Tara Adkins

Final Problem 116

Block Function(m,n)

Input: m = colored blocks to fill whole block,

n = number of whole blocks to be filled

Output: Amount of whole blocks (space size) and the number of ways to fill n with the colored blocks, m

Variable: red = the red blocks with length of 2

green = the green blocks with length of 3

blue = the blue blocks with length of 4

Main Code:

```
differentways = [1] * m + [0] * (n-m+1)
    for j in range(m, n+1):
        differentways[j] += differentways[j - 1] + differentways[j - m]
    return differentways[n] – 1
```

“Differentways” is a list of two lists that are added together. “m” is the type of colored block which is being used to fill in the whole blocks (this is in the first list). The length of m is being multiplied by the list of 1’s. In the second list, the colored blocks “m” are being subtracted by the whole block(s) and then 1 is added onto it.

The first line of the for loop is initiating the whole block in regards to the colored blocks (plus one since python won’t add the empty set).

```
print ("Space size =", n, "units")
print (red)
print (green)
print(blue)
print ("Number of ways to fill:", red+green+blue)
```

I began with printing the number of whole blocks “n” and decided to print each individual variable to see how many red, green, and blue blocks were in n, the total amount of whole blocks. Then my last print statement is the total number of combinations to fill n, with the colored blocks.

One small pattern I saw when changing n from 1-10 was that the blue blocks would slowly increase from 0 then 0, 0, 1, 2, 3, 4, 6, 9, then 13. My thing with this was that between these numbers you are adding n + 1. For example from 3 to 4 that's adding 1, but 4 to 6 you add 2, 6 to 9 you add 3, and 9 to 13 you add 4. If you change n to 11, the blue tiles are 18 which still carries on the pattern of the previous term added plus 1.