In this lab exercise, you will become familiar with the Python programming language, for which you'll be writing an interpreter in project 3.

To begin, type `python` at the Unix shell prompt — NOT from Scheme! You should see something like this:

```
Python 2.6.2 (r262:71600, Sep 11 2009, 11:35:31)
[GCC 4.4.1] on sunos5
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

The >>> is the Python prompt, like the `STk>` in Scheme. (Later, in some of the examples below, you will see a ". . ." prompt while in the middle of defining a procedure, loop, or if statement.)

Type each of the following lines into Python and note the results. Some will give error messages. If you can't make sense of the result, ask for help. Try to predict what each line will do before running it.

```
2 + 3

2.0 + 3

print 3.6/2

import this

"Hello, World!"

"Hello, World!"[2]

"Hello, World!"[13]

"Hello, World!"[-1]

"Hello, World!"[-20]

print "om" + ("nom"*2)

print "Hello, World!"[1:5]

x = 37
print math.sqrt(x)

p = 1  # I am a comment
q = 1
while p < 100:
  print q
  temp = p
  p = q
  q = p + q

def greet(who):
  print "Hello, " + who

square = lambda x: x*x

range(5,13)

square(6)

(lambda x, y: x**y)(6,4)

def divides(a, b):
  return b != 0 and a % b == 0
divides(123456789, 11)

def isEven(n):
  if divides(n, 2):
    return True
  return False

isEven

isEven(2)

a = 4
while True:
  if isEven(a):
    print a
  if a > 10:
    break
  a = a + 1

def sumDigits(n):
  s = 0
  while n > 0:
    d = n % 10
    s = s + d
    n = n / 10
  return s


range(7)

range(7)[2]
```

```
len("Go west!")

len(range(0,20,2))

nums = map(square, range(5,13,2))

nums

sum = reduce(lambda a,b: a+b, nums)

for n in range(-2,20):
  print n

for letter in "Queen Elizabeth":
  print letter

name = raw_input("What is your name? ")
print "Hello, " + name + "."

def primes_to(n):
  sieve = range(n+1)
  primes = []
  sieve[1] = 0 #mark 1 as not prime
  for i in range(len(sieve)):
    if sieve[i] != 0: # if i hasn't been marked as prime
      primes.append(i) # add it to the list of primes
      for j in range(i*i, len(sieve), i):
        sieve[j] = 0 #mark multiples of i, starting at i^2 as prime
  return primes

primes = primes_to(100)

def isPrime(n):
  if n > primes[-1]:
    return "I don't know"
  else:
    return n in primes

def isPrime2(n):
  if n > primes[-1]:
    return "I don't know"
  elif n not in primes:
    return False
  else:
    return False

for s in [i + j for i in "abc" for j in "def"]:
  print s

nums = [[1, 2, 3], [4,5,6], [7,8,9]] #Python can do deep lists, too!

column2 = [row[1] for row in nums]

print column2
                                        print nums

                                        nums.append(13**2)

                                        nums[-1] = 15**2
```