**Python Programming Practical 3**

**Due: 22 Feb 2013**

**Instructions**

Submit the following by adding to your git repository cpy5python.git. Your files should be organized as follows:

[cpy5python]

[practical03]

q1\_display\_reverse.py

q2\_display\_pattern.py

q3\_find\_gcd.py

q4\_sum\_series.py

q5\_compute\_series.py

q6\_determine\_prime.py

q7\_display\_matrix.py

q8\_convert\_milliseconds.py

**1 (Displaying an integer reversed)** q1\_display\_reverse.py

Write a function reverse\_int(n) to display an integer in reverse order:

For example, reverse\_int(3456) displays 6543.

**2 (Displaying patterns)** q2\_display\_pattern.py

Write a function display\_pattern(n) to display a pattern as follows:

              1

            2 1

          3 2 1

...

n n-1 ... 3 2 1

**3 (Computing GCD)** q3\_find\_gcd.py

Write a function gcd(m, n) that returns the greatest common divisor between two positive integers:

Write a test program that computes gcd(24, 16) and gcd(255, 25).

**4 (Summing series)** q4\_sum\_series.py

Write a function m\_series(i) to compute the following series:



Write a test program that displays the following table:

i         m(i)

1         0.5000

2         1.1667

...

19        16.4023

20        17.3546

**5 (Computing series)** q5\_compute\_series.py

Write a function to compute the following series:



i    m(i)

1    2.66666666667

3    2.89523809524

...

17   3.08607980112

19   3.09162380667

**6 (Prime number)** q6\_determine\_prime.py

Write a function to determine whether an integer is a prime number. An integer greater than 1 is a prime number if its only divisor is 1 or itself. For example, is\_prime(11) returns True, and is\_prime(9) returns False.

Use the is\_prime(n) function to find the first thousand prime numbers and display every ten prime numbers in a row, as follows:

2   3   5   7   11   13   17   19   23   29

31  37  41  43  47   53   59   61   67   71

73  79  83  89  97  ...

...

**7 (Displaying matrix of 0s and 1s)** q7\_display\_matrix.py

Write a function print\_matrix(n) that displays an n by n matrix, where n is a positive integer entered by the user. Each element is 0 or 1, which is generated randomly. A 3 by 3 matrix may look like this:

0 1 0

0 0 0

1 1 1

**8 (Converting milliseconds to hours, minutes, and seconds)** q8\_convert\_milliseconds.py

Write a method convert\_ms(n) that converts milliseconds to hours, minutes, and seconds. The method returns a string as hours:minutes:seconds. For example, convert\_ms(5500) returns a string 0:0:5, convert\_ms(100000) returns a string 0:1:40, and convert\_ms(555550000) returns a string 154:19:10.