Math 495R Homework 12

(1) Project Euler, problem 12

(2) Recall that for any nonnegative integer $n$, we defined the *factorial of* $n$, or $n!$, to be equal to 1 if $n = 0$ and to be equal to $n \cdot (n-1)!$ if $n > 0$. Thus, the factorial function of $n$ is defined in terms of its own value on integers smaller than $n$. In Python code, we can write this as follows.

```
def factorial(n)
    if n==0:
        return 1
    else:
        return n*factorial(n-1)
```

A function like this defined in terms of itself is called *recursive*. Use a recursive factorial function to solve Project Euler, problem 20.

(3) The Fibonacci sequence $F_n$ is defined for positive integers $n$ by setting $F_1 = 1$ and $F_2 = 1$, and computing $F_n$ for higher values of $n$ by the rule $F_n = F_{n-1} + F_{n-2}$. Thus, $F_3 = F_2 + F_1 = 1 + 1 = 2$, and $F_4 = F_3 + F_2 = 2 + 1 = 3$, and so on. Write a function `fibonacci` that is defined recursively, taking an input of a positive integer $n$ and returning the $n$th Fibonacci number. Use the `time` module (see Python Essentials, chapter 2) to time how long it takes to compute Fibonacci numbers using this function for several integers $n$ of different sizes. Is this faster or slower than the code you wrote to solve Project Euler, problem 2 in Homework 5?