Proof By Induction Of My GCD Function

Code:

```
def gcd(x: int, y: int ):
    if y == 0:
        return a
    else:
        return gcd(y, x%y)
```

Theorem:

Given any two non-negative integers, that are not both zero, my function gcd, which is listed above in the code section, will produce the greatest common divisor between the two non-negative integers given to the function

Proof by Induction:

In this proof, I will use x and y to represent the two non-negative integers given to my gcd function that are not both zero, f_{xy} will represent the result of the function, z will be any non-negative number less than $_{fxy}$ and x' and y' will represent my x and y variable through multiple recursions.

Let me first start by stating the definition of the greatest common divisor. I will use g_{xy} to represent the true greatest common divisor of the two non-negative integers given to my function. Since g_{xy} is the true gcd then the following statements are true:

```
x % gxy = 0
y % gxy = 0
```

It would also stand to reason that any other number that x and y are divisible by then g_{xy} would also be divisible by that same number:

```
x % z = 0
y % z = 0
gxy % z = 0
```

Now let me state my two cases in my gcd function. We will have two possible inputs that I need to handle. The first being when y = 0 and the second is when y > 0.

Case 1:

When y = 0 we can see in my code that we will first hit my base case, if b = 0, then terminate the gcd function and return the value in x.

Let's now check this with my definition of the greatest common factor. So, first, we must check does x % x = 0. The answer to this will always be yes because anything divide by itself will have no remainder. Second, since y = 0 anything moded with zero will produce zero. So, y % x = 0 stands true in the case. It also stands to reason then that any non-negative number less than x, z which is also a divisor of x, will fit the second part of my greatest common divisor definition.

Case 2:

When y > 0 we can see in my code that we will now go to the second case. Which will call my gcd method again, but this time we will pass y in as the first variable and x % y for the second.

This will do a few things for us. One if x = y then x%y will return zero and x' will be our since x=y there is not a bigger number that we need to consider. This will work with our definition since anything moded with itself is 0. Second, if x<y then x%y will essentially flip our inputs. For example, if we passed 4 and 20 in, to begin with when we recurse x' = 20 and y'=4 since 4%20=4.

Now if x!=y and x>y then there is an important note to consider for what x%y provides us. If x!=y and x>y then we can assume that:

```
y’ <= x,
y’ < y
```

With this in mind, we know that every time we call gcd again we will get a number that is smaller than both x and y but potentially still not a factor of them. We will keep this process until we get to the first number that is a factor of our subproblems and the factor of our original x and y. Since it is the first we will hit and we are going from the biggest number down it stands to reason it is also the greatest common divisor.

Conclusion:

With all that is stated above, I have shown that for any non-negative x and y values, that are also not both zero, provided to my gcd function you will get a f_{xy} that is the true greatest common divisor of that x and y.