**Question 1**NOTE: Memory map will need read and write permissions for location of STRING3

Code:

AREA question1, CODE, READONLY  
ENTRY  
ADR r1, STRING1 ; get address of STRING1  
ADR r2, STRING2 ; get address of STRING2  
ADR r3, STRING3 ; get address of STRING3

;store STRING1 into STRING3 byte by byte  
first LDRB r0, [r1], #1 ; get byte of STRING1 then post-increment pointer STRB r0, [r3], #1 ; store byte from STRING1 in empty byte in STRING3 and post-increment pointer CMP r0, #0 ; check if the EoS1 byte has been reached BNE first ; if EoS1 byte not reached then STRING1 has not been copied so repeat loop to get next byte of STRING1

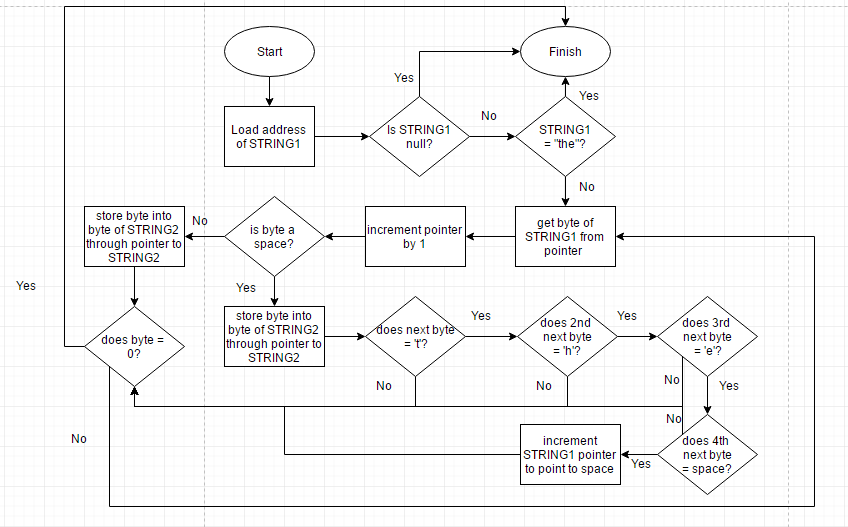
;store STRING2 into STRING3 byte by byte  
second LDRB r0, [r2], #1 ; get byte of STRING2 then post-increment pointer STRB r0, [r3], #1 ; store byte from STRING2 in empty byte in STRING3 and post-increment pointer CMP r0, #0 ; check if the EoS2 byte has been reached BNE second ; if EoS2 byte not reached then STRING2 has not been copied so repeat loop to get next byte of STRING1

loop b loop

AREA question1, DATA, READWRITE  
STRING1 DCB "This is a test string1" ;String1  
EoS1 DCB 0x00 ;end of string1  
STRING2 DCB "This is a test string2" ;String2  
EoS2 DCB 0x00 ;end of string2  
STRING3 space 0xFF END

**Question 2**NOTE: Memory map will need read and write permissions for location of STRING2

Flowchart:



Code:

AREA question2, CODE, READONLY  
ENTRY  
ADR r1, STRING1 ; get address of STRING1  
ADR r2, STRING2 ; get address of STRING2

;check if first word is 'the'   
;(the 'store' loop checks for 'the' that starts with a space, if sentence starts with 'the' then it does not start with a space so this is a special case)  
LDRB r5, [r1] ; get first byte of STRING1   
CMP r5, #0 ; check if string is empty  
BEQ loop ; if empty the no processing needed  
CMPNE r5, #0x74 ; check if byte is 't' if string is not empty  
BNE store ; if not 't' then continue to 'store' loop  
LDRB r5, [r1, #1] ; get second byte  
CMP r5, #0x68 ; check if byte is 'h'  
BNE store ; if not 'h' then continue to 'store' loop  
LDRB r5, [r1, #2] ; get third byte  
CMP r5, #0x65 ; check if byte is 'e'  
BNE store ; if not 'e' then continue to 'store' loop  
LDRB r5, [r1, #3] ; get fourth byte  
CMP r5, #0 ; check if EoS  
BEQ loop ; if byte is EoS then STRING1 == "the" and processing of string is done  
CMPNE r5, #0x20 ; check if byte is space  
BNE store ; if not space then continue to 'store' loop  
ADD r1, r1, #3 ; skip 'the' if word has been found by making pointer point to the space byte after 'the'

store LDRB r0, [r1], #1 ; get byte of STRING1 then post-increment pointer CMP r0, #0x20 ; check if the byte is a space BEQ the ; if byte is a space start checking for 'the' STRB r0, [r2], #1 ; fall through to store byte from STRING1 in empty byte in STRING2 and post-increment pointer b cont ; skip past 'the' branch to continue on

the STRB r0, [r2], #1 ; store space byte in STRING2 LDRB r3, [r1] ; get byte after space byte without post/pre increment CMP r3, #0x74 ; check if byte is 't' LDRBEQ r3, [r1, #1] ; if previous byte is 't' get next byte CMPEQ r3, #0x68 ; check if byte is 'h' LDRBEQ r3, [r1, #2] ; if previous byte is 'h' get next byte CMPEQ r3, #0x65 ; check if byte is 'e' LDRBEQ r3, [r1, #3] ; if previous byte is 'e' get next byte CMPEQ r3, #0x20 ; check if byte is a space ADDEQ r1, r1, #3 ; skip past 'the' if word is found BEQ cont ; continue on after updating pointer skip 'the' CMPNE r3, #0 ; if byte is not a space check if it is EoS BEQ loop ; if byte is EoS then string is processed  
  
cont CMP r0, #0 ; check if the EoS1 byte has been reached BNE store ; if EoS1 byte not reached then STRING1 has not been copied so repeat loop to get next byte of STRING1

loop b loop

AREA question2, DATA, READWRITE  
STRING1 DCB "the the the 123 the" ;String1  
EoS DCB 0x00 ;end of string1  
STRING2 space 0xFF END

**Question 3**

NOTE: Memory map will need read and write permissions for location of reg1, reg2, reg3

Code:

AREA question3, CODE, READONLY  
ENTRY   
MOV r0, #2\_11 ; binary value in r0, denoted by x  
BL funcY ; go to subroutine  
MOV r1, r0, LSL#1 ; store doubled value of r0 in r1

loop b loop

funcY STR r1, reg1 ; store original value of r1 in reg1 STR r2, reg2 ; store original value of r2 in reg2 STR r3, reg3 ; store original value of r3 in reg3 MUL r1, r0, r0 ; get x^2 LDR r2, a ; get a LDR r3, c ; get c MLA r3, r1, r2, r3 ; get ax^2 + c LDR r2, b ; get b MLA r0, r2, r0, r3 ; get bx + ax^2 + c LDR r1, d ; get d CMP r0, r1 ; check if r0 > d MOVGT r0, r1 ; clip y to d if y > d LDR r1, reg1 ; restore r1 LDR r2, reg2 ; restore r2 LDR r3, reg3 ; restore r3 BX lr ; exit function

AREA question3, DATA, READWRITE   
a DCD 5   
b DCD 6  
c DCD 7  
d DCD 50  
reg1 space 0x4 ; stores original value of r1  
reg2 space 0x4 ; stores original value of r2  
reg3 space 0x4 ; stores original value of r3 END