Worksheet 3 - TRAPs & Subroutines

Exercise 1:

Given a string in memory, write an LC3 program that converts lowercase to uppercase. If the character is already uppercase or non-alphabetic, do nothing on the character. The starting address is pointed by R0.

Example

input string: Ece220

output string: ECE220

```
.ORIG       x3000
        LEA     R0, MyStr
        JSR     L2U
        HALT
;; L2U: convert lowercase to uppercase
;input: R0 (starting addr of string)
;output: none
L2U:
        ST      R0, SAVER0_L2U
        ST      R1, SAVER1_L2U
        ST      R2, SAVER2_L2U
LOOP
        (1)_____________________    ;load the character pointed by R0
                                    ;to R1
        (2)BR__    NULL             ;check null-char
        LD      R2, NEG_a           ;load NEG_a
        (3)_____________________    ;check char is greater than or
equal
                                    ;to 'a'
        (4)BR__    NEXTCHAR         ;if not, do nothing
        LD      R2, NEG_z           ;load NEG_z
        (5)_____________________    ;check char is less than or
                                    ;equal to 'z'
        (6)BR__    NEXTCHAR         ;if not, do nothing
        ; lower case here
        LD      R2, DIFF_L2U
        (7)_____________________    ;convert lower to upper
        (8)_____________________    ;store the new char in the memory
NEXTCHAR
        (9)_____________________    ;increase pointer
        BRnzp   LOOP
```

```
NULL
        LD      R0, SAVER0_L2U
        LD      R1, SAVER1_L2U
        LD      R2, SAVER2_L2U
        RET
MyStr      .STRINGZ        "Ece220"
NEG_a      .FILL   #-97          ;additive inverse of ASCII 'a'
NEG_z      .FILL   #-122         ;additive inverse of ASCII 'z'
DIFF_L2U   .FILL   #-32          ;difference between uppercase and
                                 ;lowercase
SAVER0_L2U  .BLKW   #1
SAVER1_L2U  .BLKW   #1
SAVER2_L2U  .BLKW   #1
.END
```

Exercise 2:

Write an LC3 program that reverses the contents of a sequence of memory. The starting and ending address is pointed by R0 and R1.

- Use SWAP subroutine to swap the contents of two memory locations.
- *Example*

Address	Value (before)	Value (after)
x4000 (starting addr)	x0	x3
x4001	x1	x2
x4002	x2	x1
x4003 (ending addr)	x3	x0

```
        .ORIG       x3000
        LD      R0, ADDR_A
        LD      R1, ADDR_B
        JSR     REVERSE
        HALT

;;REVERSE: reverse the contents of a sequence of memory
;input: R0 (starting address), R1 (ending address)
;output: none
REVERSE:
        ;callee-save (determine which registers to save)
        ;upto 4 registers
        _____________________
        _____________________
        _____________________
        _____________________
LOOP
        _____________________     ;call SWAP subroutine
        ADD     R0, R0, #1        ;increment starting address
        ADD     R1, R1, #-1       ;increment ending address
        ;compare starting and ending address
        _____________________
        _____________________
        _____________________
        BR__     LOOP
        ;recover registers
        _____________________
        _____________________
        _____________________
        _____________________
        RET

;; SWAP: swap the contents of memory pointed by R0 and R1
;input: R0, R1
;output: none
SWAP:
        ST      R2, SAVER2_SWAP
        ST      R3, SAVER3_SWAP
        LDR     R2, R0, #0        ;R2<-mem[R0]
        LDR     R3, R1, #0        ;R3<-mem[R1]
        STR     R3, R0, #0        ;R3->mem[R0]
```

```
        STR     R2, R1, #0      ;R2->mem[R1]
        LD      R2, SAVER2_SWAP
        LD      R3, SAVER3_SWAP
        RET

ADDR_A          .FILL   x4000
ADDR_B          .FILL   x4009
;You don't need to use all of them
SAVER0_REV     .BLKW   #1
SAVER1_REV     .BLKW   #1
SAVER2_REV     .BLKW   #1
SAVER3_REV     .BLKW   #1
SAVER4_REV     .BLKW   #1
SAVER5_REV     .BLKW   #1
SAVER6_REV     .BLKW   #1
SAVER7_REV     .BLKW   #1
SAVER2_SWAP     .BLKW   #1
SAVER3_SWAP     .BLKW   #1
        .END
```