Joseph Godinez

Professor Oscar Ho

Computer Science 240-20

November 18, 2021

```
DECRYPT
            NOT R2, R2                  ; Makes it -R2
            ADD R2, R2, #1              ; Turn it into 2s compliment
DECRYPTIONLOOP
            LDR R0, R3, 0
            BRz SKIP
            ADD R0, R0, R2              ; Subtract R0-R2
            ADD R4, R4, #1;;;;R4 === #1
            AND R4, R4, R0
            BRp One_2
            ADD R4, R0, #1
            BRnzp D_Store
One_2       ADD R4, R0, #-1
D_Store
            STR R4, R5, #0
            AND R4, R4, #0;;;;;;R4 === R0 #64
            ADD R3, R3, #1
            ADD R5, R5, #1
            BRnzp DECRYPTIONLOOP
```

```
;                                       ;
EncryptLoop
            LDR R0, R3, 0               ; Load the value in the memory location pointed by R3 with 0 Offeset into R0
            BRz SKIP                    ; Ends Encryption when all characters are read
            AND R4, R0, #1              ; Determines last bit in R0. Use AND with decimal number #1
            BRp One_1                   ;
            ADD R4, R0, #1              ; If last bit in R0 is zero, adds 1 to toggle bit, stores in R4
            BRnzp E_Store
One_1       ADD R4, R0, #-1             ; If last bit is one, subtracts 1 to toggle bit, stores in R4

E_Store
            ADD R4, R4, R2              ; Adds encyption key to R4 (char w/ toggled bit)
            STR R4, R5, #0              ; Stores/write encrypyed char into its memory location "NEW'
            ADD R3, R3, #1              ; Increments character pointer to next memory address for loop "EncryptLoop"
            ADD R5, R5, #1              ; Increments memory pointer to next memory address for storing "NEW"
            BRnzp EncryptLoop
```

```
InputLoop   GETC                        ; Begin of InputLoop, gets characterinput
            OUT                         ; Write one character to console same as TRAP 21
            ADD R4, R0, R3              ; Compares input character in R0 with R3
            BRz ExitInput               ; Exits loop if "ENTER" key is detected
            STR R0, R1, #0              ; Stores input character in R0 into memory location in R1
            ADD R1, R1, #1              ; Increments R1 memory location for next character in input encrypt message
            ADD R2, R2, #1              ; Increments counter
            BRn InputLoop               ; Loops if counter is still negative
ExitInput                               ;

            AND R3, R3, #0              ; Clears R3 for later use
            AND R4, R4, #0              ; Clears R4 for later use

            ;These lines prepare for encryption/decryption
            LD R1, STORE                ; Stores x3100 in R1
            LDR R2, R1, #1              ; Stores encryption key (ASCII) in R2 from memory address in R1 + 1
            ADD R2, R2, #-16            ; These three lines convert R2 from ASCII to Decimal. "0-9" are "48-57" ASCII code
            ADD R2, R2, #-16            ; ^                                                                          ;Line 50
            ADD R2, R2, #-16            ; ^^
            ADD R3, R1, 2               ; Makes R3 point to first char in input messge for encryption from the memory location in R1
            LD R5, NEW                  ; Loads memory location NEW to store encrypted message

            LDI R1, STORE               ; Reloads R1 with x3100
            LD R6, N68                  ; Loads R6 with -68 to check for D input
            ADD R1, R6, R1              ; Adds R1 and R6 to check if D was first input
            BRz DECRYPT                 ; Goes to DECRYPT if D was first input, else it runs encyption in next instruction
.
```