```
@ Ryan Bentz
@ ECE 371
@ Programming Assignment 1 - Procedural Verstion
@ This program stores 16 bytes that represent values from the ADC and conditions
@ them based on a given condition factor
@ This program takes the 16 conditioned values and calculates the rounded average
@ for those values
@ 10/30/2017



.text
.global _start

_start:
.equ    COUNT, 16       @ define the counter

@ set the paratemers to pass to the procedure and call the conditioning procedure
LDR R0, =Fahrenheit_Rough       @ load the address for the rough values array
LDR R1, =Fahrenheit_True        @ load the address for the true values array
MOV R2, #COUNT                  @ set the counter
BL PROC_AVERAGE                 @ Call the procedure to correct the rough values
                                @ and calculate the average

NOP
NOP

B END   @ go to the end of the program

@ Procedure to determine the correction factor and condition the rough values
@ Corrects the ADC Rough values and returns the average of the corrected values
@ Inputs:   Pointer to rough values array
@           Pointer to true values array
PROC_AVERAGE:

    STMFD R13!, {R4-R8, R14}    @ save the register states and link register location
    MOV R4, R1                  @ move the address of the true values array
                                @ to maintain orginal pointer

    CORRECTION:

        LDRH R6, [R0], #2   @ load the value from the rough array
                            @ and post-index increment the pointer

        CMP R6, #20         @ Compare roughval to the upper limit to the tier
        MOV R7, #0          @ set the conditioning factor
        BLS MATCH           @ Branch to end of if-else structure if less than upper limit

        CMP R6, #39         @ Compare roughval to the upper limit to the tier
        MOV R7, #1          @ Add the conditioning factor
        BLS MATCH           @ Branch to end of if-else structure if less than upper limit

        CMP R6, #59         @ Compare roughval to the upper limit to the tier
        MOV R7, #3          @ Add the conditioning factor
        BLS MATCH           @ Branch to end of if-else structure if less than upper limit

        CMP R6, #79         @ Compare roughval to the upper limit to the tier
        MOV R7, #7          @ Add the conditioning factor
        BLS MATCH           @ Branch to end of if-else structure if less than upper limit

        CMP R6, #99         @ Compare roughval to the upper limit to the tier
        MOV R7, #12         @ Add the conditioning factor
        BLS MATCH           @ Branch to end of if-else structure if less than upper limit

        CMP R6, #120        @ Compare roughval to the upper limit to the tier
        MOV R7, #20         @ Add the conditioning factor
        BLS MATCH           @ Branch to end of if-else structure if less than upper limit

    MATCH:
        ADD R6, R6, R7          @ Add the conditioning factor to the current rough value
        SUBS R2, #1             @ decrement the counter
        STRH R6, [R4], #2       @ store the conditioned true value
        BNE CORRECTION          @ check if we are at the end of the loop

    @ Average the corrected values
    LDR R0, =AVERAGEVAL     @ store the average val pointer
    MOV R2, #COUNT          @ reset the count
    MOV R4, #0              @ clear the sum register


    AVERAGE:
        LDRH R5, [R1], #2       @ load the value from the array and
                                @post-index increment the pointer
        ADD R4, R4, R5          @ add the true value to the sum and
                                @ store it in the same place as the sum
        SUBS R2, #1             @ decrement the counter
        BNE AVERAGE             @ check if we are at the end of the loop

    MOVS R4, R4, LSR #4     @ divide by 16
    ADC R4, R4, #0          @ add contents of carry flag
    STRH R4, [R0]           @ write the average val to memory

    LDMFD R13!, {R4-R8, PC}


END:

.data
@ Define the data structures as arrays
Fahrenheit_Rough:   .HWORD  0x0F, 0x1E, 0x32, 0x46, 0x5A, 0x6E, 0x00, 0x14, 0x15, 0x27, 0x28, 0x3B, 0x3C, 0x4F, 0x50, 0x63
Fahrenheit_True:    .HWORD  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
AVERAGEVAL: .HWORD  0x00
STACK:      .rept 128
            .byte 0x00
            .endr

.END
```