# MATH 356 Number Theory
# Python Worksheet: $\pi(x)$ and The Euclidean Algorithm

1. We will use Python to calculate $\pi(x)$.

   (a) We first need our prime tester. We will modify it a bit to give us a true/false output.

   ```
   import math
   def isprime(n):
       if n==1:
           prime=False
       if n==2 or n==3:
           prime=True
       for i in range(2,math.floor(math.sqrt(n))+1):
           if n%i==0:
               prime=False
               break
           else:
               prime=True
       return prime
   ```

   Note that we also need to account for $n = 1, 2$, or 3 separately since otherwise the loop is empty (it runs from 2 to 1, so nothing happens!).

   (b) Try your code out on a few integers of your choosing.

   (c) Now write a function in Python to calculate $\pi(x)$, the number of primes less than or equal to $x$. It will require a loop that runs up to $x$ and a counter that keeps track of how many primes you've encountered along the way.

   (d) Python can also loop for as long as a condition remains true (rather than running through a predetermined sequence of numbers). Here is some Python code to calculate $\gcd(a, b)$.

   ```
   def gcd(a,b):
       ri=a
       ri1=b
       ri2=a%b
       while ri2!=0:
           ri2=ri%ri1
           if ri2!=0:
               ri=ri1
               ri1=ri2
       return ri1
   ```

   The `!=` serves as "not equal to" in Python. I have chosen the notation to be reminiscent of how we expressed the Euclidean algorithm: `ri` corresponds to $r_i$, `ri1` corresponds to $r_{i+1}$, and `ri2` corresponds to $r_{i+2}$.

   (e) Use the gcd function to look for a pattern in the gcds of consecutive Fibonacci numbers. You can do this manually or by writing a loop (but I recommend writing a loop for practice!).