ECE220 Computer Systems and Programming

Lab 2A

1 After this week's lectures, you should be able to...

1. Understand how TRAPs are invoked in LC-3 and use basic TRAPs such as GETC, OUT, etc.
2. Differentiate between callee-save and caller-save, and apply these concepts when writing or using subroutines.
3. Explain how the stack ADT works both conceptually(LIFO) and in memory.
4. Implement stack operations(PUSH & POP) in LC-3.

2 After today's lab, you should be able to...

1. Implement subroutines and invoke them by using JSR(or JSRR)and RET.
2. Use PUSH and POP operations in your MP and lab, and check for underflow.
3. Recognize the similarity between the lab exercise and the MP, and quickly get started with implementing the MP.
4. Use unit tests to check for correctness of subroutines in your code before piecing them together.

3 Exercises

1. Given the PUSH and POP subroutines, and labels STACK_START and STACK_TOP, describe how to check whether the stack is empty or not.
   Load STACK_START and STACK_TOP to register, sub two register using NOT and ADD 1, if the result is zero, then stack is empty, else if the result > 0, then stack is not empty, else stack is underflow
2. After reading through the write-up for MP2, you assess that the arithmetic operation subroutines will be easier to implement than the evaluate subroutine. Therefore, you want to start with the 'PLUS' subroutine first. How would you test the correctness of your subroutine without EVALUATE? Write a few lines of code to demonstrate.

```
.ORIGIN x3000 ;
;;;;;;;;;;;;;;;
; inputs: R3, R4
; output: R0
PLUS ; Assume already implemented
RET
;;;;;;;;;;;;;;;
; inputs: R0
; outputs: hexadecimal string to terminal
PRINT_HEX ;Assume already implemented
RET
;;;;;;;;;;;;;;;;
```

```
; you test code starts here
LD R3, TEST_VAL1
LD R4, TEST_VAL2
JSR PLUS

HALT
.END

TEST_VAL1 .FILL #5
TEST_VAL2 .FILL #6
```