

Comp 3350: Computer Organization & Assembly Language
HW # 5: Theme: Data Definitions, Addressing Modes, Arrays
All main questions carry equal weight.

(Credit awarded to only those answers for which work has been shown.)

1. **[Memory Map]** Fill in the following memory diagram with the data provided below. Please assume that the data segment begins at 0x0038E000.

```
.data
Alpha      DWORD      1A2B3C4Dh, 12AB34CCh
Beta       BYTE       7Ch
Gamma      DWORD      9ABCDEF0h
Delta      BYTE       12h
```

<i>Address</i>	<i>Variable</i>	<i>Data</i>
0038E000	<i>Alpha</i>	<i>4Dh</i>
0038E001	<i>Alpha</i>	<i>3Ch</i>
0038E002	<i>Alpha</i>	<i>2Bh</i>
0038E003	<i>Alpha</i>	<i>1Ah</i>
0038E004	<i>Alpha</i>	<i>CCh</i>
0038E005	<i>Alpha</i>	<i>34h</i>
0038E006	<i>Alpha</i>	<i>ABh</i>
0038E007	<i>Alpha</i>	<i>12h</i>
0038E008	<i>Beta</i>	<i>7Ch</i>
0038E009	<i>Gamma</i>	<i>F0h</i>
0038E00A	<i>Gamma</i>	<i>DEh</i>
0038E00B	<i>Gamma</i>	<i>BCh</i>
0038E00C	<i>Gamma</i>	<i>9Ah</i>
0038E00D	<i>Delta</i>	<i>12h</i>

2. **[Addressing Modes]** Copy the following code into your assembly development environment and single-step through it. For each single step execution, [submit the screenshot](#). For those instructions referencing memory, [do the linear address computation](#) (typewritten).

```
TITLE Addressing Modes                                (main.asm)
INCLUDE Irvine32.inc
.data
alpha      DWORD      65219751h, 24875139h
beta       DWORD      3B2C791Ah, 0A57716Dh
gamma      DWORD      0C58Bh

.code
main PROC
    mov eax, 1111h;                                Immediate
    mov ecx, eax;                                  Register to Register
    mov edi, OFFSET beta;                          Immediate
```

mov [gamma], eax;	Direct
mov esi, gamma;	Direct
mov esi, 4;	Immediate
mov eax, beta[esi];	Indirect-offset
mov ebx, OFFSET alpha;	Immediate
mov eax, [ebx];	Indirect
mov eax, 4[ebx];	Indirect-displacement
mov eax, 4[ebx][esi];	Base-Indirect-displacement

```
exit
main ENDP
END main
```

Before line 10:

```
Registers
EAX = 0019FFCC EBX = 002B6000 ECX = 00401005 EDX = 00401005 ESI = 00401005 EDI = 00401005 EIP = 00401010 ESP = 0019FF74 EBP = 0019FF80 EFL = 00000246
```

```
AddressingModes.asm - [X]
1  TITLE Addressing Modes (main.asm)
2  INCLUDE Irvine32.inc
3  .data
4  alpha DWORD 65219751h, 24875139h
5  beta  DWORD 3B2C791Ah, 0A57716Dh
6  gamma DWORD 0C58Bh
7
8  .code
9  main PROC
10 mov eax, 1111h; Immediate
11 mov ecx, eax; Register to Register
12 mov edi, OFFSET beta; Immediate
13 mov [gamma], eax; Direct
14 mov esi, gamma; Direct
15 mov esi, 4; Immediate
16 mov eax, beta[esi]; Indirect-offset
17 mov ebx, OFFSET alpha; Immediate
18 mov eax, [ebx]; Indirect
19 mov eax, 4[ebx]; Indirect-displacement
20 mov eax, 4[ebx][esi]; Base-Indirect-displacement
21 exit
22 main ENDP
23 END main
```

After line 10:

```
Registers
EAX = 00001111 EBX = 002B6000 ECX = 00401005 EDX = 00401005 ESI = 00401005 EDI = 00401005 EIP = 00401015 ESP = 0019FF74 EBP = 0019FF80 EFL = 00000246
```

```
AddressingModes.asm - [X]
1  TITLE Addressing Modes (main.asm)
2  INCLUDE Irvine32.inc
3  .data
4  alpha DWORD 65219751h, 24875139h
5  beta  DWORD 3B2C791Ah, 0A57716Dh
6  gamma DWORD 0C58Bh
7
8  .code
9  main PROC
10 mov eax, 1111h; Immediate
11 mov ecx, eax; Register to Register 51ms elapsed
12 mov edi, OFFSET beta; Immediate
13 mov [gamma], eax; Direct
14 mov esi, gamma; Direct
15 mov esi, 4; Immediate
16 mov eax, beta[esi]; Indirect-offset
17 mov ebx, OFFSET alpha; Immediate
18 mov eax, [ebx]; Indirect
19 mov eax, 4[ebx]; Indirect-displacement
20 mov eax, 4[ebx][esi]; Base-Indirect-displacement
21 exit
22 main ENDP
23 END main
```

After line 11:

```
Registers
EAX = 00001111 EBX = 002B6000 ECX = 00001111 EDX = 00401005 ESI = 00401005 EDI = 00401005 EIP = 00401017 ESP = 0019FF74 EBP = 0019FF80 EFL = 00000246

AddressingModes.asm
1  TITLE Addressing Modes (main.asm)
2  INCLUDE Irvine32.inc
3  .data
4  alpha DWORD 65219751h, 24875139h
5  beta  DWORD 3B2C791Ah, 0A57716Dh
6  gamma DWORD 0C58Bh
7
8  .code
9  main PROC
10     mov eax, 1111h; Immediate
11     mov ecx, eax; Register to Register
12     mov edi, OFFSET beta; Immediate 51ms elapsed
13     mov [gamma], eax; Direct
14     mov esi, gamma; Direct
15     mov esi, 4; Immediate
16     mov eax, beta[esi]; Indirect-offset
17     mov ebx, OFFSET alpha; Immediate
18     mov eax, [ebx]; Indirect
19     mov eax, 4[ebx]; Indirect-displacement
20     mov eax, 4[ebx][esi]; Base-Indirect-displacement
21     exit
22 main ENDP
23 END main
```

After line 12:

```
Registers
EAX = 00001111 EBX = 002B6000 ECX = 00001111 EDX = 00401005 ESI = 00401005 EDI = 00404008 EIP = 0040101C ESP = 0019FF74 EBP = 0019FF80 EFL = 00000246

0x00404010 = 0000C58B

AddressingModes.asm
1  TITLE Addressing Modes (main.asm)
2  INCLUDE Irvine32.inc
3  .data
4  alpha DWORD 65219751h, 24875139h
5  beta  DWORD 3B2C791Ah, 0A57716Dh
6  gamma DWORD 0C58Bh
7
8  .code
9  main PROC
10     mov eax, 1111h; Immediate
11     mov ecx, eax; Register to Register
12     mov edi, OFFSET beta; Immediate
13     mov [gamma], eax; Direct 51ms elapsed
14     mov esi, gamma; Direct
15     mov esi, 4; Immediate
16     mov eax, beta[esi]; Indirect-offset
17     mov ebx, OFFSET alpha; Immediate
18     mov eax, [ebx]; Indirect
19     mov eax, 4[ebx]; Indirect-displacement
20     mov eax, 4[ebx][esi]; Base-Indirect-displacement
21     exit
22 main ENDP
23 END main
```

After line 13:

```
Registers
EAX = 00001111 EBX = 002B6000 ECX = 00001111 EDX = 00401005 ESI = 00401005 EDI = 00404008 EIP = 00401021 ESP = 0019FF74 EBP = 0019FF80 EFL = 00000246
0x00404010 = 00001111

AddressingModes.asm
1  TITLE Addressing Modes (main.asm)
2  INCLUDE Irvine32.inc
3  .data
4  alpha DWORD 65219751h, 24875139h
5  beta  DWORD 3B2C791Ah, 0A57716Dh
6  gamma DWORD 0C588h
7
8  .code
9  main PROC
10     mov eax, 1111h; Immediate
11     mov ecx, eax; Register to Register
12     mov edi, OFFSET beta; Immediate
13     mov [gamma], eax; Direct
14     mov esi, gamma; Direct 51ms elapsed
15     mov esi, 4; Immediate
16     mov eax, beta[esi]; Indirect-offset
17     mov ebx, OFFSET alpha; Immediate
18     mov eax, [ebx]; Indirect
19     mov eax, 4[ebx]; Indirect-displacement
20     mov eax, 4[ebx][esi]; Base-Indirect-displacement
21     exit
22 main ENDP
23 END main
```

After line 14:

```
Registers
EAX = 00001111 EBX = 002B6000 ECX = 00001111 EDX = 00401005 ESI = 00001111 EDI = 00404008 EIP = 00401027 ESP = 0019FF74 EBP = 0019FF80 EFL = 00000246

AddressingModes.asm
1  TITLE Addressing Modes (main.asm)
2  INCLUDE Irvine32.inc
3  .data
4  alpha DWORD 65219751h, 24875139h
5  beta  DWORD 3B2C791Ah, 0A57716Dh
6  gamma DWORD 0C588h
7
8  .code
9  main PROC
10     mov eax, 1111h; Immediate
11     mov ecx, eax; Register to Register
12     mov edi, OFFSET beta; Immediate
13     mov [gamma], eax; Direct
14     mov esi, gamma; Direct
15     mov esi, 4; Immediate 51ms elapsed
16     mov eax, beta[esi]; Indirect-offset
17     mov ebx, OFFSET alpha; Immediate
18     mov eax, [ebx]; Indirect
19     mov eax, 4[ebx]; Indirect-displacement
20     mov eax, 4[ebx][esi]; Base-Indirect-displacement
21     exit
22 main ENDP
23 END main
```

After line 15:

```
Registers
EAX = 00001111 EBX = 002B6000 ECX = 00001111 EDX = 00401005 ESI = 00000004 EDI = 00404008 EIP = 0040102C ESP = 0019FF74 EBP = 0019FF80 EFL = 00000246
0x0040400C = 0A57716D

AddressingModes.asm
1  TITLE Addressing Modes (main.asm)
2  INCLUDE Irvine32.inc
3  .data
4  alpha DWORD 65219751h, 24875139h
5  beta  DWORD 3B2C791Ah, 0A57716Dh
6  gamma DWORD 0C588h
7
8  .code
9  main PROC
10     mov eax, 1111h; Immediate
11     mov ecx, eax; Register to Register
12     mov edi, OFFSET beta; Immediate
13     mov [gamma], eax; Direct
14     mov esi, gamma; Direct
15     mov esi, 4; Immediate
16     mov eax, beta[esi]; Indirect-offset 51ms elapsed
17     mov ebx, OFFSET alpha; Immediate
18     mov eax, [ebx]; Indirect
19     mov eax, 4[ebx]; Indirect-displacement
20     mov eax, 4[ebx][esi]; Base-Indirect-displacement
21     exit
22 main ENDP
23 END main
```

After line 16:

```
Registers
EAX = 0A57716D EBX = 002B6000 ECX = 00001111 EDX = 00401005 ESI = 00000004 EDI = 00404008 EIP = 00401032 ESP = 0019FF74 EBP = 0019FF80 EFL = 00000246

AddressingModes.asm
1  TITLE Addressing Modes (main.asm)
2  INCLUDE Irvine32.inc
3  .data
4  alpha DWORD 65219751h, 24875139h
5  beta  DWORD 3B2C791Ah, 0A57716Dh
6  gamma DWORD 0C588h
7
8  .code
9  main PROC
10     mov eax, 1111h; Immediate
11     mov ecx, eax; Register to Register
12     mov edi, OFFSET beta; Immediate
13     mov [gamma], eax; Direct
14     mov esi, gamma; Direct
15     mov esi, 4; Immediate
16     mov eax, beta[esi]; Indirect-offset
17     mov ebx, OFFSET alpha; Immediate 51ms elapsed
18     mov eax, [ebx]; Indirect
19     mov eax, 4[ebx]; Indirect-displacement
20     mov eax, 4[ebx][esi]; Base-Indirect-displacement
21     exit
22 main ENDP
23 END main
```

After line 17:

```
Registers
EAX = 0A57716D EBX = 00404000 ECX = 00001111 EDX = 00401005 ESI = 00000004 EDI = 00404008 EIP = 00401037 ESP = 0019FF74 EBP = 0019FF80 EFL = 00000246

0x00404000 = 65219751

AddressingModes.asm
1  TITLE Addressing Modes (main.asm)
2  INCLUDE Irvine32.inc
3  .data
4  alpha DWORD 65219751h, 24875139h
5  beta  DWORD 3B2C791Ah, 0A57716Dh
6  gamma DWORD 0C58Bh
7
8  .code
9  main PROC
10     mov eax, 1111h; Immediate
11     mov ecx, eax; Register to Register
12     mov edi, OFFSET beta; Immediate
13     mov [gamma], eax; Direct
14     mov esi, gamma; Direct
15     mov esi, 4; Immediate
16     mov eax, beta[esi]; Indirect-offset
17     mov ebx, OFFSET alpha; Immediate
18     mov eax, [ebx]; Indirect 51ms elapsed
19     mov eax, 4[ebx]; Indirect-displacement
20     mov eax, 4[ebx][esi]; Base-Indirect-displacement
21     exit
22 main ENDP
23 END main
```

After line 18:

```
Registers
EAX = 65219751 EBX = 00404000 ECX = 00001111 EDX = 00401005 ESI = 00000004 EDI = 00404008 EIP = 00401039 ESP = 0019FF74 EBP = 0019FF80 EFL = 00000246

0x00404004 = 24875139

AddressingModes.asm
1  TITLE Addressing Modes (main.asm)
2  INCLUDE Irvine32.inc
3  .data
4  alpha DWORD 65219751h, 24875139h
5  beta  DWORD 3B2C791Ah, 0A57716Dh
6  gamma DWORD 0C58Bh
7
8  .code
9  main PROC
10     mov eax, 1111h; Immediate
11     mov ecx, eax; Register to Register
12     mov edi, OFFSET beta; Immediate
13     mov [gamma], eax; Direct
14     mov esi, gamma; Direct
15     mov esi, 4; Immediate
16     mov eax, beta[esi]; Indirect-offset
17     mov ebx, OFFSET alpha; Immediate
18     mov eax, [ebx]; Indirect
19     mov eax, 4[ebx]; Indirect-displacement 51ms elapsed
20     mov eax, 4[ebx][esi]; Base-Indirect-displacement
21     exit
22 main ENDP
23 END main
```

After line 19:

```
Registers
EAX = 24875139 EBX = 00404000 ECX = 00001111 EDX = 00401005 ESI = 00000004 EDI = 00404008 EIP = 0040103C ESP = 0019FF74 EBP = 0019FF80 EFL = 00000246
0x00404008 = 3B2C791A

AddressingModes.asm
1  TITLE Addressing Modes (main.asm)
2  INCLUDE Irvine32.inc
3  .data
4  alpha DWORD 65219751h, 24875139h
5  beta  DWORD 3B2C791Ah, 0A57716Dh
6  gamma DWORD 0C58Bh
7
8  .code
9  main PROC
10     mov eax, 1111h; Immediate
11     mov ecx, eax; Register to Register
12     mov edi, OFFSET beta; Immediate
13     mov [gamma], eax; Direct
14     mov esi, gamma; Direct
15     mov esi, 4; Immediate
16     mov eax, beta[esi]; Indirect-offset
17     mov ebx, OFFSET alpha; Immediate
18     mov eax, [ebx]; Indirect
19     mov eax, 4[ebx]; Indirect-displacement
20     mov eax, 4[ebx][esi]; Base-Indirect-displacement 51ms elapsed
21     exit
22 main ENDP
23 END main
```

After line 20:

```
Registers
EAX = 3B2C791A EBX = 00404000 ECX = 00001111 EDX = 00401005 ESI = 00000004 EDI = 00404008 EIP = 00401040 ESP = 0019FF74 EBP = 0019FF80 EFL = 00000246

AddressingModes.asm
1  TITLE Addressing Modes (main.asm)
2  INCLUDE Irvine32.inc
3  .data
4  alpha DWORD 65219751h, 24875139h
5  beta  DWORD 3B2C791Ah, 0A57716Dh
6  gamma DWORD 0C58Bh
7
8  .code
9  main PROC
10     mov eax, 1111h; Immediate
11     mov ecx, eax; Register to Register
12     mov edi, OFFSET beta; Immediate
13     mov [gamma], eax; Direct
14     mov esi, gamma; Direct
15     mov esi, 4; Immediate
16     mov eax, beta[esi]; Indirect-offset
17     mov ebx, OFFSET alpha; Immediate
18     mov eax, [ebx]; Indirect
19     mov eax, 4[ebx]; Indirect-displacement
20     mov eax, 4[ebx][esi]; Base-Indirect-displacement
21     exit 51ms elapsed
22 main ENDP
23 END main
```

Linear Addresses for Each:

mov eax, 1111h;	
mov ecx, eax;	
mov edi, OFFSET beta;	
mov [gamma], eax;	0x00404010 = 1111h
mov esi, gamma;	esi = 01111h (0x00404010)
mov esi, 4;	
mov eax, beta[esi];	beta + 4, 00404012 (0x00404012)
mov ebx, OFFSET alpha;	
mov eax, [ebx];	eax = 65219751 (0x00404000)

<code>mov eax, 4[ebx];</code>	<code>4 + ebx, eax = 24875139 (0x00404004)</code>
<code>mov eax, 4[ebx][esi];</code>	<code>4 + ebx + esi, eax = 3B2C791A (0x00404008)</code>

3. **[Indirect addressing]** Write a program that subtracts the corresponding even indexed elements of Array2 from Array1 and stores the results in Array3; e.g. for the 8th element, Array3 [7] ← Array1 [7] - Array2 [7]. Note that Array3 will have about half the number of elements of the other two arrays. Include commands to display the elements of all the arrays. Submit screenshot of the displays of the elements of all the arrays. You can use WriteInt or WriteHex to display the elements of the arrays. Fill in Array1 and Array2 each by your own ten numbers.

```

1 | [TITLE SubArrays - Cameron Mathis (AddArrays.asm)
2 | INCLUDE Irvine32.inc
3 | .data
4 | Array1 WORD 7h, 8h, 7h, 6h, 5h, 4h, 3h, 2h, 1h, 0h, 1h
5 | Array2 WORD 4h, 2h, 3h, 4h, 5h, 6h, 7h, 8h, 9h, 8h, 7h
6 | Array3 WORD 10 DUP (?)
7 | a1message BYTE "Array1 Values: ",0
8 | a2message BYTE "Array2 Values: ",0
9 | a3message BYTE "Array3 Values: ",0
10 |
11 | .code
12 | main PROC
13 |     mov esi, OFFSET Array1
14 |     mov ecx, 5
15 |     mov edx, OFFSET Array2
16 |     mov ebx, OFFSET Array3
17 | L1:
18 |     call ArraySub
19 |     add edx, TYPE DWORD
20 |     add esi, TYPE DWORD
21 |     mov [ebx], ax
22 |     add ebx, TYPE DWORD
23 |     loop L1
24 |
25 |     call crlf
26 |     mov edx, OFFSET a1message
27 |     call WriteString
28 |     call crlf
29 |     movzx eax, [Array1 + 0]
30 |     call WriteHex
31 |     call crlf
32 |     movzx eax, [Array1 + 2]
33 |     call WriteHex
34 |     call crlf
35 |     movzx eax, [Array1 + 4]
36 |     call WriteHex
37 |     call crlf
38 |     movzx eax, [Array1 + 6]
39 |     call WriteHex
40 |     call crlf
41 |     movzx eax, [Array1 + 8]
42 |     call WriteHex
43 |     call crlf
44 |     movzx eax, [Array1 + 10]
45 |     call WriteHex
46 |     call crlf
47 |     movzx eax, [Array1 + 12]
48 |     call WriteHex
49 |     call crlf
50 |     movzx eax, [Array1 + 14]
51 |     call WriteHex

```



```

52     call crlf
53     movzx eax, [Array1 + 16]
54     call WriteHex
55     call crlf
56     movzx eax, [Array1 + 18]
57     call WriteHex
58     call crlf
59     movzx eax, [Array1 + 20]
60     call WriteHex
61     call crlf
62
63     call crlf
64     mov edx, OFFSET a2message
65     call WriteString
66     call crlf
67     movzx eax, [Array2 + 0]
68     call WriteHex
69     call crlf
70     movzx eax, [Array2 + 2]
71     call WriteHex
72     call crlf
73     movzx eax, [Array2 + 4]
74     call WriteHex
75     call crlf
76     movzx eax, [Array2 + 6]
77     call WriteHex
78     call crlf
79     movzx eax, [Array2 + 8]
80     call WriteHex
81     call crlf
82     movzx eax, [Array2 + 10]
83     call WriteHex
84     call crlf
85     movzx eax, [Array2 + 12]
86     call WriteHex
87     call crlf
88     movzx eax, [Array2 + 14]
89     call WriteHex
90     call crlf
91     movzx eax, [Array2 + 16]
92     call WriteHex
93     call crlf
94     movzx eax, [Array2 + 18]
95     call WriteHex
96     call crlf
97     movzx eax, [Array2 + 20]
98     call WriteHex
99     call crlf
100
101     call crlf
102

```

```

102     mov edx, OFFSET a3message
103     call WriteString
104     call crlf
105     movzx eax, [Array3 + 0]
106     call WriteHex
107     call crlf
108     movzx eax, [Array3 + 4]
109     call WriteHex
110     call crlf
111     movzx eax, [Array3 + 8]
112     call WriteHex
113     call crlf
114     movzx eax, [Array3 + 12]
115     call WriteHex
116     call crlf
117     movzx eax, [Array3 + 16]
118     call WriteHex
119     call crlf
120     movzx eax, [Array3 + 20]
121     call WriteHex
122     call crlf
123
124     exit
125     main ENDP
126
127     ArraySub PROC USES esi
128     mov eax, 0
129     add eax, [esi]
130     sub eax, [edx]
131
132     ret
133     ArraySub ENDP
134
135     END main

```

```

Array1 Values:
00000007
00000008
00000007
00000006
00000005
00000004
00000003
00000002
00000001
00000000
00000001

Array2 Values:
00000004
00000002
00000003
00000004
00000005
00000006
00000007
00000008
00000009
00000008
00000007

Array3 Values:
00000003
00000004
00000000
0000FFFC
0000FFFF
00007241

```

4. **[Loops]** Write a program to compute the sum of first n integers of the series: $Sum = 1 + 2 - 3 + 4 - 5 + 6 - 7 + 8 - 9 \dots$ Your program must:
 - a. Prompt user for integer n ,
 - b. Read the value of n from user input
 - c. Calculate Sum , and;
 - d. Print Sum on screen.

Please use the “WriteInt” procedure, not “DumpRegs”. Other relevant procedures: “ReadInt” and “WriteString.” The calculation can be done in many ways, and all submissions that evidence proper programming practice are acceptable. In your homework submission, please embed both the code and one screen shot for $n = 10$.

```

1  TITLE Program - Cameron Mathis (Program.asm)
2  INCLUDE Irvine32.inc
3
4  .data
5  message1 BYTE "Please enter your desired number: ",0
6  message2 BYTE "Your output is: ",0
7  intVal SDWORD ?
8
9  .code
10 main PROC
11     mov edx, OFFSET message1
12     call WriteString
13     call ReadInt
14     mov intVal, eax
15     call crlf
16     mov ecx, intVal
17     mov eax, 0
18
19     L1:
20         add eax, ecx
21         dec ecx
22         .if ecx == 2
23             dec ecx
24         .endif
25         loop L1
26         mov ecx, intVal
27
28     L2:
29         dec ecx
30         sub eax, ecx
31         .if ecx == 3
32             dec ecx
33             dec ecx
34         .endif
35         loop L2
36
37         inc eax
38
39         mov edx, OFFSET message2
40         call WriteString
41         call WriteInt
42         call crlf
43
44     exit
45 main ENDP
46 END main

```

Microsoft Visual Studio Debug Console

Please enter your desired number: 10

Your output is: +7