Assignment # 04

Microprocessor & Assembly Language

BS (CS)

Fall semester 2018

Use the following data for Questions 1-5:

```
.data
val1 BYTE 10h
val2 WORD 8000h
val3 DWORD 0FFFFh
val4 WORD 7FFFh
```

Q.1 Write an instruction that increments **val2**.

Q.2 Write an instruction that subtracts **val3** from EAX.

Q.3 Write instructions that subtract **val4** from **val2**.

Q.4 If **val2** is incremented by 1 using the ADD instruction, what will be the values of the CF and SF?

Q.5 If **val4** is incremented by 1 using the ADD instruction, what will be the values of the OF and SF?

Q.6 Where indicated, write down the values of the CF, SF, ZF, and OF after each instruction has executed:

```
mov ax,7FF0h
add al,10h          ; a. CF =      SF =      ZF =      OF =
add ah,1            ; b. CF =      SF =      ZF =      OF =
add ax,2            ; c. CF =      SF =      ZF =      OF =
```

Use the following data definitions for Questions 7 and 8:

```
myBytes BYTE 10h,20h,30h,40h
myWords WORD 8Ah,3Bh,72h,44h,66h
myDoubles DWORD 1,2,3,4,5
myPointer DWORD myDoubles
```

Q.7 Fill in the requested register values on the right side of the following instruction sequence:

```
mov esi,OFFSET myBytes
mov al,[esi]                ; a. AL =
mov al,[esi+3]              ; b. AL =
mov esi,OFFSET myWords + 2
mov ax,[esi]                ; c. AX =
mov edi,8
mov edx,[myDoubles + edi]   ; d. EDX =
mov edx,myDoubles[edi]      ; e. EDX =
mov ebx,myPointer
mov eax,[ebx+4]             ; f. EAX =
```

Q.8 Fill in the requested register values on the right side of the following instruction sequence:

```
mov esi,OFFSET myBytes
mov ax,[esi]                ; a. AX =
mov eax,DWORD PTR myWords   ; b. EAX =
mov esi,myPointer
```

```
mov ax,[esi+2]              ; c. AX =
mov ax,[esi+6]              ; d. AX =
mov ax,[esi-4]              ; e. AX =
```

Q.9 What will be the final value of EAX in this example?

```
mov eax,0
mov ecx,10      ; outer loop counter
L1:
mov eax,3
mov ecx,5       ; inner loop counter
L2:
add eax,5
loop L2         ; repeat inner loop
loop L1         ; repeat outer loop
```

Q.10 Revise the code from the preceding question so the outer loop counter is not erased when the inner loop starts.

Q.11 Write a sequence of MOV instructions that will exchange the upper and lower words in a doubleword variable named **three**.

Q.12 Using the XCHG instruction no more than three times, reorder the values in four 8-bit registers from the order A, B, C, D to B, C, D, A.

Q.13 Transmitted messages often include a parity bit whose value is combined with a data byte to produce an even number of 1 bits. Suppose a message byte in the AL register contains 01110101. Show how you could use the PF combined with an arithmetic instruction to determine if this message byte has even or odd parity.

Q.14 Write code using byte operands that adds two negative integers and causes the OF to be set.

Q.15 Write a sequence of two instructions that use addition to set the ZF and CF at the same time.

Q.16 Write a sequence of two instructions that set the CF using subtraction.

Q.17 Implement the following arithmetic expression in assembly language: EAX = –val2 + 7 –val3 + val1. Assume that val1, val2, and val3 are 32-bit integer variables.

Q.18 Write a loop that iterates through a doubleword array and calculates the sum of its elements using a scale factor with indexed addressing.

Q.19 Write a sequence of two instructions that set both the CF and OF at the same time.

Q.20 Write a sequence of instructions showing how the ZF could be used to indicate unsigned overflow after executing INC and DEC instructions.

Use the following data definitions for Questions 21–26:

```
.data
myBytes BYTE 10h,20h,30h,40h
myWords WORD 3 DUP(?),2000h
myString BYTE "ABCDE"
```

Q.21 What will be the value of EAX after each of the following instructions execute?

```
mov eax,TYPE myBytes        ; a.
mov eax,LENGTHOF myBytes    ; b.
mov eax,SIZEOF myBytes      ; c.
mov eax,TYPE myWords        ; d.
mov eax,LENGTHOF myWords    ; e.
mov eax,SIZEOF myWords      ; f.
mov eax,SIZEOF myString     ; g.
```

Q.22 Write a single instruction that moves the first two bytes in **myBytes** to the DX register. The resulting value will be 2010h.

Q.23 Write an instruction that moves the second byte in **myWords** to the AL register.

Q.24 Write an instruction that moves all four bytes in **myBytes** to the EAX register.

Q.25 Insert a LABEL directive in the given data that permits **myWords** to be moved directly to a 32-bit register.

Q.26 Insert a LABEL directive in the given data that permits **myBytes** to be moved directly to a 16-bit register.

Q.27 Write a program that uses the variables below and MOV instructions to copy the value from **bigEndian** to **littleEndian**, reversing the order of the bytes. The number's 32-bit value is understood to be 12345678 hexadecimal.

```
.data
bigEndian BYTE 12h,34h,56h,78h
littleEndian DWORD?
```

Q.28 Write a program that uses a loop to copy all the elements from an unsigned Word (16-bit) array into an unsigned doubleword (32-bit) array.

Q.29 Use a loop with indirect or indexed addressing to reverse the elements of an integer array in place. Do not copy the elements to any other array. Use the SIZEOF, TYPE, and LENGTHOF operators to make the program as flexible as possible if the array size and type should be changed in the future.

Q.30 Write a program with a loop and indirect addressing that copies a string from **source** to **target**, reversing the character order in the process. Use the following variables:

```
source BYTE "This is the source string",0
target BYTE SIZEOF source DUP('#')
```