National University of Computer and Emerging Sciences, Lahore Campus

Course:	**COAL**	**Course Code:**	**EE2003**
Program:	**BS(CS,DS)**	**Semester:**	**Fall 2022**
Duration:	**1 Hour**	**Total Marks:**	**30**
Paper Date:	**27-09-2022**	**Page(s):**	**5**
Section:	**All**	**Roll No.**	
Exam:	**Midterm-I**	**Your Section:**	

Instruction/Notes: **This is an open notes/book exam. Sharing notes and calculators is NOT ALLOWED.** All the answers should be written in provided space on this paper. Rough sheets can be used but will not be collected and checked. In case of any ambiguity, make reasonable assumptions. Questions during exams are not allowed.

Question 1 [CLO 1] [3x5 = 15 Marks]: Short questions

i) For each of the instructions given below, **identify whether the instruction is valid or invalid:**

Instruction	Valid/Invalid
mov [Bp-si],2	Invalid
mov [bl], 2	Invalid
mov word[bx],2	Valid
mov [bp+bx+si],2	Invalid
mov word[bx+ax],3	Invalid
mov word[num1], word[num2]	Invalid

ii) Given the following jump statements and their opcodes, **identify the type of jump (near or short) and the offset (value of IP) to which the jump will take place.** Show your complete working to get credit**.**

Offset of Opcode	Opcode	Type of Jump	Offset
011D	76E1 ;76 is opcode of jump	short	0x0100
0100	E9FF1E ; E9 is opcode of jump	near	0x2002

Show your working here:

(1) 0x011D + 0x2 = 0x011F

0xE1 (two's complement) = 0x1F

0x011F – 0x1F = 0x0100

(2) 0x0100 + 0x3 = 0x0103

0x0103 + 0x1EFF (little endian) = 0x2002

iii) Given the following values of cs, ds, ss, bp, bx, si and di in **hexadecimal**. **What would be the value of ax register after executing the instruction given below?** Show your working to get full marks.
CS: 0x1E0A, DS: 0x1EED, SS: 0xFFEF, BP: 0x011F, SI: 0x0114, DI: 0x0112, BX: 0x0115

Instruction: Mov ax, [bp+si+15]

Value of AX after Execution of above instruction: _____0009_ (little endian) _____

Physical Memory Addresses	Memory Content
0x1F112	0A00
0x1E2E2	0500
0x00132	0900
0x00138	0700
0x1F118	0600

Show your working here:

Offset = bp+si+0xF (hex value of 15)

0x11F + 0x0114 + 0xF = 0x0242

Segment to be used = stack segment (ss) = 0xFFEF

Physical address = segment * 0x10 + offset = 0xFFEF * 0x10 + 0x0242 = 0x00132

Ax = 0009 (little endian)

iv) Write assembly language code that checks if the contents of two registers **AX** and **BX** are equal and jumps to LabelXYZ (if contents are equal). You are NOT ALLOWED to use ***cmp, sub*** and ***sbb*** instructions. No credit will be given if your code exceeds two lines.

```
xor ax, bx

jz LabelXYZ
```

v) For the code snippet given below, write the byte sized data value that is stored in memory label lab2 after the execution of the program. Briefly explain the working of this program.

Value of Lab2 after execution of program: ____-1_____

Write values of following flags after instruction "cmp dl, [bx+lab1]" (highlighted in code) for first iteration only.

SF = ____1_____ OF= _____0_____ CF= _____1_____ ZF=_____0_____

Code Snippet:	Show your working here (for flag values):

```
[org 0x100]
        jmp start

lab1:   db -3, -1, 4
lab2:   db 0

start:  mov bx, 1
        mov dl, [lab1]

loop1:  cmp dl, [bx+lab1]
        jae C1

        mov dl, [bx+lab1]

C1:     add bx, 1
        cmp bx, 3
        jne loop1

        mov [lab2], dl

        mov ax, 0x4c00
        int 0x21
```

What is above code doing? Describe in one line only.

Negative numbers are stored in an array, but we are using the jump instruction (jae), which is valid for the unsigned numbers. This program finds out the maximum unsigned number that is stored in the memory array lab1. In unsigned format, -1 has the maximum value (i.e. 0xFF). Therefore, -1 is stored in memory label lab2, after the execution of the program.

Question 2 [CLO 3] [15 Marks]: Difference of two sets (Set1 – Set2) is a set having elements of Set 1 which are NOT Present in Set 2, see following examples for detail**. Your task is to write a program in Assembly Language that finds Difference of two sets (Set1-Set2).** Note that both the sets are sorted and have distinct elements only.

Example 1	Example 2
Set1: -3, -1, 2, 5, 6, 8, 9 Set2: -2, 2, 6, 7, 9 Difference: -3, -1, 5, 8	Set1: -3, -1, 2, 5, 6, 8, 9 Set2: 1, 3, 7 Difference: -3, -1, 2, 5, 6, 8, 9

Solution:

```
[org 0x100]
                    jmp start
sizeSet1:           db 7 ; There are 7 elements in Set 1
sizeSet2:           db 5 ; Set 2 contains 5 Elements
sizeSetDiff:        db 7 ; Maximum 7 elements' space
Set1:               db -3, -1, 2, 5, 6, 8, 9 ; Elements of Set1
Set2:              db -2, 2, 6, 7, 9; Elements of Set2
Difference:         db 0, 0, 0, 0, 0, 0,0 ; Max 7 elements' space available
start:

 mov bx, 0

mov di, 0

mov ch, 0

mov cl, [sizeSet1]

mov dh, 0

mov dl, [sizeSet2]

outerloop: mov al, [Set1+bx]

mov si, 0

innerloop: cmp al, [Set2+si]

je next

add si,1

cmp si, dx

jne innerloop

mov [Difference + di], ax

add di, 1

next: add bx, 1

cmp bx, cx

jne outerloop

mov ax, 0x4C00

int 21h
```