ReadMe
#formula

Formula.C is a program that reads in an integer value and outputs the extended form of (1+x)^N where N is the input. Additionally, Formula.C works with a –h help tag.

While this would be a simple project to code in plain C, the two functions, factorial and nCr were required to be coded in assembly, and a linker nCr.h was provided.

The majority of the Formula.C code is simply a loop that makes calls to the linked nCr in order to compute the correct values and output the extended formula.

nCr.s is the file containing the assembly code.
nCr.s contains two functions, factorial, and nCr.

Factorial:

Factorial is designed to calculate the factorial value of a given integer.
In C, the %rax register is used as the output and the %rdi register is used as the input.

Using this knowledge, Factorial fills %rax with 1, then creates a loop that utilizes mul %rdi in order to constantly multiply and assign the value into %rdi, then decrementing %rdi. This loop continues until %rdi is 0, then returns the value inside %rdi.

nCr utilizes call factorial, but involves much more memory management.
Since there are a limited number of registers to use, holding on to required values and keeping proper registers open was difficult, but with a few push-pops to the stack, the method to solving nCr was rather simple.

Since factorial changes the given %rdi and %rsi, both had to be pushed to the stack before each call of factorial.

Then, using %rbx, %rcx, and %rdx as temporary holders, I was able to compute n!, n-r!, and r! and have them properly stored. Finally, I had to be sure to that %rdx was empty, because the div instruction uses %rdx and if I used it for a swap, that could throw off the entire calculation.

In the end, the biggest challenges with NCR had to do with calls to factorial, and how certain calls affected unexpected registers. It took a lot of work to find out why my %rdx or %rbi were randomly changing when I didn't explicitly access them in factorial. However, using gdb and the i r function, I was able to view the registers and catch what was changing them.