1. S(n, k)

{

if(k == 1 OR k == n) return 1;

else return k × S(n - 1, k) + S(n - 1, k - 1)

}

2. I would use a 2D-array since this algorithm requires two parameters to run -- n and k. Consequently, the result of any execution of this algorithm will rely on these two arguments.

3. dyn(n, k)

{

arr = array[n][k] // Initialize 2D-array

for(i = 0; i < n; i++) // For every column

{

for(j = 0; j < k; j++) // For every row

{

arr[i][j] = 0 // Set the cell to zero

}

}

}

S(n, k)

{

if(arr[n][k] != 0) return arr[n][k] // Return known result

if(k == 1 OR k == n) // Base Case

{

arr[n][k] = 1 // Result is 1

}

else

{

arr[n][k] = k × S(n - 1, k) + S(n - 1, k - 1)

}

return arr[n][k] // Return result

}