

CS 161 Lab Activity: Iteration Structures

Trials to illustrate behavior of Python **for**.

Note that in python interactive, you must do a blank carriage return after the last indented statement before writing the next statement that is not indented.

```
1) >>> for x in range (10):
...     print x
...
>>>print "After loop: ", x
```

```
2) >>>for y in [3, 7, 9, 12, 4 ]:
...     print y, y*y
```

```
3) >>>for x in range (5):    ← This is really different from other languages
...     x = 10
...     print x
...
```

Some examples of how the **range** function behaves. Type these in and examine the lists they return.

- 1) range (10)
- 2) range (2, 11)
- 3) range (2, 11, 2)
- 4) range (10, 2)
- 5) range (10, 2, -2)

Another way to do counter-controlled iteration, using the **while** structure:

```
1) >>>z = 0
>>>while z < 10:
...     print z
...     z = z + 1
...
>>>print "After loop: ", z

2) >>>z = 0
>>>while z < 10:
...     z = z + 1
...     print z
...
>>>print "After loop: ", z
```

—————> **Note the difference between the output of numbers one and two.**

```
3) >>>z = 0
>>>while z < 3:
...     print "Oh No!"
```

What went wrong?

Since nothing inside the loop changed the value of z, it repeats forever (or until you do a **Control—C** to kill it).