CS 2208 Assignment 4 Tyler Browne 250814405

Question One Code:

```
            AREA Assignment4_Q1, CODE, READWRITE
            ENTRY

            LDR r0, =STRING1    ; load string1 into register 0
            MOV r1, #0          ; move 0 into register 1
            LDR r2, =STRING2    ; load string2 (empty) into register 2
            LDR r4, =EoS        ; load EoS (end of string/null character) into register 4
            MOV r5, #0          ; move 0 into register 5
            MOV r7, #0          ; move 0 into register 7
            B FIRST                     ; jump to first (skips the check for a space)

LOOP        LDRB r3, [r0, r1]   ; loads r0 (string1) in position r1 into r3
            CMP r3, #0x20               ; compares r3 to the ASCII character space (0x20)
            BNE NEXT            ; if r3 is not a space then the program jumps to NEXT

            ADD r1, r1, #1      ; increments r1 by 1

FIRST       LDRB r3, [r0, r1]   ; loads r0 (string1) in position r1 into r3
            CMP r3, #0x74               ; compares r3 to the ASCII character "t" (0x74)
            BNE NEXT            ; if r3 is not a "t" then the program jumps to NEXT

            ADD r1, r1, #1

            LDRB r3, [r0, r1]   ; loads r0 (string1) in position r1 into r3
            CMP r3, #0x68               ; compares r3 to the ASCII character "h" (0x68)
            BNE NEXT            ; if r3 is not a "h" then the program jumps to NEXT

            ADD r1, r1, #1      ; increments r1 by 1

            LDRB r3, [r0, r1]   ; loads r0 (string1) in position r1 into r3
            CMP r3, #0x65               ; compares r3 to the ASCII character "e" (0x65)
            BNE NEXT            ; if r3 is not an "e" then the program jumps to NEXT

            ADD r1, r1, #1      ; increments r1 by 1

            LDRB r3, [r0, r1]   ; loads r0 (string1) in position r1 into r3
            CMP r3, #0x20               ; compares r3 to the ASCII character space (0x00)
            BEQ UPDATE          ; if r3 is a space, then jump to UPDATE

            CMP r3, r4          ; compares r3 to r4, which is EoS (0x00)
            BNE NEXT            ; if r3 is not the end of string, then jump to NEXT
            BEQ DONE            ; if r3 is equal to the end of string, then jump to DONE

UPDATE              ADD r5, r5, r1      ; add r1 to r5, and store in r5
            MOV r1, r5          ; move the value in r5 into r1
            B LOOP                      ; jump to LOOP
```

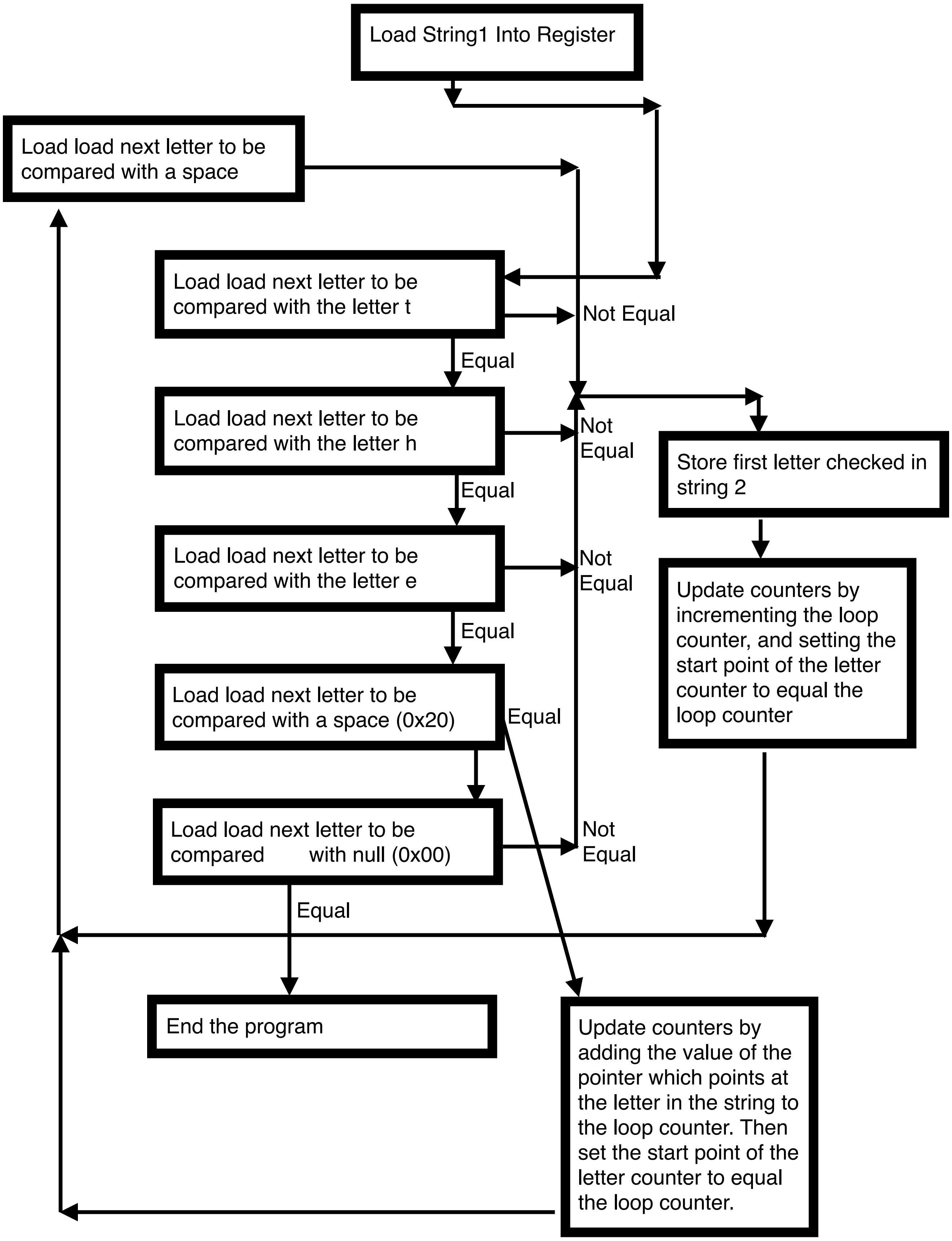
```
NEXT        LDRB r6, [r0, r5]       ; loads r0 (string1) in position r5 into r6
            STR r6, [r2, r7]        ; store the value r6 into r2 at the position specified by r7
            ADD r5, r5, #1          ; increment r5 by 1
            ADD r7, r7, #1          ; increment r7 by 1
            B LOOP                          ; jump to LOOP

DONE                                ; end of program

            AREA Assignment4_Q1, DATA, READWRITE

STRING1     DCB "and the man said they must go"
EoS         DCB 0x00
STRING2     space 0xFF

            END
```

Question Two Code:

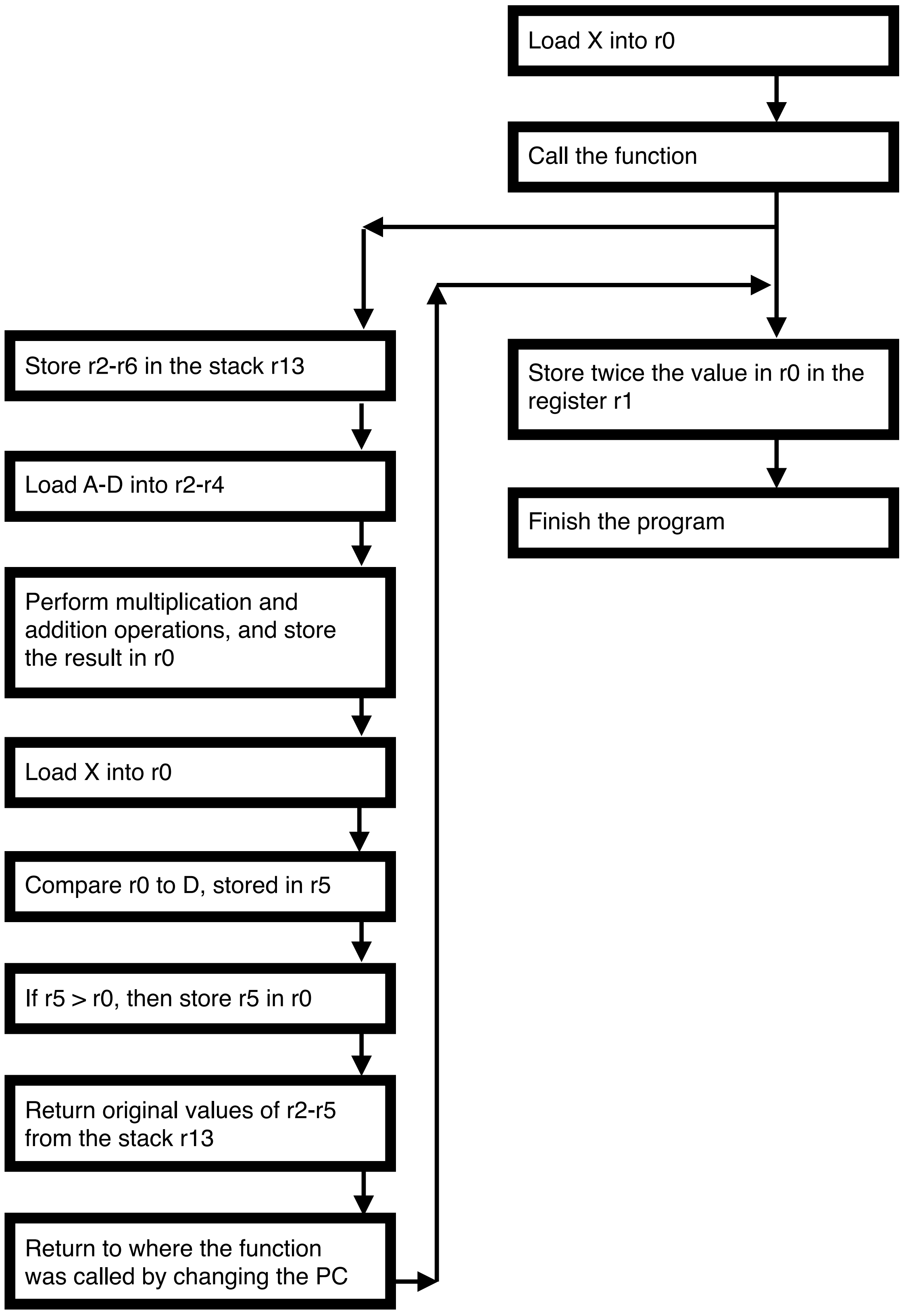
```
                AREA Assignment4_Q2, CODE, READWRITE
                ENTRY

                LDR r0, =X              ; load the value in X into register 0
                BL FUNC                         ; call the function
                MOV r1, r0, LSL #1      ; move two times the number in r0 into r1
LOOP            B LOOP                          ; infinite loop end to the program

;-------------------------------------------
;-------------------------------------------

FUNC            STM r13!, {r2-r6}       ; store registers r2-r6 in the stack r13
                LDR r2, =A              ; load the value A into register 2
                LDR r3, =B              ; load the value B into register 3
                LDR r4, =C              ; load the value C into register 4
                LDR r5, =D              ; load the value D into register 5
                MUL r6, r0, r0          ; multiply r0 by itself and store in r6
                MUL r4, r2, r6          ; multiply r2 by r6 and store it in r4
                MLA r2, r3, r0, r4      ; multiply r3 by r0 and add r4 to it
                ADD r0, r4, r2          ; add r2 to r4 and store it in r0
                CMP r0, r5              ; compare r0 to r5 (D)
                MOVGT r0, r5            ; if r5 is greater than r0, then store r5 in r0
                LDM r13!, {r2-r6}       ; restore the original values of r2-r6 from r13
                MOV r15, r14            ; return the function to the calling location
                                        ;       via modifying the Program Counter

;-------------------------------------------
                AREA Assignment4_Q2, DATA, READWRITE

A               DCD 5
B               DCD 6
C               DCD 7
D               DCD 90
X               DCD 3

                END
```

CS 2208 Assignment 4 Tyler Browne 250814405

Question 1 Flowchart:


Load String1 Into Register
Load load next letter to be compared with a space
Load load next letter to be compared with the letter t
Not Equal
Equal
Load load next letter to be compared with the letter h
Not Equal
Equal
Store first letter checked in string 2
Load load next letter to be compared with the letter e
Not Equal
Update counters by incrementing the loop counter, and setting the start point of the letter counter to equal the loop counter
Equal
Load load next letter to be compared with a space (0x20)
Equal
Load load next letter to be compared with null (0x00)
Not Equal
Equal
End the program
Update counters by adding the value of the pointer which points at the letter in the string to the loop counter. Then set the start point of the letter counter to equal the loop counter.

Question 2 Flowchart:


Load X into r0
Call the function
Store r2-r6 in the stack r13
Load A-D into r2-r4
Perform multiplication and addition operations, and store the result in r0
Load X into r0
Compare r0 to D, stored in r5
If r5 > r0, then store r5 in r0
Return original values of r2-r5 from the stack r13
Return to where the function was called by changing the PC
Store twice the value in r0 in the register r1
Finish the program