Lab Assignment 2

Part 1

1. **Memory and Stack Access**

   (a) Trace the program below (without a computer, then with one).

```
mov bx,4000h ; address
mov ax,1144h
mov [bx],ax
mov word [bx+2],4455h
mov byte [bx+4],66h
mov byte [bx+5],77h
```

   (b) Trace the program below. The initial value of SP can be any value (without a computer, then with one).

```
mov ax,1144h
push ax
mov ax,4455h
push ax
mov ax,6677h
push ax
mov bp,sp
mov al,[bp+3]
mov bl,[bp+1]
```

   (c) Use push and pop to store and recall a certain number of data items (push-pop).

   (d) Use push and pop to swap the values of 2 data items.

   (e) Trace the program below:

```
MOV AX,1A25h
AND AX, 0xF0FFh
```

2. **Arithmetic Instructions**
   Write an 8086 assembly program that performs each of the following operations:

   (a) Addition of two signed 16-bit numbers.

   (b) Addition of one signed 16-bit number and one signed 8-bit number.

   (c) Addition of two signed 32-bit numbers, using the DD directive (Define Double).

   (d) Multiplication of two unsigned integer numbers: both 8 bits; both 16 bits; one 16-bit and one 8-bit.
   Provide the result and the state of the CF and OF flags after executing each of the following programs:

```
org 100h
mov al, 2
mov bl, 129
mul bl
ret
```

```
org 100h
mov ax, 1000h
mov bx, 10h
mul bx
ret
```

   (e) Multiplication of two **signed** integer numbers: both 8 bits; both 16 bits; one 16-bit and one 8-bit.
   Provide the result and the state of the CF and OF flags after executing each of the following three programs:

```
org 100h
mov al, 2
mov bl, 129
imul bl
ret
```

```
org 100h
mov al, -13
mov bl, 10
imul bl
ret
```

```
org 100h
mov ax, -3277
mov bx, 10
imul bx
ret
```

   (f) Calculate the expression: $5 \times A - 12 \times B$, where A and B are unsigned integer numbers encoded on one byte.

   (g) Divide two unsigned integer numbers: one on 16 bits (dividend) and the other on 8 bits (divisor); both on 8 bits; both on 16 bits.
   What happens after executing the following program:

```
org 100h
mov ax, 1456h
mov bl, 10h
div bl
ret
```

(h) Divide two signed integer numbers: one on 16 bits (dividend) and the other on 8 bits (divisor); both on 8 bits; both on 16 bits.
What happens after executing the following program:

```
org 100h
mov ax, 1300
mov bl, 10
idiv bl
ret
```

(i) Calculate the sum of integers from 1 to 10 (using 'loop').

Write an 8086 assembly program that declares an array of bytes (example: MEM db 0x1B, 127, 255, 0CFh), then displays the hexadecimal memory content corresponding to this array using the INT 21h instruction.

3. **Variable Usage (Declarations)**

(a) Perform the following operations using variable declarations:
- Addition of two signed 16-bit numbers.
- Multiplication of two signed integer numbers, one on 16 bits and the other on 8 bits.
- Division of two signed 16-bit numbers.

(b) Write a series of instructions to:
i. Add two vectors of the same size (using 'LEA' and 'loop').
ii. Sum the elements of an array.

4. **Using DOS Functions for Display (INT 21h)**
Write an 8086 assembly program that allows you to:

(a) Read a character from the keyboard.
(b) Display a character on the screen.
(c) Read and display a character.
(d) Use the characters: newline and carriage return.
(e) Display a string of characters (using 'mov dx, offset message' or 'lea dx, message').
(f) Display the list of alphanumeric characters (ASCII table).
(g) Convert a lowercase character to uppercase (with user prompt).

Part 2 (Looping and Branching)

1. Unroll the following program sequences; what do they do? (Use the emulator afterwards).

```
org 100h
Xor ax,ax
Mov cx, 10
etiq: add ax, cx
Loop etiq
ret
```

```
org 100h
.data
Num db 02, 06, 08, 04, 76, 07
.code
Lea si, Num
Mov cx, 5
Mov al, [si]
etiq1: cmp al, [si+1]
jge etiq2
mov al, [si+1]
etiq2: inc si
loop etiq1
ret
```

```
org 100h
.data
alpha db 'ABcdefg24 ?*',0
.code
mov bx,offset alpha
boucle: mov al,[bx]
Cmp al, 0
Je fin
Cmp al, 'a'
Jl etiq1
Cmp al, 'z'
Jg etiq1
Add al, 'A'-'a'
Mov [bx], al
etiq1: inc bx
Jmp boucle
fin: int 21h ; stop
```

2. Provide the assembly equivalent of the following portions of sequences (in C); $op1$, $op2$, $var1$, $var2$, $var3$, $var4$, and X are arbitrary signed variables on 16 bits:

if($op1 == op2$)	if($var1 <= var2$)	if $(al > bl)$ AND $(bl > cl)$	while$(ax < bx)$
$X = 1$;	$var3 = 10$;	$X = 1$;	$ax = ax + 1$;
Else $X = 2$;	else {		
	$var3 = 6$;		
	$var4 = 7$;}		