(150 pts) CS3843 Computer Organization Exam #2 Name/abc123:_____
(50 + 5 pts) Part 2

Stack (35 pts)

1. (18 pts) Complete the stack frame given the following assembly code **eip starts at 0x402A45 and ends at 0x4029B4**. All blank spaces have an answer.

   Given: **esp = 0x12904, ebp = 0x12908 when eip = 0x402A45**

```
Func0:
00402A45 8B 55 0C              mov     edx, [ebp+0Ch]
00402A48 52                    push    edx
00402A49 8B 45 08              mov     eax, [ebp+8]
00402A4C 50                    push    eax
00402A4D E8 4E FF FF FF        call    Func1
00402A52 83 C4 08              ____________

Func1:
004029A0 55                    push    ebp
004029A1 8B EC                 mov     ebp, esp
004029A3 83 EC 10              sub     esp, 10h
004029A6 C7 45 F8 00 00 00 00  mov     [ebp-8], FECAADDEh
004029AD C7 45 FC 3F 12 00 00  mov     [ebp-4], 123Fh
004029B4                       mov     [edx],F00D
```

Address (4 pts)	Value (6 pts)	Description (8 pts)
0x12904	**00 00 00 00**	Local variable in Func0
0x12908	**80 29 01 00**	ebp for the function calling Func0
0x1290C	**D3 22 40 00**	Return address of function calling Func0
	F2 92 01 00	
	C4 29 01 00	
	?? ?? ?? ??	Unknown

2. (3 pts) Assuming the local variables are all integers, the instruction at **0x004029A3** reserved space for how many? [ ]

3. (3 pts) What is the value of ebp when eip = 0x004029B4? [ ]

4. (3 pts) For that last instruction in Func1, at what address is "0xF00D" stored?

5. (2 pts) What is the return address for the function calling Func0? Do not show in Little Endian format.

6. (3 pts) What is the address of Func1?

7. (3 pts) What assembly instruction would you expect to find at address **0x00402A52**?

Reading Code (20 pts)

```
int func ( char *name, unsigned int lengthName ){
      int result;
      __asm {
            xor  ecx,ecx            ----------------------------------------------

      next:
            mov  edx,name           ----------------------------------------------

            add  edx,ecx            ----------------------------------------------

            mov  al, byte ptr[edx]  ----------------------------------------------

            test al,al              ----------------------------------------------

            je   done               ----------------------------------------------

            cmp  al,'A'             ----------------------------------------------

            je   done               ----------------------------------------------

            inc  ecx                ----------------------------------------------

            cmp  ecx,lengthName     ----------------------------------------------

            jb   next               ----------------------------------------------

      done:
            mov  result,ecx         ----------------------------------------------
      }
      return result;
}
```

8. (12 pts) Comment each line of code. Write comments that add information to the code and not simply say what the assembly instruction already says. As a non-programmer, I should be able to read the comments and know what the code is doing. Best thing is to review all of the code, figure out what the overall functionality is and then comment each line.

   For example, saying "set ecx to zero" is not sufficient – a non-programmer would still not know what that meant. Figure out what ecx is being used for and reference that. For instance, even though ecx is NOT a counter in this case, if it were, you would say, "set the counter to zero."

9. (4 pts) What is the "big picture" function of the code shown?

10. (2 pts) We already check for zero, why do this? "`cmp ecx,lengthName`"

11. (2 pts) What assembly instruction would you expect the compiler to produce for the C code "return result"?