CSC 3210 Computer Organization and Programming Lab 5 Answer Sheet

Student Name: **Hemant Kosaraju**

Section: **020**

Lab 5(a): Create a project and generate listing file (3 points)

screenshot of the listing file content for the program (slide 38):

mov 20 to AL register

mov 100 to AX register

mov 1000 to EAX register

add 2 to the content of EAX register

subtract 20 from al register

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Lab 5(b): Write a program (slide 55) to evaluate an expression with variables (3 Points)

Debug the code until the ‘invoke ExitProcess, 0’.

Attach screenshot showing the content of AX register at the end along with the code

Line number: **26**

Instruction: **mov ebx, Yval**

Register value: **EBX = 0000001E**

Explanation: **The Yval variable was initialized in the .data directive and was assigned the value 30, this is shown by 0000001E because 1 ∗** **16 = 16 + 14 [E] = 30 which is moved to EBX register**

**A screenshot of a computer

Description automatically generated**

Line number: **27**

Instruction: **add ebx, Zval**

Register value: **EBX = 00000046**

Explanation: **The instruction on this line adds the Zval variable also declared in the .data directive and was assigned the value 40, and when doing 30 + 40 = 70 we can see why the EBX register has the value 00000046 because 4 ∗ 16 = 64 + 6 = 70 which outputs after adding Zval [40] to Yval [30]**

**A screenshot of a computer

Description automatically generated**

Line number: **28**

Instruction: **sub Xval, ebx**

Register value: **This Xval memory address is where the signed value is stored 0x008A4000 = FFFFFFD4**

Explanation: **This instruction line is subtracted the EBX register of 00000046 [70] from the Xval variable declared and initialized in the .data directive as 26 and because of this the calculation turns into negative due to 26 – 70 = -44 and because of this the bits of the register are flipped from being 6 zeros to 6 Fs which is the maximum single bit hexadecimal storage**

**A screenshot of a computer

Description automatically generated**

Line number: **29**

Instruction: **mov eax, Xval**

Register value: **EAX = FFFFFFD4**

Explanation: **The flipped value previously stored in the Xval memory address at 0x008A4000 created for the signed value has been moved by this instruction to the EAX register = FFFFFFD4**

A screenshot of a computer

Description automatically generated

(copy the above format if you need more )

Lab 5(c): Write a program (slide 59) to see the data items in an array (2 points)

Debug the code until the ‘invoke ExitProcess, 0’.

Attach screenshot showing the content of AX register at the end along with the code

Line number: **16**

Instruction: **mov EAX, 0**

Register value: **EAX = 00000000**

Explanation: **The instruction on this line tells the assembler to move the value of 0 to the EAX register which sets the EAX register to include 8 zeros**

A screenshot of a computer

Description automatically generated

Line number: **17**

Instruction: **mov AX, myWord + 0**

Register value: **EAX = 00000001**

Explanation: **This is because myWord variable was assigned to 4 DUP (1,2,3,4,5) which would be interpreted as 1,2,3,4,5,1,2,3,4,5,1,2,3,4,5,1,2,3,4,5 and so the mov AX, myWord + 0 gets the first zero 0 byte value from the list which is 1 and moves it to the EAX register which is why the register value was 00000001**

**A screenshot of a computer

Description automatically generated**

Line number: **18**

Instruction: **add AX, myWord + 2**

Register value: **EAX = 00000003**

Explanation: **For this instruction the existing EAX register value was 00000001, however now from the myWord + 2 instruction do 2/2 Bytes and it would be the first index two byte value from the list that is 2 which is why 00000001 + 00000002 = EAX = 00000003**

**A screenshot of a computer

Description automatically generated**

Line number: **19**

Instruction: **add AX, myWord + 4**

Register value: **EAX = 00000006**

Explanation: **For this instruction the existing EAX register value was 00000003, however now from the myWord + 4 instruction do 4/2 Bytes and it would be the second two byte value from the list that is 3 which is why 00000003 + 00000003 = EAX = 00000006**

**A screenshot of a computer

Description automatically generated**

Line number: **20**

Instruction: **add AX, myWord + 6**

Register value: **EAX = 0000000A**

Explanation: **For this instruction the existing EAX register value was 00000006, however now from the myWord + 6 instruction do 6/2 Bytes and it would be the third two byte value from the list that is 4 which is why 00000006 + 00000004 = EAX = 0000000A [10]**

A screenshot of a computer

Description automatically generated

Line number: **21**

Instruction: **add AX, myWord + 8**

Register value: **EAX = 0000000F**

Explanation: **For this instruction the existing EAX register value was 0000000A, however now from the myWord + 8 instruction do 8/2 Bytes and it would be the fourth two byte value from the list which is 5 which is why 0000000A + 00000005 = 0000000F [15]**

**A screenshot of a computer

Description automatically generated**