# README

HW 8: asmcoding

By: Arnav Kothari (akotha02) and Frank Ma (jma07)

Date: 9/12/19

## Acknowledgements

We would like to thank TA Danielle and Byron for helping us structure our print function.

## Completion

From our knowledge and attention to the reference implementation, everything has been correctly implemented.

## Deviation from calling conventions

We followed some calling conventions recommended in the spec:

- r0 always holds 0
- r6 and r7 always used as temp registers

However, we made a few changes:

- r1 holds the return address of the functions main and printd. However, in the functions that perform arithmetic and bitwise operations, it is used to store the result of a calculation, which is then pushed onto the value stack. This is similar in concept to a "return value", but is not really comparable since that value is stored onto the value stack rather than used by the caller.
- r3 is designated to be the stack pointer of the value stack. This is consistent throughout all modules.
- Arguments to functions are passed by register r4, rather than on the call stack. This is because for this specific program, no function calls require more than 1 argument (all the arithmetic and bitwise operations that requires two inputs get their inputs from the value stack). As such, we designated r4 as the register to be used whenever a value needs to be passed from one function to another. Doing this allows us to perform less operations and allows the program to be more efficient. We are aware that this calling method would be problematic for functions with more than 1 argument as we would run out of available registers, but we believe that it is a valid tradeoff for the purpose of the RPN program.
- r5 is sometimes used as a temporary register if one is needed, such as for if statements.
- In all other cases, r4 and r5 are used as general purpose registers.

## Print module

We chose to implement the print module by using a stack onto which we push the digit(s) we are trying to print (in the reverse order), then popping each digit from this digit stack and printing them out in the correct order.

## Hours spent

Hours spent analyzing the assignment:

~2 hours

Hours spent writing assembly code:

~9 hours

Hours spent debugging your calculator:

~1 hour