Final Problem (Project Euler #116)

Function reds(), greens(), blues()

* Calculates the number of different ways each color can fill the spaces (N)
* Uses the variable: red, green, blue, and N = number of spaces to fill in total red = length of red tile green = length of green tile blue = length of blue tile

**Input:** N, red, green, blue – fix these variables to specific integers, however my functions need no parameters as outside variable are used within the functions.

**Output:** Total number of possible fill ways

**Main Code:** Inside of each function defined for reds, blues, greens.

Initialize N to however many spaces you want the combinations filled for.

**In each function defined**, amount = [1] \* 2 + [0] \* (N - red + 1).

**Amount =** creates the initial list containing 1’s and 0’s in length of the amount of spaces we are trying to fill (N). The 1’s represents the color tiles, while 0’s represent the black tiles which are each of length one.

For ii an in the range of (the color, length of spaces + 1)

Amount[ii] += Uses Fibonacci sequencing to alternate each [ii] in the amount list to adjust the color tile of each specified length throughout the grey tiles to get a number of different possibilities>

Return amount[N] – 1 is the recursive method I have used which takes away 1 length of possibilities each iteration of the for loop as I have used a recursive Fibonacci sequence to determine each round of possibility and gives the amount of possible ways that the spaces can be filled with each specific length of colored tile.

This explanation is the same for each of my created functions except the fact that each function takes into the account of the length of each separate color.

**Print functions**: print out each returned total for each defined function giving a total number of possibilities for colored tile of specific length.

**Note: Please excuse the date at the top of my program as I used and altered my previous Fibonacci sequence class program to create the final project answers.**

**Part 2**

As I have increased the N variable by 1 each iteration (example: N = 5, 6, 7, 8…) there is an apparent pattern for each tile of specified length. There is a pattern for each color scheme which can easily be assumed as this function is almost completely similar in comparison to the Fibonacci sequence function. As a matter of fact, when I had used my original Fibonacci sequence from earlier in the semester with a length of 5 in the sequence, the last number in the sequence each time – 1 would give the total number of possibilities for the red tiles regardless of what variable N was.

It is also obvious that when N is increased, the total number of possibilities increase in drastic changes as this is just another complex use of combinatorics regardless of position.

As noted above, the most important question is how is this related to math and its applications. I would say again this is very similar to and almost identitcal to the Fibonacci sequence, pascals triangle and the golden ration.