Bryant Baus

Euler 116: Red, Green, or Blue Tiles

This program is a def function to compute possibilities of combinations of a m-length tile in an n-length row.

Input: m,r

Output: # of possibilities

possible = [1] + [0]*(n)

This creates a list of length n, with 1 in the 0 index, demonstrating the first instance where there is 1 possibility, a row of black tiles. Therefore in range 1 to n+1:

possible[x] += possible[x-1]

Means the possibilities at the next index will be the same as that before it, so we add in addition to the 1 possible way, the next term becomes the possible ways for a tile of length M to lay in a set of tile of length n, and will remain 1, because for the tiles of lengths 2,3,4, there is still only 1 possibility, where it is all black.

if x >= m:

When the loop reaches a value equivalent or greater to the tile trying to be fitted, the possibilities becomes as many possible ways of the sum of x possibilities and x-m possibilities.

This will continue to alter the values until the value at the index of the length is reached.

For instance, the tile of length 2 in the row of length 5.
Initially there is 1 possibility (all black), then 1 (all black), then 2 (1 black), then 3 (all red, all black, one red), this continues until we reach the length of the row, where we see the possibility of 8 combinations, which excluding the all black one is 7 possibilities.

The function returns the final element, but subtracts 1 to exclude the case where the entire tile row is empty. The sum of the possibilities of red, blue, and green is then returned.

E117 is conducted in a similar manner, but the possibility of N length is determined by the addition of n-2,n-3,and n-4 terms, so the value may be significantly larger. I was unfortunately unable to discover a method to add all of these sums into the value at this index.