operations and string concatenation.

a) Where in questions (3) could we make use of an augmented assignment operation?

b) If varible `counter` contains an integer, then write an augmented assignment operation to decrement it by 1.