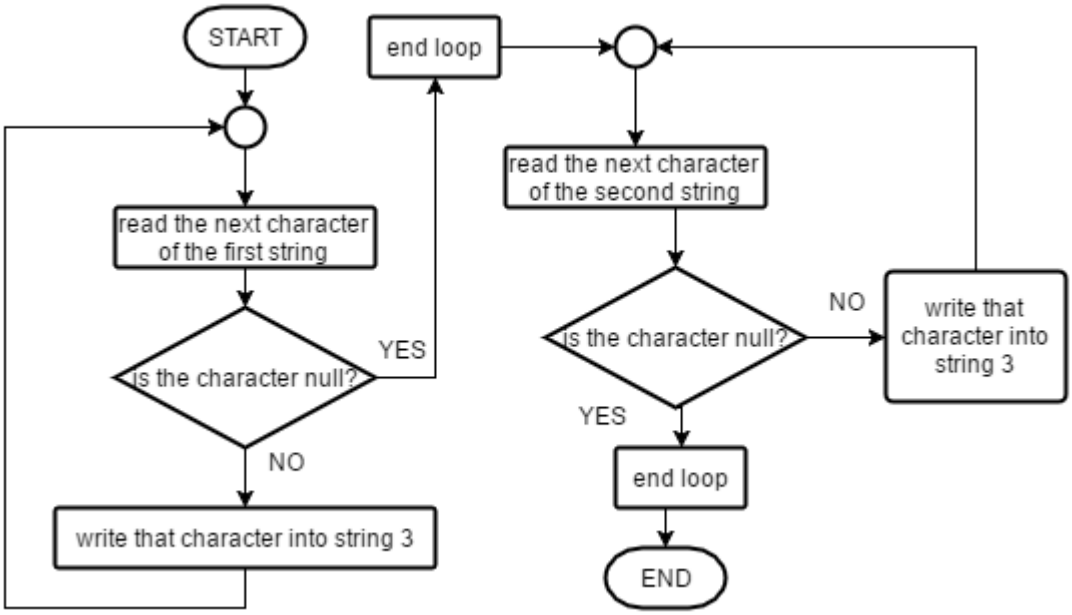
Question 1

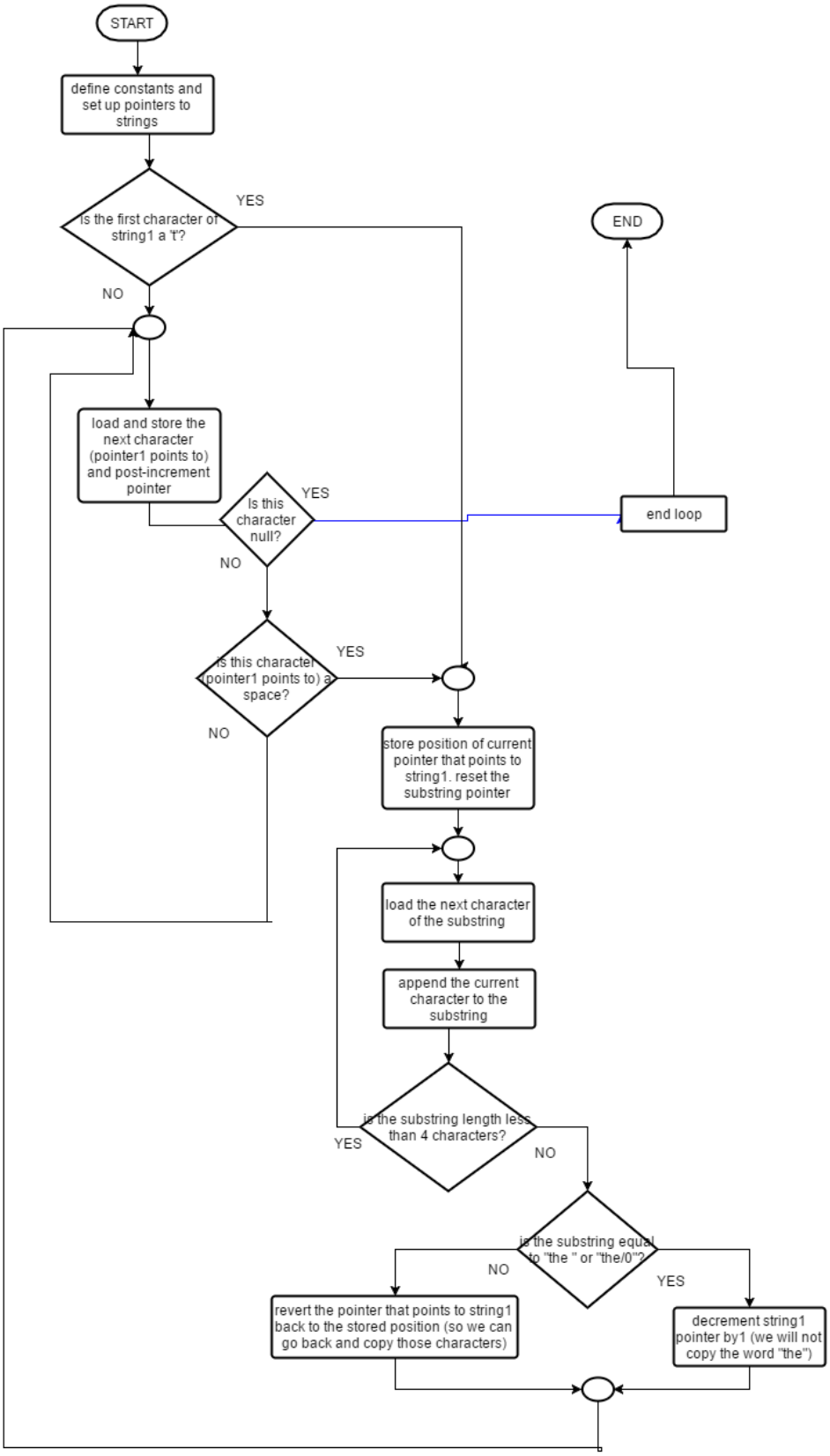

START
read the next character of the first string
Is the character null?
YES
NO
write that character into string 3
end loop
read the next character of the second string
Is the character null?
NO
write that character into string 3
YES
end loop
END

CS2208 ASS 4 - vlam54

```
;Vivian Lam
;program that concatenates two strings into another string
           AREA prog1, CODE, READONLY
        ENTRY
        LDR r0, =STRING1 ;load address of string1 into register 0
        LDR r1, =STRING2 ;load address of string2 into register 1
        LDR r3, =STRING3 ;load address of string3 into register 3
           LDR r5, EoS1;make r5 point to the EoS character

loopStr1   LDRB r4, [r0], #1 ;load the first bit of string1 into r4

                ;check if null character first
                CMP r4, #0x00         ;check to see if we reached end
of the string
                BEQ loopStr2          ;if null then go to next loop
                STRB r4, [r3], #1     ;otherwise store that character
in r4 into string3
                B loopStr1            ;continue looping if haven't
reached EoS character

loopStr2        LDRB r4, [r1], #1 ;load the first bit of string1 into
r4
                CMP r4, #0x00      ;check to see if we reached end of
the string
                BEQ theEnd            ;if null then we are done
copying the strings
                STRB r4, [r3], #1 ;store that character in r4 into
string3
                B loopStr2      ;continue looping if haven't reached
EoS character


theEnd
        LDRB r4, [r5], #1 ;load and store the EoS null character into
string3 to signal the end of the string
        STRB r4, [r3]

loop b loop;infinite loop so no error

        ALIGN
STRING1 DCB "This is a test string1"            ;String1
EoS1    DCB 0x00                                ;end of string1
STRING2 DCB "This is a test string2"            ;String
EoS2    DCB 0x00                                ;end of string2
STRING3 space 0xFF
        END
```

$$

Question 2


START
define constants and set up pointers to strings
Is the first character of string1 a 't'?
YES
NO
END
load and store the next character (pointer1 points to) and post-increment pointer
Is this character null?
YES
NO
end loop
is this character (pointer1 points to) a space?
YES
NO
store position of current pointer that points to string1. reset the substring pointer
load the next character of the substring
append the current character to the substring
is the substring length less than 4 characters?
YES
NO
is the substring equal to "the " or "the/0"?
NO
YES
revert the pointer that points to string1 back to the stored position (so we can go back and copy those characters)
decrement string1 pointer by1 (we will not copy the word "the")

```
;Vivian Lam
;program to copy a string into another string, but remove any instance
of the word "the"
          AREA prog2, CODE, READONLY
        ENTRY

;define contants
spacechara EQU  0x20 ;space character constant used to check if
current character is a space
charat          EQU  0x74 ;'t' character constant used to check if
current character is t
null       EQU  0x00 ;null character constant used to determine if end
of string

;set up the strings
        ADR r0, STRING1 ;load address of string1 into register 0
           ADR r1, STRING2 ;load address of string2 into register 1
           LDR r2, =0x74686520 ;load the string "the " into r2. This
will be used for comparing to check if the instance of it occurs
           LDR r3, =0x74686500 ;load the string "the/0" into r3. This
will be used for comparing to check if the instance of it occurs
           ;r4 will be used to load the next byte in string1
           ;r5 will be used to build the substring
           ;r6 stores the original position of pointer1 before
oomparing


;since we want to remove the word "the" we must consider diff
scenarios:
;case1: "the" is at the beginning of the string (no space before t)
;case2: " the " is in the middle of the string (space before and after
the word "the")
;case3: "the/0" is at the end of the string (space before "the" and
null character after)


;first check case1: if "the" is at begnning of string
           LDRB r4, [r0];load first character from strin1
           CMP r4,#charat;if the character is a 't' branch to check if
the first word is actually "the " or "the/0"
           BEQ check



;loop to go through the string
loop  LDRB r4, [r0], #1     ;load character from string1 and post-
increment pointer
           STRB r4, [r1], #1     ;Store the character and post-
increment pointer

           CMP r4, #null   ;check to see if current char is null
```

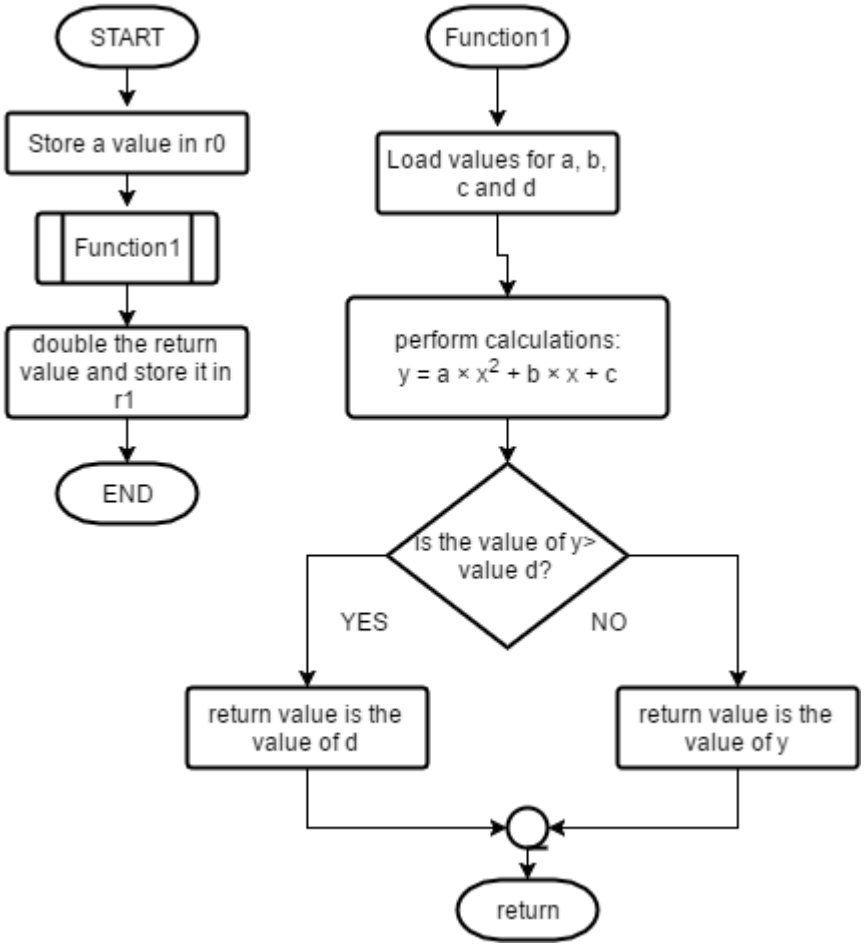
```
          BEQ endless          ;if so then branch to end of program,
we are dont copying

          CMP r4, #spacechara ;check to see if current char is a
space
          BNE loop   ;if so then we might have case 2 or 3 (" the " or
" the/0"), in which we would continue through the code.
                           ;otherwise go back to beginning of loop to
get next character


;now to check if we have case 2 or 3:
check MOV   r6, r0     ;store current position of strin1 pointer (if we
don't get an instance of the word "the",
                           ; the value of the pointer will be reverted
back to this stored value)
          MOV   r5, #0     ;Reset r5 because this serves as a pointer
for the next 4 characters (for checking substring) in string1

;building substring to check if instance of "the" follows after the
space
substr    LDRB r4, [r0], #1 ;load character into temp substring
          CMP r4, #null    ;check if current char is Eos chara
          ;If so then we have case 3, thus when we do the next check
we will be comparing "the/0"
          MOVEQ r2, r3;and so we must change the value of r2 to be
"the/0", instead of " the "

          ADD        r5, r4
     ;Append the character(r4) to the substring (r5)
          CMP        r5, #0x10000000
     ;Check if the substring contains less than 4 characters
          LSLLT      r5, #8                                    ;If
yes then shift the substring 1 byte left
          BLT        substr
     ;continue building the substring (else, the substring is built)

;check if substring follows case 2 or 3 ("the " or "the/0")
          CMP r5, r2       ;checking case 2
          SUBEQ r0, #1     ;if so then decrement string1 pointer by 1
          MOVNE r0, r6     ;otherwise revert pointer to position
before the check
     ;    CMP   r5, r3          ;checking case 3
     ;    SUBEQ r0, #1     ;if so then decrement string1 pointer by 1
     ;    MOVNE r0, r6     ;otherwise revert pointer to position
before the check


          B loop ;repeat loop. loop will exit when above condition
(null character reached) is fulfilled

endless B endless ;Infinite loop so no error
```

CS2208 ASS 4 - vlam54

```
STRING1 DCB "the they The  the 123 athe   the" ;String1
;STRING1 DCB "" ;String1
;STRING1 DCB "the" ;String1
;STRING1 DCB "The" ;String1
;STRING1 DCB "them   the   the1" ;String1
;STRING1 DCB "and the man said they must go" ;String1
EoS DCB 0x00 ;end of string1
STRING2 space 0xFF

           ALIGN
           END
```

$$$

Question 3


START
Store a value in r0
Function1
double the return value and store it in r1
END
Function1
Load values for a, b, c and d
perform calculations: $y = a \times x^2 + b \times x + c$
Is the value of y> value d?
YES
NO
return value is the value of d
return value is the value of y
return

```
;Vivian Lam
;program that calls a subroutine to perform a calculation, clips the
result if it's > d, and then return and double that return value and
store in r1

          AREA prog3, CODE, READONLY
        ENTRY
;-----------------------------------------------------
Main
          MOV r0, #3 ;store a value in r0
          ;MOV r2, #1          ;test value to see if register is
restored after funciton call
          ADR SP, Stackk  ;initialize pointer to stack
          BL Function1 ;jump to the function

          ADD r1, r0, r0 ;double the new value of r0 and store it
into r1

loop B loop          ;infinite loop so no error

;-----------------------------------------------------
;performs the calculation: y = a × x^2 + b × x + c
;where x is r0. the return value is stored in r0
     ;use r2-r5 to store a,b,c,d (respectively)
     ;use r6 to store x^2
     ;use r8 to store b × x
     ;use r7 to store current output value
Function1  STMIA SP!, {r2-r8};store working registers and link
register

                LDR r2, memA ;load values a,b,c,d into registers r2-r5
(make r2 point to memA etc.)
                LDR r3, memB
                LDR r4, memC
                LDR r5, memD

          ;perform calculation: y = a × x^2 + b × x + c
                MUL r6, r0, r0        ;y=x^2
                MUL r7, r6, r2        ;y=a × y
                MUL r8, r3, r0        ;b × x
                ADD r7, r7, r8        ;y = y+ b × x
                ADD r7, r7, r4        ;y=y+c

                MOV r0, r7      ;store the output value to r0
                CMP r0, r5 ;check to see if the result is > value of d
                MOVGT r0, r5;if so then the output value will be
clipped to d and is stored in r0
                ;otherwise the output value is not clipped and is
stored in r0

          LDMDB SP!, {r2-r8};restore the working registers back to
normal
```

```
            MOV PC,LR                    ;return from function 1 by
making the value of PC equal to the link register
;----------------------------------------------------

memA  DCD 5 ;values for a, b, c, d, which are used in the function and
for clipping
memB  DCD 6
memC  DCD 7
memD  DCD 10
Stackk      space 0xFF ;space for stack
            ALIGN
            END
```

$$