Formula Design

This program takes a positive integer and prints out the expanded form of the expression (1 + x)^n. So, if the input is n then the output of the program is:

(1 + x)^n = 1 + nC1*x^1 + nC2*x^2+ ... + nCr*x^r + ... + nCn*x^n

where nCr = the coefficient calculated by the Factorial and nCr functions. The formula for nCr is:

n \ r!(n - r)!

Implementation of Factorial

To calculate the factorials of the equation used a slightly optimized algorithm to calculate factorial compared to the definition n! = n * (n - 1) * (n - 2) * .... * 2 * 1

In order to split the number of multiplications I took advantage of the fact that for even numbers

$8! = 8 * (8 + 6 = 14) * (14 + 4 = 18) * (18 + 2 = 20)$

$8! = 8 * 14 * 18 * 20$

and for odd numbers

$9! = 9 * (9 + 7 = 16) * (16 + 5 = 21) * (21 + 3 = 24) * (\text{roundUp}(9/2) = 5)$

$9! = 9 * 16 * 21 * 24 * 5$

So the runtime of the Factorial function is n/2 so O(n).

Implementation of nCr

The nCr function just returns the division of the numerator and denominator found by passing the args n, r, and n - r to Factorial. So the runtime of nCr 3 * (n / 2) which is still O(n).

Runtime of Formula

nCr is called n times in the main function so the overall runtime is O(n^2)

Space Usage

Memory use is constant for most of the program except for the array that holds the coefficients calculated by nCr which is then printed out. So space usage is O(n).

Limitations

I wrote my nCr and Factorial functions in 32 bit assembly instructions so the maximum factorial value that can be calculated is 12! before overflow conditions are met. So the maximum value for which the program will work is 12. Any other integer will result in the program returning 0 and exiting.