CSC 3210

Computer Organization and Programming

Lab 6

Answer Sheet

Student Name: **Asrar Syed**

Section: **022 / CRN: 17915**

**Lab 5(a)**

Debug through each line of your code.

* Execute the instruction
* Take a screenshot of the code and register window
* Record the line number, instruction, Register values in the answer sheet.
* Also add the screenshot
* Then explain the register contents.

**End Debug (zoom in)**

A screenshot of a computer

Description automatically generated

**Line number: 14**

**Instruction: mov bx, 0A69Bh**

**Register value: EBX = 0116A69B, EIP = 00F71014**

**Explanation: bx is 16-bit register, and the given data is 20 bits, so only 16 bits LSB's are moved to bx. So bx contains A69Bh in hexadecimal form.**

**Line number: 15**

**Instruction: movzx eax, bx**

**Register value: EAX = 0000A69B, EIP = 00F71017**

**Explanation: movzx is moving the data of bx to eax with 32 bits by extending the bits from 16-32 by 0 because we use the zero extension here.**

**Line number: 17**

**Instruction: movzx eax, myByte1**

**Register value: EAX = 0000009B, EIP = 00F7101E**

**Explanation: This instruction moved the data of Byte1 i.e., 9Bh to eax but with extension to 32 bits with zero.**

**Line number: 19**

**Instruction: mov bx, 0A69Bh**

**Register value: EIP = 00F71022**

**Explanation: This is the same as Line number 14:**

**Line number: 20**

**Instruction: movsx eax, bx**

**Register value: EAX = FFFFA69B, EIP = 00F71025**

**Explanation: The instruction movsx is used that extends the bits to 32 by 1 as unsigned number.**

**Lab 5(b)**

Debug through each line of your code.

* Execute the instruction
* Take a screenshot of the code and register window
* Record the line number, instruction, Register values in the answer sheet.
* Also add the screenshot
* Then explain the register contents.

**End Debug (zoom in)**

**A screenshot of a computer

Description automatically generated**

**Line number: 17**

**Instruction: mov eax, arrayD**

**Register value: EAX = 00010000, EIP = 00AC1015**

**Explanation: EAX register is 32-bit long with a signed integer variable. This register is updated with arrayD (10000h) in its contents by mov.**

**Line number: 18**

**Instruction: mov ebx, [arrayD+4]**

**Register value: EBX = 00020000, EIP = 00AC101B**

**Explanation: EBX register is 32-bit long with an unsigned integer variable. This register is updated with 200000h. By adding 4 to arrayD, we are accessing the array element that is 4 bytes offset to the first one. This is the 2nd element of arrayD.**

**Line number: 19**

**Instruction: mov edx, [arrayD+8]**

**Register value: EDX = 00030000, EIP = 00AC1021**

**Explanation: EDX register is updated with 300000h. Similar to line 18 this is a unsigned integer variable. By adding 8 to arrayD, we are accessing the array element that is 8 bytes away from the first byte. This is the 3rd element of arrayD.**

**Lab 5(c)**

Debug through each line of your code.

* Create a new project to run the following program
* Declare an array:
* arrayB WORD 1,2,3,4
* Write code to Rearrange the array as follows:
* 3,4,2,1
* Debug the code until you reach “INVOKE ExitProcess, 0”
* Add the screenshot of your code in the answer sheet.

**Start Debug**

**A screenshot of a computer

Description automatically generated**

**I HAVE A M2 MACBOOK SO I CAN”T CHECK**

**I CHANGED THE CODE (Look next Page)**

**Ok this revised code (simply explained)**

; Asrar Syed

; Lab 6 (Problem 3) - 6(c)

.386

.model flat, stdcall

.stack 4096

ExitProcess proto dwExitCode:dword

.data

arrayB WORD 1,2,3,4

; Change the array to 3,4,2,1

.code

main proc

mov ax, arrayB

xchg ax, [arrayB + 6]

xchg ax, [arrayB + 2]

xchg ax, [arrayB + 4]

mov arrayB, ax

invoke ExitProcess, 0

main endp

end main

**WORD is a 16-bit unsigned integer**

**First line moves first part of arrayB**

**Into the ax register**

**+6 is 4th element of array**

**+2 is 2nd element of array**

**+4 is 3rd element of array**

**Last line moves whatever is in ax**

**Back into the first part of arrayB**