

CSC 3210
Computer Organization and Programming
Lab 9
Answer Sheet

Student Name: Vivian Do
Section: CRN 90913; 11:00-12:40
Oct 22 2021

Lab 9(a)

Debug through each line of instructions.
Take screenshot that includes code and register window.
Record the register content.
and explain the register contents.

Code:

```
lab9a.asm
1  ; Lab 9a
2  .386
3  .model flat, stdcall
4  .stack 4096
5  ExitProcess proto, dwExitCode:dword
6
7  .data
8      intarray DWORD 10000h, 20000h, 30000h, 40000h
9
10 .code
11 main proc
12     mov edi, OFFSET intarray      ; EDI = address of intarray
13     mov ecx, LENGTHOF intarray    ; initialize loop counter
14     mov eax, 0                    ; sum = 0
15     L1:                            ; mark beginning of loop
16     add eax, [edi]                ; add an integer
17     add edi, TYPE intarray         ; point to next element
18     loop L1                       ; repeat until ECX = 0
19     invoke ExitProcess, 0
20
21 main endp
22 end main
23
```

Line number: 12

Instruction: `mov edi, OFFSET intarray`

Register Values: `EDI = 00C74000`

Screenshot:

```
Registers
EAX = 004FFCD4 EBX = 00269000 ECX = 00C71005 EDX = 00C71005 ESI = 00C71005 EDI = 00C74000 EIP = 00C71015 ESP = 004FFC7C EBP = 004FFC88 EFL = 00000246
OV = 0 UP = 0 EI = 1 PL = 0 ZR = 1 AC = 0 PE = 1 CY = 0
```

Explanation: setting the number address of intarray (00C74000)

(Copy this format as needed)

Line number: 13

Instruction: `mov ecx, LENGTHOF intarray`

Register Values: ECX = 00000004

Screenshot:

```
Registers
EAX = 004FFCD4 EBX = 00269000 ECX = 00000004 EDX = 00C71005 ESI = 00C71005 EDI = 00C74000 EIP = 00C7101A ESP = 004FFC7C EBP = 004FFC88 EFL = 00000246

OV = 0 UP = 0 EI = 1 PL = 0 ZR = 1 AC = 0 PE = 1 CY = 0
```

Explanation: ECX register is updated, returning the number of elements in intarray (4)

Line number: 14

Instruction: mov eax, 0

Register Values: EAX = 00000000

Screenshot:

```
Registers
EAX = 00000000 EBX = 00269000 ECX = 00000004 EDX = 00C71005 ESI = 00C71005 EDI = 00C74000 EIP = 00C7101F ESP = 004FFC7C EBP = 004FFC88 EFL = 00000246

OV = 0 UP = 0 EI = 1 PL = 0 ZR = 1 AC = 0 PE = 1 CY = 0

0x00C74000 = 00010000
```

Explanation: EAX register has been zero'd in order to initialize the sum

Line number: 16

Instruction: add eax, [edi]

Register Values: EAX = 00010000

Screenshot:

```
Registers
EAX = 00010000 EBX = 00E84000 ECX = 00000004 EDX = 00C71005 ESI = 00C71005 EDI = 00C74000 EIP = 00C71021 ESP = 00C1FC48 EBP = 00C1FC54 EFL = 00000206

OV = 0 UP = 0 EI = 1 PL = 0 ZR = 0 AC = 0 PE = 1 CY = 0
```

Explanation: EAX register is updated, adding the first value of intarray (1000h)

Line number: 17

Instruction: add edi, TYPE intarray

Register Values: EDI = 00C74004

Screenshot:

```
Registers
EAX = 00010000 EBX = 00E84000 ECX = 00000004 EDX = 00C71005 ESI = 00C71005 EDI = 00C74004 EIP = 00C71024 ESP = 00C1FC48 EBP = 00C1FC54 EFL = 00000202

OV = 0 UP = 0 EI = 1 PL = 0 ZR = 0 AC = 0 PE = 0 CY = 0
```

Explanation: EDI register is updated by pointing to the next element in intarray (4)

Line number: 18

Instruction: loop L1

Register Values: ECX = 00000003

Screenshot:

```
Registers
EAX = 00010000 EBX = 00E84000 ECX = 00000003 EDX = 00C71005 ESI = 00C71005 EDI = 00C74004 EIP = 00C7101F ESP = 00C1FC48 EBP = 00C1FC54 EFL = 00000202

OV = 0 UP = 0 EI = 1 PL = 0 ZR = 0 AC = 0 PE = 0 CY = 0

0x00C74004 = 00020000 |
```

Explanation: ECX is updated with the next iteration

Since lines 16, 17, and 18 are in the loop L1, it will continue to add the next value to EAX from EDI and point to the next element in intarray

- EAX = 00030000 EDI = 00C74008 ECX = 00000002
- EAX = 00060000 EDI = 00C7400C ECX = 00000001
- EAX = 000A0000 EDI = 00C74010 ECX = 00000000

End Results:

```
Registers
EAX = 000A0000 EBX = 00E84000 ECX = 00000000 EDX = 00C71005 ESI = 00C71005 EDI = 00C74010 EIP = 00C71026 ESP = 00C1FC48 EBP = 00C1FC54 EFL = 00000212

OV = 0 UP = 0 EI = 1 PL = 0 ZR = 0 AC = 1 PE = 0 CY = 0

|
```

EAX = 000A0000
ECX = 00000000

Lab 9(b)

Debug until you reach “INVOKE ExitProcess, 0”.

Take a screenshot of the code and register window at the end

Record the content of the EAX register

Then explain the register content.

EAX Register value: EAX = 0000001C

Screenshot:

The screenshot shows a debugger window with two panes. The top pane, titled 'Registers', displays the current state of the CPU registers. The bottom pane shows the assembly code for a program named 'lab9b.asm'. The code includes a main procedure with two nested loops, L1 and L2. The L1 loop decrements ECX from 10 to 0, and the L2 loop increments EAX by 5 and decrements ECX from 5 to 0. The program then calls 'invoke ExitProcess, 0'.

```
Registers
EAX = 0000001C EBX = 0115F000 ECX = 00000000 EDX = 00FD100A ESI = 00FD100A EDI = 00FD100A EIP = 00FD1067 ESP = 00F4FF20 EBP = 00F4FF2C EFL = 00000202
OV = 0 UP = 0 EI = 1 PL = 0 ZR = 0 AC = 0 PE = 0 CY = 0

100 %
Memory 1
lab9b.asm
1 ; Lab 9b
2 .386
3 .model flat, stdcall
4 .stack 4096
5 ExitProcess proto, dwExitCode:dword
6
7 .data
8     temp DWORD ?
9
10 .code
11 main proc
12     mov eax, 0
13     mov ecx, 10      ; outer loop counter
14     L1:
15         mov eax, 3
16         mov temp, ecx
17         mov ecx, 5    ; inner loop counter
18         L2:
19             add eax, 5
20             loop L2    ; repeat inner loop
21             mov ecx, temp
22             loop L1    ; repeat outer loop
23     invoke ExitProcess, 0 ; 51ms elapsed
24
25 main endp
26 end main
27
```

Explanation: This program never stops running. The results of the outer loop (L1) decrements ECX to zero and the inner loop (L2) decrements ECX to FFFFFFFFh. This will then cause the outer loop (L1) to repeat again, hence the program continuously running.

Lab 9(c)

Debug through each line of instructions.
Take screenshot that includes code and register window.
Record the register content.
and explain the register contents.

Code:

```

lab9c.asm  lab9b.asm
1  ; Lab 9c
2  .386
3  .model flat, stdcall
4  .stack 4096
5  ExitProcess proto, dwExitCode:dword
6
7  .code
8  main proc
9      mov al, 01101111b
10     and al, 00101101b      ; a. AL =
11     mov al, 6Dh
12     and al, 4Ah           ; b. AL =
13     mov al, 00001111b
14     or al, 61h            ; c. AL =
15     mov al, 94h
16     xor al, 37h           ; d. AL =
17     invoke ExitProcess, 0
18
19 main endp
20 end main
21

```

Line number: 9

Instruction: `mov al, 01101111b`

Register Values: `EAX = 0097FA6F`

Screenshot:

```

Registers
EAX = 0097FA6F EBX = 006F8000 ECX = 002E100F EDX = 002E100F ESI = 002E100F EDI = 002E100F EIP = 002E1086 ESP = 0097FA60 EBP = 0097FA6C EFL = 00000246
OV = 0 UP = 0 EI = 1 PL = 0 ZR = 1 AC = 0 PE = 1 CY = 0

```

Explanation: AL in EAX register is updated with 01101111b (6Fh)

(Copy this format as needed)

Line number: 10

Instruction: `and al, 00101101b`

Register Values: `EAX = 0097FA2D`

Screenshot:

```

Registers
EAX = 0097FA2D EBX = 006F8000 ECX = 002E100F EDX = 002E100F ESI = 002E100F EDI = 002E100F EIP = 002E1088 ESP = 0097FA60 EBP = 0097FA6C EFL = 00000206
OV = 0 UP = 0 EI = 1 PL = 0 ZR = 0 AC = 0 PE = 1 CY = 0

```

Explanation: AL in EAX register is updated with 2Dh (00101101b).
 $01101111b (6Fh) \wedge 00101101b (2Dh) \rightarrow 00101101b (2Dh)$

Truth Table		
6F	2D	6F ^ 2D
0	0	0
1	0	0
1	1	1
0	0	0
1	1	1
1	1	1
1	0	0
1	1	1

Line number: 11

Instruction: `mov al, 6Dh`

Register Values: `EAX = 0097FA6D`

Screenshot:

```
Registers
EAX = 0097FA6D EBX = 006F8000 ECX = 002E100F EDX = 002E100F ESI = 002E100F EDI = 002E100F EIP = 002E108A ESP = 0097FA60 EBP = 0097FA6C EFL = 00000206

OV = 0 UP = 0 EI = 1 PL = 0 ZR = 0 AC = 0 PE = 1 CY = 0
```

Explanation: AL in EAX register is updated with 6Dh

Line number: 12
 Instruction: and al, 4Ah
 Register Values: EAX = 0097FA48
 Screenshot:

```
Registers
EAX = 0097FA48 EBX = 006F8000 ECX = 002E100F EDX = 002E100F ESI = 002E100F EDI = 002E100F EIP = 002E108C ESP = 0097FA60 EBP = 0097FA6C EFL = 00000206

OV = 0 UP = 0 EI = 1 PL = 0 ZR = 0 AC = 0 PE = 1 CY = 0
```

Explanation: AL in EAX register is updated with 48h (01001000b)
 6Dh (01101101b) ^ 4Ah (01001010b) -> 48h (01001000)

Truth Table		
6D	4A	6D ^ 4A
0	0	0
1	1	1
1	0	0
0	0	0
1	1	1
1	0	0
0	1	0
1	0	0

Line number: 13
 Instruction: mov al, 00001111b
 Register Values: EAX = 0097FA0F
 Screenshot:

```
Registers
EAX = 0097FA0F EBX = 006F8000 ECX = 002E100F EDX = 002E100F ESI = 002E100F EDI = 002E100F EIP = 002E108E ESP = 0097FA60 EBP = 0097FA6C EFL = 00000206

OV = 0 UP = 0 EI = 1 PL = 0 ZR = 0 AC = 0 PE = 1 CY = 0
```

Explanation: AL in EAX register is updated with 00001111b (0Fh)

Line number: 14
 Instruction: or al, 61h
 Register Values: EAX = 0097FA6F
 Screenshot:

```
Registers
EAX = 0097FA6F EBX = 006F8000 ECX = 002E100F EDX = 002E100F ESI = 002E100F EDI = 002E100F EIP = 002E1090 ESP = 0097FA60 EBP = 0097FA6C EFL = 00000206

OV = 0 UP = 0 EI = 1 PL = 0 ZR = 0 AC = 0 PE = 1 CY = 0
```

Explanation: AL in EAX register is updated with 6Fh (01101111b)
 0Fh (00001111b) v 61h (01100001b) -> 6Fh (01101111b)

Truth Table		
0F	61	0F v 61
0	0	0
0	1	1
0	1	1
0	0	0
1	0	1
1	0	1
1	0	1
1	1	1

Line number: 15
 Instruction: mov al, 94h
 Register Values: EAX = 0097FA94

Screenshot:

```
Registers
EAX = 0097FA94 EBX = 006F8000 ECX = 002E100F EDX = 002E100F ESI = 002E100F EDI = 002E100F EIP = 002E1092 ESP = 0097FA60 EBP = 0097FA6C EFL = 00000206

OV = 0 UP = 0 EI = 1 PL = 0 ZR = 0 AC = 0 PE = 1 CY = 0

|
```

Explanation: AL in EAX register is updated with 94h

Line number: 16

Instruction: xor al, 37h

Register Values: EAX = 0097FAA3

Screenshot:

```
Registers
EAX = 0097FAA3 EBX = 006F8000 ECX = 002E100F EDX = 002E100F ESI = 002E100F EDI = 002E100F EIP = 002E1094 ESP = 0097FA60 EBP = 0097FA6C EFL = 00000206

OV = 0 UP = 0 EI = 1 PL = 1 ZR = 0 AC = 0 PE = 1 CY = 0

|
```

Explanation: AL in EAX register is updated with A3h (10100011b)
94h (10010100b) xor 37h (00110111b) -> A3h (10100011b)

Truth Table		
94	27	94 XOR 37
1	0	1
0	0	0
0	1	1
1	1	0
0	0	0
1	1	0
0	1	1
0	1	1