National University of Computer and Emerging Sciences, Lahore Campus

Course:	COAL	Course Code:	EE2003
Program:	BSCS,BSDS,BSR	Semester:	Fall 2023
Duration:	1 Hour	Total Marks:	30
Paper Date:	08-Nov-2023	Page(s):	5
Section:	All	Roll No.	
Exam:	Midterm – II		

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 2] [15 marks]: Answer the following short questions.

(i) **[3 marks]** What will be printed on display memory after the execution of following piece of code? Also tell the color of the printed character.

```
[org 0x0100]
mov ax, 0xb800
mov es, ax; point es to video base
mov byte[es:0], 0x31
mov byte[es:1], 01
mov ax, 0x4c00
int 0x21
```

3

Character: 1

Color: 01 in Hex is 0x01 which is for blue text on black back so color is blue.

(ii) **[2 marks]:** Will this code properly clear the stack? Answer in only Yes/No.

```
[org 0x100]
Mov ax,0x100
push ax
Call Done
Mov ax, 0x4c00
int 0x21
Done:
Add sp,1
Ret 1
```

Show your working here:

ax = 0x100

FFFC	ax
FFFA	Address

Answer: Yes

2

(iii) **[5 marks]:** The following subroutine max takes 3 numbers as parameters and returns the largest number through stack.

```
[org 0x100]
jmp start
max:
push bp
mov bp, sp
sub sp,2
push ax
push cx
mov ax, [bp+4]
cmp ax, [bp+6]
ja next
mov ax, [bp+6]
next:mov [bp-2],ax
mov cx, [bp+8]
cmp cx, [bp-2]
ja next1
mov cx,[bp-2]
next1: mov [bp+8],cx
pop cx
pop ax
mov sp, bp
pop bp
ret 6
```

```
start:
sub sp,2
push 8
push 5
push 6
call max
pop dx
mov ax, 0x4C00
int 21h
```

Show your working here:

ax = 6
cmp ax, 5
L2 = 6
cx = 8
cmp cx, 6
bp+8, 8

pop cx
pop ax

dx = Empty

Stack is still not empty because of L1

FC	+10	Local space 1
FA	+8	8
F8	+6	5
F6	+4	6
F4	bp+2	Address
F2	Sp → bp+0	bp
F0	−2	Local space 2
F	−4	ax
F	−6	cx

Assuming that the initial value of sp is 0xFFFE, answer the following questions.

(a) What is the value of sp after

1. call max instruction is executed? 0xFFF4 [1]
2. ret 6 instruction is executed? 0xFFFC [1]

(b) What is the value of bp after mov bp, sp instruction is executed? 0xFFF2 [1]

3

(c) It is noticed that after the execution of the subroutine max, correct return value is not popped in dx register. Identify and correct the error. After ret 6 the stack has Local 1 and Local 2. Local 1 is empty while L2 has value of 6. We should again use pop dx to get desired value. [2]

0501

(iv) **[5 marks]:** Write the contents of "str1" after executing the following program. Show complete working to get marks.

```
[org 0x0100]
mov si, str2
mov di, str1
mov cx, [len]
add di, cx
dec di
push ds
pop es
cld
next:
movsb
sub di, 2
loop next
mov ax, 0x4c00
int 0x21
len: dw 17
str1: db 'Never odd or even'....
str2: db 'Final Examination'
```

Show your working here:

si = str2
di = str1
cx = len = 17
di ①
di = str1 + 17 ds = es
[es:di] = [ds:si]
[es:di]
[ds:si]

Never odd or even
Final Examination

str1: Never odd or even Final Examination

Question 2 [CLO 3] [15 Marks]: Write a function **RotateScreenRight** that rotates a 5x5 square portion of the display screen towards right. Following sample run shows the screen before and after the function call, a 5x5 square portion (highlighted with bold letters) of screen starting from top-left point (2,3) i.e. 3rd cell of 2nd row has been rotated towards right. Your function should take two parameters: RowNo and ColNo. Following call is with parameters RowNo=2 and ColNo = 3.
Credit will be given for using String instructions where required. Properly call this function in your program. Safely assume that dimensions of square will always fit in the display memory (of 25x80 cells that we did in class).

Sample Run: (This sample is for a display memory of 8x10 grid.)

a	b	c	d	e	f	g	h	i	J
k	l	m	n	o	p	q	r	s	t
u	v	w	x	y	z	A	B	C	D
E	F	G	H	I	J	K	L	M	N
O	P	Q	R	S	T	U	V	W	X
Y	Z	0	1	2	3	4	5	6	7
8	9	!	@	#	$	%	^	&	*
(	)	_	+	=	-	[	]	{	}

Screen Before Function Call

a	b	c	d	e	f	g	h	i	J
k	l	m	n	o	p	q	r	s	t
u	v	w	@	1	R	H	x	C	D
E	F	G	#	2	S	I	y	M	N
O	P	Q	$	3	T	J	z	W	X
Y	Z	0	%	4	U	K	A	6	7
8	9	!	^	5	V	L	B	&	*
(	)	_	+	=	-	[	]	{	}

Screen After Function Call