**Carson Blevins**

**Bellarmine University**

**Due Date: 10/12/15**

**Date Submitted: 10/23/15**

**CS 305 C Assembly Language Fall 2015**

**Due Date:** 10/12/15

**Exercise #1:**

**Source Code of Exercise #5 of Chapter 5:**

TITLE Random Number (Rand.asm)

; Creates a procedure that generates a random number from M to N-1.

Include Irvine32.inc

.code

main PROC

mov EBX,-300 ;lower bound

mov EAX,100

call BetterRandomRange

exit

main ENDP

BetterRandomRange PROC

;Creates a random number from -300 to 100 and prints out each result

neg EBX

add EBX,EAX

mov ECX,50

L1:

mov EAX,EBX ;upper bound

call RandomRange

sub EAX,300

call WriteInt

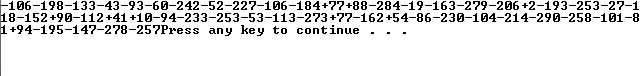
loop L1

ret

BetterRandomRange ENDP

END main

**Here is the output:**



**Exercise #2:**

**Source Code of Exercise #6 of Chapter 5:**

TITLE Random Strings (RStrings.asm)

; Creates a procedure that generates a random string and then hold those strings in an array

; The procedure will be called 20 times

Include Irvine32.inc

.data

array DWORD 0

.code

main PROC

;Calls Randomize to choose initiate the seed for the Random procedures and calls the newly created

;Random string to create a random string, print out the value, and assign each value to an array

call Randomize

call RandomString

INVOKE ExitProcess,0

main ENDP

RandomString PROC

;Creates a random string, prints out the value, and assigns each value to an array

mov esi,0

mov ecx,20

L1:

mov eax,26

call RandomRange

add eax,65

call WriteChar

mov array[esi],eax

add esi,TYPE array

Loop L1

ret

RandomString ENDP

END main

**Here is the output:**

KKSRBPFZALNXBEMUWOOTPress any key to continue . . .

**Exercise #3:**

**Source Code of Exercise #10 of Chapter 5:**

TITLE Fibonacci Generator (Fibg.asm)

; Creates a procedure that generates the Fibonacci number series and stores them in an array of

; doubleword

Include Irvine32.inc

.data

array DWORD 48 DUP(?)

.code

main PROC

call FibGen

call Display

INVOKE ExitProcess,0

main ENDP

FibGen PROC

;Generates the Fibonacci sequence and stores each value in an array

mov esi,0

mov eax,1

mov array[esi],eax

add esi,TYPE array

mov ebx,1

mov array[esi],ebx

mov esi,TYPE array

mov ecx,45

mov edx,0

L1:

push ecx

add eax,ebx

mov ecx,eax

mov eax,ebx

mov ebx,ecx

mov array[esi],eax

add esi,TYPE array

pop ecx

Loop L1

mov array[esi],ebx

ret

FibGen ENDP

Display PROC USES ESI ECX

;prints the elements in an array

mov esi,0

mov ecx,47

L2:

mov eax,array[esi]

call WriteDec

call WriteChar

add esi,TYPE array

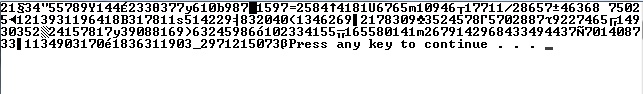
Loop L2

ret

Display ENDP

END main

**Here is the output:**

****