Comp 3350: Computer Organization & Assembly Language

HW5: Theme: Addressing modes

{6 Total Questions – Each worth 16.5 points }

1. Fill in the following memory diagram with the data provided below. Please assume that the data will begin being assigned at 00404000, which is the bottom row of the grid.

```
.data
Alpha  WORD   54, 76h
Beta   BYTE   1h
Gamma  DWORD 56789h
Delta  BYTE   2h
```

Address	*Variable*	*Data*
00404009	***Delta***	***02***
00404008	***Gamma+3***	***00***
00404007	***Gamma+2***	***05***
00404006	***Gamma+1***	***67***
00404005	***Gamma***	***89***
00404004	***Beta***	***01***
00404003	***Alpha+3***	***76***
00404002	***Alpha+2***	***00***
00404001	***Alpha+1***	***54***
00404000	***Alpha***	***00***

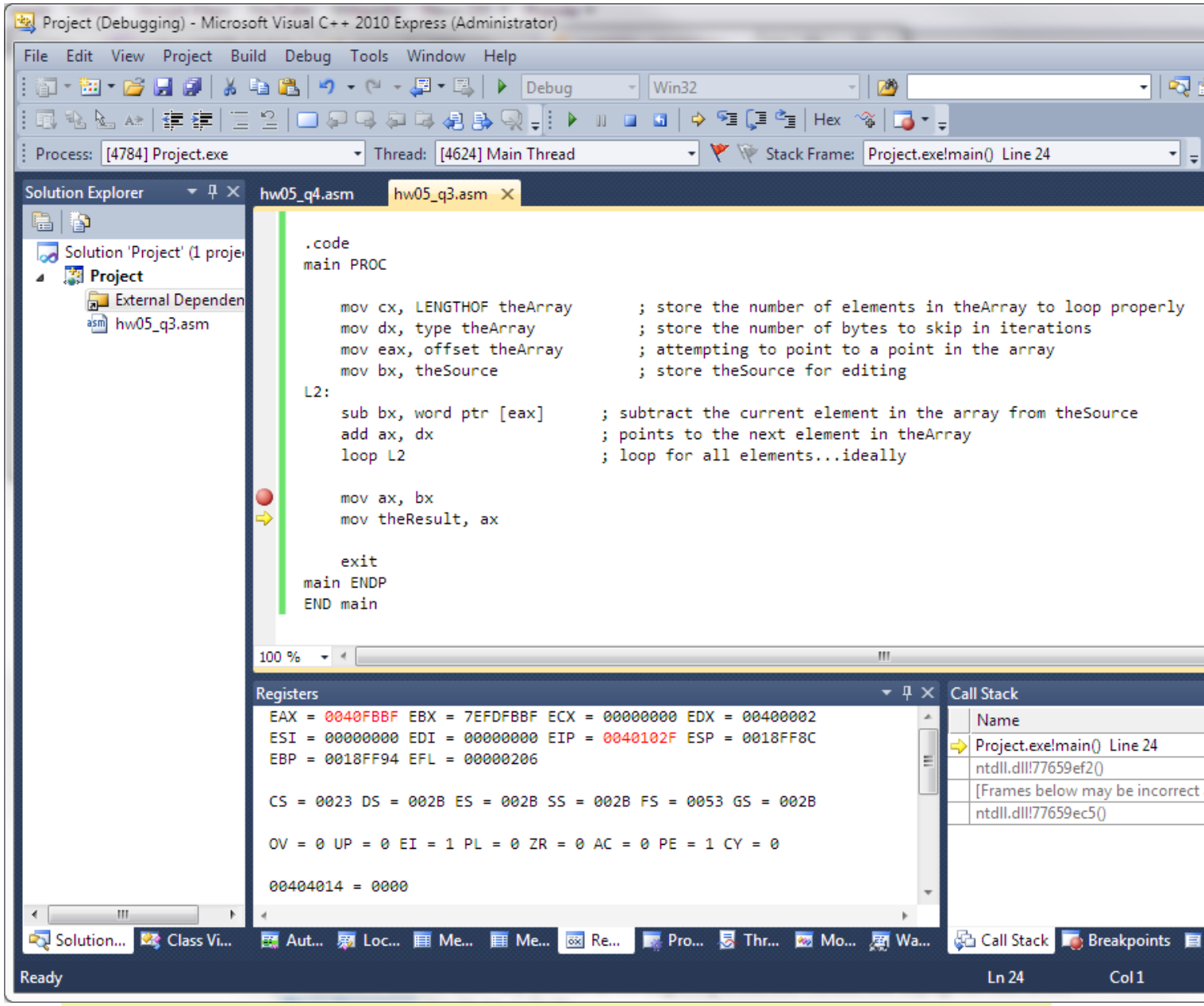
2. Copy the following code into your assembly development environment and single-step through it. For those instructions referencing memory, write the linear address.

```
TITLE Homework 5, Question 1                    (main.asm)

; Description: Memory reference exercise.
; Author: Matthew J Swann
; Version: 1.0, 2012-08-02

INCLUDE Irvine32.inc
.data
alpha  DWORD         1h, 2h
beta   DWORD         3h, 4h
gamma  DWORD         5

.code
main PROC
```

```
mov eax, 0Ah;              Immediate
mov ecx, eax;              register to register
mov edi, OFFSET beta;      Immediate
mov [gamma], eax;          Indirect // 00404010 = 00000005
mov esi, [gamma];          Direct // 00404010 = 0000000A
mov esi, 4;                Immediate
mov eax, beta[esi];        Indirect-offset // 0040400C = 00000004
mov ebx, OFFSET alpha;     Immediate
mov eax, [ebx];            Indirect // 00404000 = 00000001
mov eax, 4[ebx];           Indirect-displacement
                           ;      // 00404004 = 00000002
mov eax, 4[ebx][esi];      Base-Indirect-displacement
                           ;      // 00404008 = 00000003
mov eax, 8[ebx][esi];      Base-Indirect-displacement
                           ;      // 0040400C = 00000004
mov eax, 12[ebx][esi];     Base-Indirect-displacement
                           ;      // 00404010 = 0000000A

        exit
        main ENDP
        END main
```

3. Draft the `.code` section of a program that subtracts each element of an array from a single value. The `.data` section of the code is provided below. The program should: 1) iterate through "*theArray*", 2) subtracts the value at each index from "*theSource*', and 3) stores the resulting value in "*theResult*". Please embed your code into your homework submission along with a screenshot showing the final value. (Screenshots are in separate PDF).

```
TITLE Sum of elements of a DWORD array
; Author: Matthew J Swann
; Version 1.0, 2012-08-02

INCLUDE Irvine32.inc
.data
theArray WORD 1h, 2h, 4h, 8h, 16h, 32h, 64h, 128h, 256h
theSource WORD 0FFFFh
theResult WORD ?

.code
main PROC

   mov cx, LENGTHOF theArray   ; store number of elements in theArray to loop properly
   mov dx, type theArray       ; store the number of bytes to skip in iterations
   mov eax, offset theArray    ; reference a point in theArray
   mov bx, theSource           ; store theSource for editing
L2:
   sub bx, word ptr [eax]      ; subtract current element in theArray from theSource
   add ax, dx                  ; points to the next element in theArray
   loop L2                     ; loop for all elements...ideally

   mov ax, bx                  ; used only to highlight the result in screenshot
   mov theResult, bx           ; stores the result in theResult

        exit
        main ENDP
        END main
```

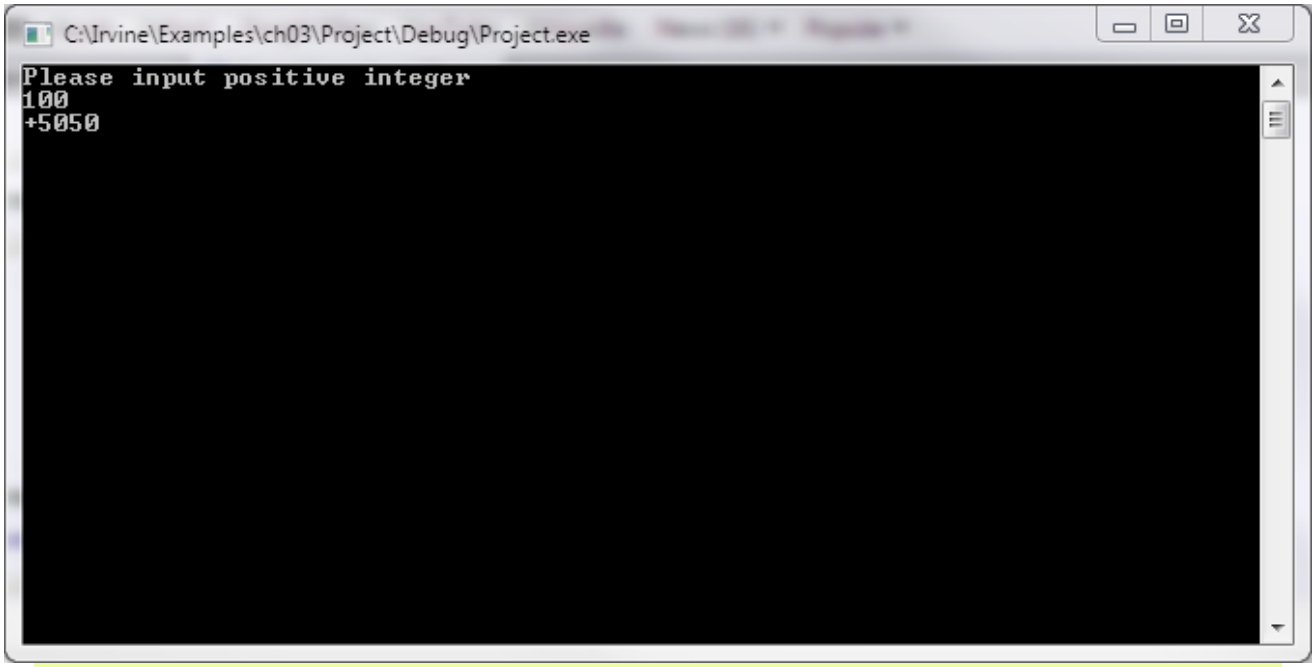


Note: the result is stored in EAX and EBX in this screenshot.

4. A Triangular Sequence is calculated as the summation of all positive integer values up to and including *n*. As such, $t_n = n + (n - 1) + (n - 2) + ... + 2 + 1$. Draft a program that:
   1) Prompts the user for integer input,
   2) Takes integer input from the user,
   3) Stores that value in a variable called "n",
   4) Calculates t_n, and;
   5) Prints the final value to the screen.

   Use the "call WriteInt" invocation, not "call DumpRegs". Other invocations that are likely necessary include: "call ReadInt", "call WriteString." The calculation can be done numerous ways, and all submissions that evidence proper programming practice are acceptable (including loops, recursion, etc.). In your homework submission, please embed both the code and one screen shot with user input supplied as 100.

```
TITLE Triangular Sequence Calculator                (hw05_q4.asm)

; Description: Takes an integer and sums all integers
; using the formula n + (n-1) + (n-2) + ... + 2 + 1
; Revision date: 2012/09/20

INCLUDE Irvine32.inc
.data
myMessage BYTE "Please input positive integer",0dh,0ah,0
n DWORD 0


.code
main PROC
	call Clrscr

	mov	edx,offset myMessage
	call WriteString		; Write prompt to the screen

	call ReadInt			; Allows user to input integer
	mov n, eax		; writes input integer to variable n
	mov ecx, n		; writes variable n to ecx for loop decrement
	dec ecx			; decrements ecx by 1, for initial n-1 addition

L0:				; begin the loop here
	add n, ecx		; add the decremented number to the current position
	loop L0			; jumps for looping; ends once ecx=zero is reached

	mov eax, n		; copies the calculated value to edx for display
	call writeInt		; displays the calculated data


	exit
main ENDP

END main
```


C:\Irvine\Examples\ch03\Project\Debug\Project.exe
Please input positive integer
100
+5050