EE213 Computer Organization and Assembly Language
Quiz III (GR I) – Spring 2019
Thursday, May 02, 2019
Paper-B-Solution
Student Name: ______________________________ Roll#__________________________

1. Given the following recursive procedure, and that **EAX = 01h, EBP = 100h** and **ESP =1000h**, draw out the whole stack (and stack frames) with addresses, till after func1's first recursive call. No point will be awarded without correct addresses. **[06 points]**

```
main PROC LOCAL X: WORD
0010   PUSH   EBP
0014   MOV    EBP, ESP
0018   MOV    X, 1Fh
001C
0020   INVOKE func1,X,¥
0024   LEAVE
0028   RET
main   ENDP
```

```
func1 PROC, param1:Word USES EAX
0900   ENTER  4, 1
0904   MOV    EAX, 0
0908   MOV    AX, param1
090C   ADD    AX, param1
0910   INC    param1
0914   INVOKE func1, param1
0918   LEAVE
091C   RET
func1  ENDP
```

FFA	1F
FF8	0024
FF4	1000
FF0	
FEB	01
FE7	20
FE5	0918
FE1	FF4
FDE	
FDA	1F

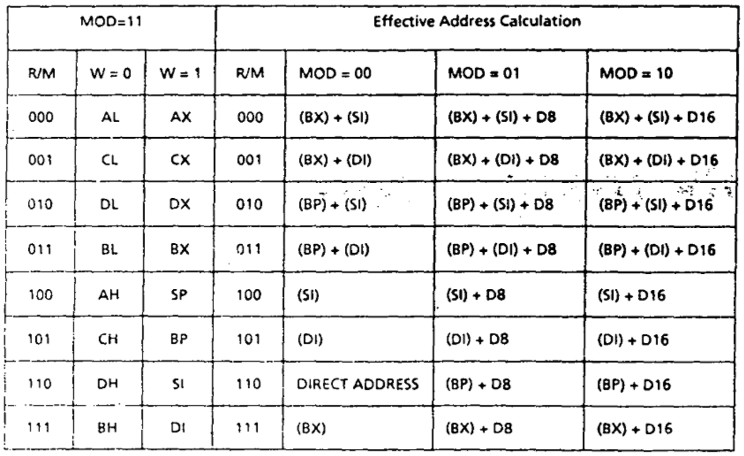
1000	100	;EBP = 1000h now
FFC	1F	;X (local of main)
FFA	1F	;param1 (word)
FF8	0024	;return to main
FF4	1000	EBP PUSHED,EBP=FF4 NOW
FF0		;4-bytes reserved for local data
FEB	01	EAX pushed
FE7	20	;param1(word)
FE5	0918	;return to func1

FE1	FF4	EBP PUSHED,EBP=FE1 NOW
FDE		;4-bytes reserved for local data
FDA	1F	;EAX Pushed

2. Write equivalent x86 assembly PROTOTYPE for the following C++ function: **[02 Points]**
int sample (int, int, int*, char*, short *)

Answer:

```
sample PROTO,var1:DWORD,var2:DWORD,ptr1: PTR DWORD,ptr2:PTR BYTE,ptr2:PTR WORD
```



MOD=11			Effective Address Calculation			
R/M	W = 0	W = 1	R/M	MOD = 00	MOD = 01	MOD = 10
000	AL	AX	000	(BX) + (SI)	(BX) + (SI) + D8	(BX) + (SI) + D16
001	CL	CX	001	(BX) + (DI)	(BX) + (DI) + D8	(BX) + (DI) + D16
010	DL	DX	010	(BP) + (SI)	(BP) + (SI) + D8	(BP) + (SI) + D16
011	BL	BX	011	(BP) + (DI)	(BP) + (DI) + D8	(BP) + (DI) + D16
100	AH	SP	100	(SI)	(SI) + D8	(SI) + D16
101	CH	BP	101	(DI)	(DI) + D8	(DI) + D16
110	DH	SI	110	DIRECT ADDRESS	(BP) + D8	(BP) + D16
111	BH	DI	111	(BX)	(BX) + D8	(BX) + D16

DEC	48h
ADD	0000 00DW (EXT 000)
ADD reg16/mem16, imm16	81h
CMP	0011 10DW (EXT 111)
SUB	1000 00DW (EXT 101)
SUB reg16/mem16, imm16	81h
MOV	1000 10DW (EXT 000)
PUSH reg16/reg32	50h
PUSH mem16/mem32	FFh (EXT 110)

3. Encode the following instructions, provide only the hex-decimal encoded values: **[4 Points]**

1. CMP AL, BL

```
0011 10 0 0     11 011 000
=38 D8h
```

2. MOV [ESI+0FC1],DX

```
1000 10 0 1     10 010 100
=89 94 C1 0Fh
```

3. DEC ESI

```
48 + 6(ESI)
=4Eh
```

4. ADD EDI, 42Fh

```
81 + 7 ← 2F 04 00 00
=88 2F 04 00 00h
```