**Andrew Tran**

**Lab Experience Twelve**

**Objectives:**

1. Understanding the shift instructions.
2. Performing addition/subtraction and multiplication/division.
3. Multiplying in Assembler
4. Dividing in Assembler
5. Examining the addresses pushed onto the runtime stack when invoke is used.
6. Utilizing the INVOKE directive

**What you are to do:**

1. Do problems 4 and 6 on pages 284-285.
2. Do problem 10 on pages 350-351.
   1. Copy and paste each screen shots showing the execution of your program for all cases into your word document. Do not paste the assembler code.

**What to hand in:**

1. Printouts of all the assembler programs from Visual Studio.

1. A printout of the word document.
2. Compress the assembler file and the word file into a single file using your name and the lab number as the filename.
3. Place the compressed file into the D2L Dropbox folder titled Lab Experience Twelve.

**Due Date: As indicated on the dropbox folder.**

Problem 1:

INCLUDE Irvine32.inc

.data

key BYTE -6, 4, 2, 0, -7, 3, -1, 5, -4, 8

array BYTE "AndrewTranLab12", 0

.code

main PROC

mov eax, OFFSET array

mov edi, OFFSET key

mov ecx, 20

mov edx, OFFSET array

call WriteString

call Crlf

call Encrypt

mov edx, OFFSET array

call WriteString

call Crlf

exit

main ENDP

Encrypt Proc

encryption:

push ecx

mov cl, [edi]

cmp cl, 0

jge right

rol BYTE PTR[eax], cl ;left

jmp incre

right:

ror BYTE PTR[eax], cl ;right

incre:

inc eax

inc edi

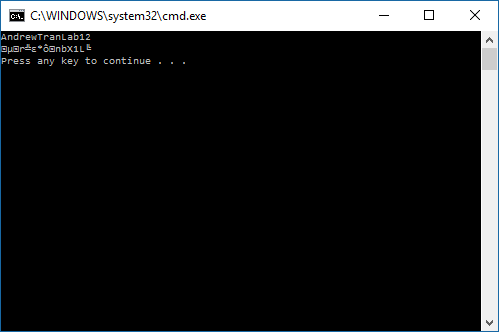
pop ecx

loop encryption

ret

Encrypt ENDP

END main



Problem 2:

INCLUDE Irvine32.inc

.data

input BYTE "Enter an integer: ", 0

ansGCD BYTE "GCD: ", 0

val1 DWORD ?

val2 DWORD ?

.code

main PROC

mov edx, OFFSET input

call WriteString

call ReadDec

mov val1, eax

mov edx, OFFSET input

call WriteString

call ReadDec

mov val2, eax

mov eax, DWORD ptr[val1]

mov ebx, DWORD ptr[val2]

push val2

push val1

call gcd

exit

main ENDP

gcd PROC

xor edx, edx

mov eax, DWORD ptr[esp + 8]

mov ebx, DWORD ptr[esp + 4]

hi:

div ebx; eax / ebx

cmp edx, 0

je L1

mov eax, ebx

mov ebx, edx

jmp hi

L1:

mov eax, ebx

mov edx, OFFSET ansGCD

call WriteString

call WriteDec

call Crlf

ret 8

gcd ENDP

END main

