CSE 379
Lab #3
Part 2
Spring 2019

Description
Write four (4) ARM assembly language subroutines: *read_string*, *output_string*, *uart_init*, and *lab3*. These subroutines will allow the ARM processor to receive user-entered data from *PuTTy* and transmit data to be displayed in *PuTTy* via the UART. All user input should be echoed back to the display via your program. Your program should ask the user to enter two integers, between 0 and 999 (inclusive), and then report which is larger along with their difference. After the user enters each number they should hit the *Enter* key to indicate they have finished entering the number. Each integer entered should be stored as a null terminated ASCII string in memory at the labels *num1* and *num2* respectively. The user should NOT be asked how many ***digits*** will be entered for each number that is entered.

Presentation is important. For example, when user data is expected, the program should prompt the user for data, and supply the appropriate instructions for doing so. When the program reports the results they should be descriptive, explicitly stating which number is larger and the difference.

The subroutines that need to be submitted (including those from Part #1) are defined as follows:

- *uart_init* initializes the user UART for use. This is your version (in assembly language) of the C function *serial_init*.
- *output_character* transmits a character from the UART to *PuTTy*. The character is passed in *r0*.
- *read_string* reads a string entered in *PuTTy* and stores it as a null-terminated string in memory. The user terminates the string by hitting *Enter*. The base address of the string should be passed into the routine in *r4*.
- *output_string* transmits a null-terminated string for display in *PuTTy*. The base address of the string should be passed into the routine in *r4*.
- *read_character* reads a character which is received by the UART from *PuTTy*, returning the character in *r0*.
- *lab3* is the subroutine which handles getting the user data, comparing the numbers, calculating the difference, and displaying the results. This routine is called by your C program and is your top-level assembly language routine.

Skeleton Code
The following assembly language example shows you how to store an ASCII string in memory. Pay particular attention to the code in red. This can be added to the code developed in Part #1.

```
    .data
    .global prompt
    .global expression
prompt:   .string "Your prompts are placed here",0
expression: .string "Your expression is stored here",0
    .text
    .global lab3
U0LSR:   .equ 0x18        ; UART0 Line Status Register
ptr_to_prompt: .word prompt

lab3:
    STMFD SP!,{lr} ; Store register lr on stack
    ldr r4, ptr_to_prompt
     ; Your code is placed here
    LDMFD sp!, {lr}
    mov pc, lr
```

```
        .end
```

Testing

Use the C subroutine *serial_init* to get your program working. After your program works, replace *serial_init* with *uart_init*. By developing the program in this manner, it will be easier to identify and fix errors in *uart_init*.

Partners

You will work with a partner in this lab. Your partner *MUST* be the same partner you worked with on Part #1 of this lab.

Documentation

Your program must be clearly commented, and documentation must also be provided. The documentation must follow the guidelines covered in lecture (found on the *Lectures* webpage of the course website). Your comments should describe what *each* section of your program does. To receive full credit on your documentation, you must submit a draft of your flowchart before you start working on the lab in your regularly scheduled lab time on Wednesday, February 19 or Thursday, February 20.

Submissions

Your source code (C and assembly) and your documentation (as a PDF) must be submitted online using the submit command (*submit_cse379 lab3wrapper.c lab3.s lab3documentation.pdf*) on *timberlake.cse.buffalo.edu* ***before 11:59 PM on Friday, February 21, 2020***. Your documentation will be used along with the code you submitted when you perform the debug exercise for Lab #3.