Computer Architecture Summer 2018
Masood Karimi
Professor Brian Russell

Synopsis:

Calc is a program that takes in four arguments on start up and conducts an addition, subtraction or multiplication calculation using those arguments. The first argument is the operation being conducted ("*", "+", "-"). The second argument is the first integer for the calculation and begins with an identifier ("d", "x", "o", "b") to determine how we should interpret the following value (ex. "d12345" represents the number 12345 in decimal). The third argument is the second integer for the calculation and follows the same format as the second argument. The final argument is the identifier for how we should output the result of the calculation ("d","b","x","o").

Format is a program that takes in two arguments and returns a formatted integer or float based upon those received arguments. The first argument is a 32-bit sequence representing either an integer or a float. This is determined by the second argument which will either be "int" or "float".

Calc.c Implementation and Design:

This file was the only file used in the creation of the **Calc** program. It follows a simple and easy to follow design. On startup, the program begins by checking for any errors and reporting them via the printf command. If no errors are found the program continues by converting the second and third argument into decimal form (if it isn't already). One the two values are in decimal form, there is a function that conducts the calculation based upon the operation provided in the first argument. Once the calculation is complete, the program takes the final value and converts it into appropriate integer format (octal, hexadecimal, or binary) and prints the result.

Overall Analysis:

The overall run time, or Big-O, of this program would be some arbitrary value based upon the computer's specs and the given input values. There is no algorithm that consistently loops for "n" times, but rather set constants. This makes the program very efficient and have essentially an O(1) run time.