Lab 8

Instructions:

Provide your solutions in a file named **lab8.py**. Make sure you run doctest on your solutions before submitting.

Problems:

1. Write a recursive function `printPattern()` that will generate the following patterns. Note that the first argument is always a power of 2 and is the max number of stars appearing in the middle row. The second argument (which defaults to 0) is the number of spaces to put before each patterns rows of stars.

```
>>> printPattern(8,0)
*
**
 *
****
  *
  **
   *
********
    *
    **
     *
    ****
       *
       **
        *

>>> printPattern(4,2)
  *
  **
   *
  ****
    *
    **
     *
>>> printPattern(4)
*
**
 *
****
  *
  **
   *
```

2. Write a recursive function `gcd(m,n)` that returns the greatest common divisor of a pair of numbers. The gcd of m and n is the largest number that divides both m and n. If one of the numbers is 0, then the gcd is the other number. If m is greater than or equal to n, then the gcd of m and n is the same as the gcd of n and m-n. If n is greater than m, then the gcd is the same as the gcd of m and n-m.

```
>>> gcd(5,0)
5
>>> gcd(15,5)
5
>>> gcd(5,7)
1

>>> gcd(24,144)
24
>>> gcd(124,144)
4
>>>
```

3. Write a recursive function `count(lst,target)` that finds the number of occurrences of target in a nested list, where each item in the list is either a number, or, another list (which itself could be listed). The list transversal will be similar to the function `total` written in class. Example:

```
>>> count( [1,2,3,[4,5,5],[[5,2,1],4,5],[3]], 1 )
2
>>> count( [1,2,3,[4,5,5],[[5,2,1],4,5],[3]], 5 )
4
>>> count( [1,2,3,[4,5,5],[[5,2,1],4,5],[3]], 0 )
0
```