Question #1 (15 points)

Use the following data definition:

```
var1    WORD    5000h, 4000h, 3000h, 2000h, 1000h
var2    DWORD   40000h, 30000h, 20000h, 10000h
```

a. What will be the final value of **ax** after this code has executed?

```
        mov   esi, OFFSET var1
        mov   ecx, 4
        mov   eax, 100h

L1:     add   ax, [esi]
        add   ax, 8
        add   esi, TYPE var1
        loop  L1
```

What will be the final value of ax after this code has executed?

```
        mov   edx, OFFSET var1+8
        mov   ecx, 2

L1:     mov   ax, [edx]
        add   ax, 30h
        sub   edx, 4
        loop  L1
```

Suppose we want **eax** to contain the sum of the **var1** array. Complete the remaining three instructions to be able to do so.

```
mov    edi, OFFSET var1
mov    ecx, LENGTHOF var1
?
?
?
loop   L1
```

Question #2 (10 points)

What are the contents of **eax** and **answer** after executing the following?

```
.data
alist    word        4000h, 5000h, 6000h
answer dword         0
.code
        mov    ebx, offset alist
        mov    eax, [ebx]
        add    eax, [ebx+4]
        sub    eax, [ebx+2]
        mov    [ebx+6], eax
```

Question #3 (10 points)

What is the result of the **eax** register after executing the following?

```
.data
array1  word        600h, 700h, 800h, 500h, 400h
count = ($ - array1) / 2
.code
        mov     eax, 0
        mov     edi, offset array1
        mov     ecx, count
L1:     add     eax, [edi]
        add     edi, 2
        loop    L1
```

Question #4 (25 points)

Use the following data definitions:

```
byte1       BYTE        0FFh, 1, 2
byte2       BYTE        14h
word1       WORD        0FFFFh, 1, 2
word2       WORD        3
word3       SWORD       7FFFh, 8000h
word4       SWORD       9000h
dword1      DWORD       10h, 20h, 30h, 40h
dArray      DWORD       10 DUP (?)
```

For each of the following instructions, indicate whether it is legal (L) or illegal (I)

a. mov byte2, 0FFh
b. mov word1, byte2
c. mov word2, 10000h
d. mov si, word1
e. movzx ax, byte1
f. movzx edx, bl
g. movzx word2, al
h. movsx dl, al
i. mov dx, word3
j. movsx eax, byte1

For the above data definitions, indicate the hexadecimal value of the destination operand. Use the letter (I) to indicate that a particular instruction is illegal

a. mov ax, [word3+2] ax =
b. mov eax, [dword1+4] eax =
c. mov al, [byte1+1] al =
d. mov eax, [word3+4] eax =

e.
```
mov  ax, word1
inc  ax          ax =
dec  ax          ax =
mov  ax, word3
neg  ax          ax =
add  ax, 0C2A5h  ax =
```

f.
```
mov  al, 7Fh
add  al, 2       ZF=  CF=  SF=  OF=
sub  al, 5       ZF=  CF=  SF=  OF=
mov  al, 80h
add  al, 80h     ZF=  CF=  SF=  OF=
neg  al          ZF=  CF=  SF=  OF=
```

Question #5 (10 points)

Use the following data definitions:

```
byte1       BYTE        0FFh, 1, 2
byte2       BYTE        14h
word1       WORD        0FFFFh, 1, 2
word2       WORD        3
word3       SWORD       7FFFh, 8000h
word4       SWORD       9000h
dword1      DWORD       10h, 20h, 30h, 40h
dArray      DWORD       10 DUP (?)
```

Implement the following expressions in assembly language, using 32-bit integ
may modify any registers you wish).

a. **eax = dword1 + ebx - ecx**
b. **eax = -dword1 + (edx - ecx) + 1**

Question #6 (15 points)

Write an Assembly Language program to find the lowest and the highest value among 6 different numbers introduced in the program. Output the original numbers using a loop. Output the lowest and the highest value.

Question #7 (15 points)

1) Perform the following operations:

473_8 * 27_8

$9FDE_{16}$ - $5EFF_{16}$

00111010_2 * 00010111_2

2) Change the following:

$5EBA_{16}$ to Binary

$7ABD_{16}$ to Octal

$2BF_{16}$ to Decimal

4256_8 to Binary

765_8 to Decimal

4665_8 to Hexadecimal