Algorithms - a simple introduction in Python: Part Three

Coding is the new Latin.
Alex Hope (co-author of the NextGen Report)

In this section I am going to look at addition, subtraction, multiplication and division.

Our starting point is some Axioms about natural numbers (positive integers) defined by Giuseppe Peano.

The axioms say that 0 is a natural number and that for every natural number n, its successor is a natural number. In other words, we can get to any integer by starting from zero and adding one until we get there. This may seem obvious, but it's an important idea.

So, here's a program that uses "Peano Arithmetic" to add two integers.

```
# peano_add.py - the algorithm for addition
# using only "-1" and "+1".

def inc(x):
    """Returns x + 1"""
    return x+1

def dec(x):
    """Returns x - 1"""
    return x-1

def peano(x, y):
    """Takes two integers and adds them using Peano addition."""
    if x == 0:
        return y
    else:
        return peano (dec(x), inc(y))

# main
if __name__ == "__main__":
    print("Peano Addition.")
    a = int(input("First number: "))
    b = int(input("Second number: "))
    print(peano(a, b))
```

So, now our computer can add two integers! Woot! We simply keep adding one to x and taking one from y until y is zero. (Another key axiom states that O is not the successor of any natural number).

Subtraction works in a similar way.

```
# peano_minus.py - the algorithm for subtraction
# using only "-1" and "+1".

def peano_minus(x, y):
    """Takes two integers and subtracts second from first."""
    if y == 0:
        return x
    else:
        return peano_minus (x-1, y-1)

# main
if __name__ == "__main__":
    print("Peano Subtraction.")
    a = int(input("First number: "))
    b = int(input("Second number: "))
    print(peano_minus(a, b))
```

For the sake of simplicity, I haven't bothered with inc() and dec() this time. You might notice that this program will happily give us a negative result if we try to subtract a number from a smaller number. You might be happy to live with this, otherwise you could alter the function a bit:

```
    elif x == 0:
        return "Error.\nZero is not the successor of a natural number."
    else:
        return peano_minus (x-1, y-1)
```

Anyway. Let's look at division next.

Division is just repeated subtraction ... really.

```
# peano_div.py - the algorithm for division
# using only "-1" and "+1".

def peano_minus(x, y):
    """Takes two integers and subtracts second from first."""
    if y == 0:
        return x
    else:
        return peano_minus(x-1, y-1)

def divide(x, y, result):
    """Repeatedly subtracts y from x.
    'result' stores the number of subtractions made."""
    # check to see if we are finished
    if x < y:
        if x == 0:
            # return the answer as a list.
            return [result, 0]
        else:
            # this is for when there is a remainder
            return [result, x]
    else:
        # call the subtraction algorithm
        x = peano_minus(x, y)
        # recursively call 'divide', with result incremented by one.
        return divide(x, y, result + 1)

# main
if __name__ == "__main__":
    print("Peano Division.")
    a = int(input("First number: "))
    b = int(input("Divided by: "))
    ans = divide(a, b, 0)
    if ans[1] == 0:
        print(ans[0])
    else:
        print(ans[0], "remainder", ans[1])
```

Division is a little tricky in that you have to take account of the possibility that there will be a remainder. As you can see, the function returns a list, but the remainder is only printed out if is not zero.

Task
Use what you have learned here to write a program that can multiply integers using repeated addition. Don't forget to use doc strings for your functions, and add some comments, to make your code more readable.