**Lab Set 2 (Due: Tuesday, 04/29/2014, 40 points)**

Problems:

**Week 5**: Perform programming exercise 8 (Copy a String Backwards) on pages 130-131.

Modify programming exercise 8 to include another way to copy a string backward.  Declare a new string as the second data item, copy the original string backward to this new string using push and pop instructions (i.e., push original string onto stack and pop it back to the new string), and output the new string (make sure no extra characters).

**Week 6**: Perform Programming Exercise 9 (Summation Program) on page 179 of text book.

**Week 7**: Put together a simple program to demonstrate:  
a.  Flags are set properly for both unsigned and signed comparisons (cmp instruction)  
b.  Assume that XOR instruction is not available, set up some instructions to simulate XOR of two bits.  Use AL to hold 1 bit, use BL  to hold the other bit, and assign result to CL (CL = AL XOR BL).  Hint: can either use AND, OR, and NOT instructions or CMP instruction.

Set up a procedure IsAlpha (similar to IsDigit discussed in class) that receives a character in AL register and it sets ZF if the character is an uppercase or lowercase letter.  Test your procedure by setting up an array of characters like "This is a TEST! 1 2 3", process the array to count and then print the number of letters in the array.

**Week 8**: Perform Programming Exercise 4 on page 268: Encryption Using Rotate Operations.  Encrypt the following string "send more money" and output the message using hexadecimal (output each encrypted character as 2-digit hexadecimal).

Perform Programming Exercise #6 on page 268 (GCD).  Assume that 32-bit unsigned values are used and store result in eax.

**Week 9**: Perform Programming Exercise #6 on page 330 ( recursive GCD).  Assume that 32-bit unsigned values are used and store result in eax.  Must use stack parameters and set up your own stack frame (i.e., don't use MASM directives).  Use recursive factorial as an example.

8. Copy a String Backwards

Write a program using the LOOP instruction with indirect addressing that copies a string from

sourceto target, reversing the character order in the process. Use the following variables:

source BYTE "This is the source string",0

target BYTE SIZEOF source DUP('#')

Insert the following statements immediately after the loop to display the hexadecimal contents

of the target string:

mov esi,OFFSET target ; offset of variable

mov ebx,1 ; byte format

mov ecx,SIZEOF target ; counter

call DumpMem

If your program works correctly, it will display the following sequence of hexadecimal bytes:

67 6E 69 72 74 73 20 65 63 72 75 6F 73 20 65 68

74 20 73 69 20 73 69 68 54

(The DumpMem procedure is explained in Section 5.3.2.) (A VideoNote for this exercise is posted

on the Web site.)