EE213 Computer Organization and Assembly Language
Quiz III – Spring 2019
Thursday, May 02, 2019
Paper-B-Solution
Student Name: _____ Roll#_____ Section: _____

1. Given the following recursive procedure, and that **EAX = 0F1h, EBP = 7F7Fh** and **ESP = FFFFh**, 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:BYTE, Y:BYTE
       PUSH   EBP
       MOV    EBP, ESP
       MOV    X, 01h
       MOV    Y, 04h
       INVOKE func1,X,Y
       LEAVE
       RET
main   ENDP
```

```
func1 PROC, param1:byte, param2:byte USES EAX
       ENTER  4, 1
       MOV    AL, 0
       MOV    AL, param1
       ADD    param1, AL
       INC    param2
       INVOKE func1, param1, param2
       LEAVE
       RET
func1  ENDP
```

FFFF	7F7F	;Pushed EBP, EBP = FFFF now
FFFB	01	;X (local of main)
FFFA	04	;Y (local of main)
FFF9	01	;param1 (byte)
FFF8	04	;param2 (byte)
FFF7	ret(main)	;return to main
FFF3	FFFF	;EBP =FFF3 now
FFEF		;4-bytes reserved for local data
FFEB	F1	;EAX Pushed
FFE7	02	;param1(byte)
FFE6	05	;param2(byte)
FFE5	ret(func1)	;return to func1
FFE1	FFF3	;EBP Pushed, EBP=FFE1 now
FFDD		;4-bytes reserved for local data
FFD9	01	;EAX Pushed

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

Answer:

```
sample PROTO, ptr1: PTR BYTE, ptr2: PTR DWORD, var1:BYTE, ptr3: PTR DWORD
```

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. **ADD [EBP+ESI], ECX**

   0000 00 0 1 00 001 010
   =01 0Ah

2. **MOV DX, [EBP+ESI+108h]**

   1000 10 1 1 10 010 010 ← 08 01h
   =8B A2 08 01h

3. **DEC ECX**

   48 + 1(ECX)
   =49h

4. **SUB ESP, 04h**

   81h + 04h(ESP) ← 04 00 00 00
   =85 04 00 00 00h