CSE 320 **Fall 2020**

Computer Project #10

Assignment Overview

This assignment develops familiarity with arrays in assembly language. You will develop a set of ARM assembly language functions which implement common string operations.

It is worth 30 points (3% of course grade) and must be completed no later than 11:59 PM on Thursday, 12/3.

Assignment Deliverables

The deliverables for this assignment are the following files:

`proj10.support.s` – the source code for your support module
`proj10.driver.c` – the source code for your driver module
`proj10.makefile` – the makefile which produces **`proj10`**

Be sure to use the specified file names and to submit them for grading via the CSE handin system.

Assignment Specifications

A low-level character string is defined as a sequence of zero or more characters in an array of type "char", with a null byte to mark the end of the string.

1. You will develop the ARM assembly language functions listed below:

```
unsigned int length( const char* source );

void copy( char* dest, const char* source );

void append( char* dest, const char* source );

char* duplicate( const char* source );

int compare( const char* source1, const char* source2 );
```

Those five functions (and any "helper" functions which you develop) will constitute a module named "proj10.support.s". The functions in that module will not call any C library functions except function "malloc".

Function "length" will return the number of characters in string "source"; the count will not include the terminating null byte.

Function "copy" will copy the contents of string "source" into string "dest".

Function "append" will append the contents of string "source" to the end of string "dest".

Function "duplicate" will return the address of a duplicate of string "source"; it will use function "malloc" to allocation additional memory and will copy string "source" into that memory.

Function "compare" will return the results of comparing string "source1" and string "source2". If "source1" and "source2" are equal, it will return 0. If "source1" is less than "source2", it will return a negative integer. If "source1" is greater than "source2", it will return a positive integer.

2. You will develop a driver module to test your implementation of the support module. The driver module will consist of function "main" and any additional helper functions which you choose to implement. All output will be appropriately labeled.

Your driver module may not be written as an interactive program, where the user supplies input in response to prompts. Instead, your test cases will be included in the source code as literal constants.

Assignment Notes

1. The functions in your support module must be hand-written ARM assembly language functions (you may not submit compiler-generated assembly language functions).

2. Your program will be translated and linked using "gcc". For example, the following commands could be used to translate and link your program, then load and execute it:

```
<prompt> gcc -c proj10.support.s
<prompt> gcc –Wall –c proj10.driver.c
<prompt> gcc proj10.support.o proj10.driver.o –o proj10
<prompt> proj10
```

3. In order to interface ARM assembly language functions with C functions, you must follow certain conventions about register usage.

The calling function will place up to four parameters in registers R0 through R3 (with the first argument in register R0).

The called function must save and restore registers R4 through R11 if it uses any of those registers (the calling function assumes that registers R4 through R11 are not altered by calling another function).

The called function place its return value in register R0 before returning to the calling function.

Registers R12, R13, R14 and R15 are used by the system and their contents must not be modified by your functions.