Electrical and Computer Engineering

ENCS2380 – Computer Organization and Microprocessors- Fall 2023

Course Project (2)

Student Name:	Student ID NO:
Saif Al-Deen Battah	1170986
Abdulkailq injas	1173251

Instructor: Dr. Abualseoud Hanani.

Section NO: 1.

Date: 20/2/2023.

Contents

Part 1

```
1
2       PRESERVE8
3       THUMB
4       AREA RESET, DATA, READONLY
5        EXPORT __Vectors
6
7       ;part 1
8
9           AREA      String, DATA, READWRITE
10          sentence1  DCB "Arm Assembly Programming",0
11          sentence2  DCB "Keil uVision 5 Software",0
12
```

Part 2

```
13  ;part 2
14
15      AREA     Convert, CODE, READONLY
16
17          ; Declare the procedure
18          CNV_CASE
19          EXPORT  CNV_CASE
20          CNV_CASE
21          PUSH    {LR}
22          MOV     r2, #0          ; Count of converted letters
23      loop
24          LDRB    r3, [r0], #1    ; Load a byte from the string
25          CMP     r3, #0          ; Check for end of string
26          BEQ     end
27          CMP     r3, #'A'        ; Check if the character is a capital letter
28          BLT     not_capital
29          CMP     r3, #'Z'
30          BGT     not_capital
31          ADD     r3, r3, #32     ; Convert to lowercase
32          ADD     r2, r2, #1      ; Increment count of converted letters
33      not_capital
34          STRB    r3, [r1], #1    ; Store the converted byte in the new string
35          B       loop
36      end
37          MOV     r0, r2          ; Return the count of converted letters in R0
38          POP     {PC}
39
40          ; Convert the first string to lowercase
41          AREA    TXT1, DATA, READWRITE
42          CNV_CASE
43          LDR     r0, =sentence1  ; Load the address of the first string
44          LDR     r1, =TXT1       ; Load the address of the new string
45          BL      CNV_CASE        ; Call the procedure
```

```
45          BL      CNV_CASE        ; Call the procedure
46          LDR     r1, =Count1     ; Load the address of the count variable
47          STR     r0, [r1]        ; Store the count of converted letters
48
49          ; Convert the second string to lowercase
50          AREA    TXT2, DATA, READWRITE
51          CNV_CASE
52          LDR     r0, =sentence2  ; Load the address of the second string
53          LDR     r1, =TXT2       ; Load the address of the new string
54          BL      CNV_CASE        ; Call the procedure
55          LDR     r1, =Count2     ; Load the address of the count variable
56          STR     r0, [r1]        ; Store the count of converted letters
```

Part 3

```
;part 3
AREA     Common, DATA, READWRITE

; Declare the procedure
COMMON_CHARS
EXPORT  COMMON_CHARS
COMMON_CHARS
PUSH    {LR}
MOV     r2, #0          ; Counter for common characters
MOV     r4, #0          ; Index for string 2
loop1
LDRB    r3, [r0], #1    ; Load a byte from the first string
CMP     r3, #0          ; Check for end of string
BEQ     end
MOV     r1, #0          ; Index for string 2
loop2
LDRB    r5, [r1, r5]    ; Load a byte from the second string
CMP     r5, #0          ; Check for end of string
BEQ     end2
CMP     r5, #32         ; Convert to lowercase if it's a capital letter
BLT     not_capital2
CMP     r5, #'Z'
BGT     not_capital2
ADD     r5, r5, #32
not_capital2
CMP     r3, r5          ; Compare the characters
BNE     not_common
ADD     r2, r2, #1      ; Increment the counter
B       end2
not_common
ADD     r1, r1, #1      ; Increment the index for string 2
B       loop2
```

```
B       loop2
end2
ADD     r4, r4, #1      ; Increment the index for string 1
MOV     r5, #0          ; Reset the index for string 2
B       loop1
end
MOV     r0, r2          ; Return the counter in R0
POP     {PC}

; Compute the number of common characters between the two strings
AREA    COMMON, DATA, READWRITE
COMMON_CHARS
LDR     r0, =TXT1       ; Load the address of the first string
LDR     r1, =TXT2       ; Load the address of the second string
BL      COMMON_CHARS    ; Call the procedure
LDR     r1, =COMMON     ; Load the address of the counter variable
STR     r0, [r1]        ; Store the counter
```

Part 4

```
107     ; part 4
108
109             LSL     r3, r3, #1      ; Shift the carry bit into bit 0
110         BIC     r3, r3, #1      ; Invert bit 0
111         STRB    r3, [r2], #1    ; Store the encrypted byte and increment the destination pointer
112         CMP     r3, #0          ; Check for end of string
113         BNE     loop
114         POP     {LR}
115
116         ; Encrypt the first string
117         AREA    Encryption, DATA, READWRITE
118         ENCRYPT_STRING1
119         LDR     r0, =TXT1       ; Load the address of the first string
120         LDR     r1, =ENCRYPT1   ; Load the address of the destination array
121         BL      ENCRYPT_STRING  ; Call the encryption procedure
122
123         ; Encrypt the second string
124         ENCRYPT_STRING2
125         LDR     r0, =TXT2       ; Load the address of the second string
126         LDR     r1, =ENCRYPT2   ; Load the address of the destination array
127         BL      ENCRYPT_STRING  ; Call the encryption procedure
128
129
130
```

Appendix

```
        PRESERVE8
        THUMB
        AREA RESET, DATA, READONLY
         EXPORT __Vectors

        ;part 1

                AREA     String, DATA, READWRITE
    sentence1  DCB "Arm Assembly Programming",0
    sentence2  DCB "Keil uVision 5 Software",0

  ;part 2

         AREA     Convert, CODE, READONLY

    ; Declare the procedure
    CNV_CASE
    EXPORT  CNV_CASE
    CNV_CASE
    PUSH    {LR}
    MOV     r2, #0          ; Count of converted letters
  loop
    LDRB    r3, [r0], #1    ; Load a byte from the string
```

```
    CMP     r3, #0          ; Check for end of string
    BEQ     end
    CMP     r3, #'A'        ; Check if the character is a capital letter
    BLT     not_capital
    CMP     r3, #'Z'
    BGT     not_capital
    ADD     r3, r3, #32     ; Convert to lowercase
    ADD     r2, r2, #1      ; Increment count of converted letters
not_capital
    STRB    r3, [r1], #1    ; Store the converted byte in the new string
    B       loop
end
    MOV     r0, r2          ; Return the count of converted letters in R0
    POP     {PC}

    ; Convert the first string to lowercase
    AREA    TXT1, DATA, READWRITE
    CNV_CASE
    LDR     r0, =sentence1  ; Load the address of the first string
    LDR     r1, =TXT1       ; Load the address of the new string
    BL      CNV_CASE        ; Call the procedure
    LDR     r1, =Count1     ; Load the address of the count variable
    STR     r0, [r1]        ; Store the count of converted letters
```

```
    ; Convert the second string to lowercase
    AREA    TXT2, DATA, READWRITE
    CNV_CASE
    LDR     r0, =sentence2  ; Load the address of the second string
    LDR     r1, =TXT2       ; Load the address of the new string
    BL      CNV_CASE        ; Call the procedure
    LDR     r1, =Count2     ; Load the address of the count variable
    STR     r0, [r1]        ; Store the count of converted letters

                ;part 3
                AREA     Common, DATA, READWRITE

    ; Declare the procedure
    COMMON_CHARS
    EXPORT  COMMON_CHARS
    COMMON_CHARS
    PUSH    {LR}
    MOV     r2, #0          ; Counter for common characters
    MOV     r4, #0          ; Index for string 2
loop1
    LDRB    r3, [r0], #1    ; Load a byte from the first string
    CMP     r3, #0          ; Check for end of string
    BEQ     end
    MOV     r1, #0          ; Index for string 2
```

```
loop2
   LDRB   r5, [r1, r5]   ; Load a byte from the second string
   CMP    r5, #0         ; Check for end of string
   BEQ    end2
   CMP    r5, #32        ; Convert to lowercase if it's a capital letter
   BLT    not_capital2
   CMP    r5, #'Z'
   BGT    not_capital2
   ADD    r5, r5, #32
not_capital2
   CMP    r3, r5         ; Compare the characters
   BNE    not_common
   ADD    r2, r2, #1     ; Increment the counter
   B      end2
not_common
   ADD    r1, r1, #1     ; Increment the index for string 2
   B      loop2
end2
   ADD    r4, r4, #1     ; Increment the index for string 1
   MOV    r5, #0         ; Reset the index for string 2
   B      loop1
end
   MOV    r0, r2         ; Return the counter in R0
   POP    {PC}
```

```
; Compute the number of common characters between the two strings
AREA    COMMON, DATA, READWRITE
COMMON_CHARS
LDR     r0, =TXT1       ; Load the address of the first string
LDR     r1, =TXT2       ; Load the address of the second string
BL      COMMON_CHARS    ; Call the procedure
LDR     r1, =COMMON     ; Load the address of the counter variable
STR     r0, [r1]        ; Store the counter

; part 4

            LSL     r3, r3, #1     ; Shift the carry bit into bit 0
BIC     r3, r3, #1     ; Invert bit 0
STRB    r3, [r2], #1   ; Store the encrypted byte and increment the destination
pointer
CMP     r3, #0         ; Check for end of string
BNE     loop
POP     {LR}

; Encrypt the first string
AREA    Encryption, DATA, READWRITE
ENCRYPT_STRING1
LDR     r0, =TXT1       ; Load the address of the first string
LDR     r1, =ENCRYPT1   ; Load the address of the destination array
```

```
BL      ENCRYPT_STRING  ; Call the encryption procedure


; Encrypt the second string
ENCRYPT_STRING2
LDR     r0, =TXT2       ; Load the address of the second string
LDR     r1, =ENCRYPT2   ; Load the address of the destination array
BL      ENCRYPT_STRING  ; Call the encryption procedure
```