Exercise Problems - 11

Functions - 1

1. Write a recursive function `gcd` that takes two non-negative integers a and b, with $a \geq b$ as input parameters and returns greatest common divisor of a and b, after computing it using the following recursive definition:

$$gcd(a,b) = \begin{cases} gcd(b, a\%b) & : b > 0 \\ a & : b = 0 \end{cases}$$

   Write a program that takes two non-negative integers x and y as input from the user and outputs the greatest common devisor of x and y using the gcd function developed above. Your program should work even if $x < y$. What happens if the order of parameters is changed in the recursive call inside the `gcd` function?

2. This exercise is intended to show you how a badly defined recursive function affects the running time of a program. Let us consider three ways of implementing a function `Comb(n, r)` that represents the number of different ways of choosing r items from a collection of n distinct items, for $0 <= r \leq n$.

   Method 1: The iterative method used in the solution of Question 2 of Quiz 2 (modify to allow $n <= 60$).

   Method 2: Use a recursive definition of the function using the following recursion.

$$Comb(n,r) = \begin{cases} Comb(n, r-1) \times (n-r+1)/r & : r > 0 \\ 1 & : r = 0. \end{cases}$$

   Method 3: Use a recursive definition of the function using the following recursion.

$$Comb(n,r) = \begin{cases} Comb(n-1, r) + Comb(n-1, r-1) & : 0 < r < n \\ 1 & : r = 0, r = n. \end{cases}$$

   Write (three) programs that take n and r as inputs from the user and outputs the value of `Comb(n, r)` computed using each of the methods given above. For values of $n = 60$ and $r = 30$, compare the running time of these programs. How do you explain your observations? If an arithmetic operation is counted as one elementary step and returning a value to a calling function is also counted as one elementary step, how many elementary steps are executed for computing `Comb(60, 30)` in each of the above methods?