

# Microprocessor Lab – All Sample Programs (8086 / EMU8086)

Complete set: Arithmetic, Logic, Array, String, Factorial, Sorting

Each exercise: Problem, Code, Expected Output

(Only 16-bit programs as requested)

---

## 1. Ex-1: Add two 16-bit numbers (A + B)

```
; Program: Add two 16-bit numbers
MOV AX, 1234H
MOV BX, 1111H
ADD AX, BX
; Result in AX
; Output: AX = 2345H
HLT
Output: AX = 2345H (hex)
```

## 2. Ex-2: Subtract two 16-bit numbers (A - B)

```
; Program: Subtract two 16-bit numbers
MOV AX, 4567H
MOV BX, 1234H
SUB AX, BX
; Result in AX
; Output: AX = 3333H
HLT
Output: AX = 3333H (hex)
```

## 3. Ex-3: Add three 16-bit numbers (A + B + C)

```
; Program: A + B + C
MOV AX, 1000H
MOV BX, 2000H
MOV CX, 3000H
ADD AX, BX
ADD AX, CX
; Result in AX
; Output: AX = 6000H
HLT
Output: AX = 6000H (hex)
```

## 4. Ex-4: Subtract three 16-bit numbers (A - B - C)

```
; Program: A - B - C
MOV AX, 5000H
MOV BX, 1000H
MOV CX, 0500H
SUB AX, BX
SUB AX, CX
; Result in AX
; Output: AX = 3A00H (hex) ; 5000H-1000H-0500H = 3A00H
Output: AX = 3A00H (hex)
```

## 5. Ex-5: Add four 16-bit numbers (A+B+C+D)

```
; Program: A + B + C + D
MOV AX, 1000H
MOV BX, 2000H
MOV CX, 3000H
MOV DX, 4000H
ADD AX, BX
ADD AX, CX
ADD AX, DX
; Result in AX
; Output: AX = A000H
HLT
Output: AX = A000H (hex)
```

## 6. Ex-6: Multiply two 16-bit numbers (A \* B)

```
; Program: Multiply two 16-bit numbers  
MOV AX, 0005H  
MOV BX, 0003H  
MUL BX      ; AX * BX -> DX:AX  
; Result: AX = 000Fh, DX = 0000h  
HLT  
Output: AX = 000Fh, DX = 0000h
```

## 7. Ex-7: Divide two 16-bit numbers (A / B)

```
; Program: Divide two 16-bit numbers  
MOV AX, 0012H    ; dividend (18)  
MOV BX, 0005H    ; divisor (5)  
XOR DX, DX  
DIV BX          ; AX / BX -> quotient in AX, remainder in DX  
; Output: AX = 3, DX = 3  
HLT  
Output: Quotient AX = 0003H, Remainder DX = 0003H
```

## 8. Ex-8: Square of a number (A\*A)

```
; Program: Square of A  
MOV AX, 0005H  
MUL AX      ; AX * AX -> DX:AX  
; Output: AX = 0019H (25 decimal)  
HLT  
Output: AX = 0019H, DX = 0000H
```

## 9. Ex-9: Cube of a number (A\*A\*A)

```
; Program: Cube of A (A^3)  
MOV AX, 0003H  
MUL AX      ; AX = A*A  
; after first MUL DX:AX contains A^2  
MOV BX, AX  
MOV AX, BX  
MUL BX      ; AX * BX -> DX:AX gives A^3 in DX:AX  
; Output: for A=3, AX=0027H (39), check DX if large  
HLT  
Output: AX = 0027H (39 decimal)
```

## 10. Ex-10: (A + B) \* (C + D)

```
; Program: (A+B)*(C+D)  
MOV AX, 0002H  
MOV BX, 0003H  
MOV CX, 0004H  
MOV DX, 0005H  
ADD AX, BX    ; AX = A+B = 5  
ADD CX, DX    ; CX = C+D = 9  
MUL CX      ; AX * CX -> DX:AX  
; Output: 5 * 9 = 45 -> AX = 002DH  
HLT  
Output: AX = 002DH (45 decimal)
```

## 11. Ex-11: (A \* B) + (C \* D)

```
; Program: (A*B)+(C*D)  
MOV AX, 0002H  
MOV BX, 0003H  
MUL BX      ; AX = A*B -> AX=6  
MOV SI, AX    ; store partial result  
MOV AX, 0004H  
MOV BX, 0005H  
MUL BX      ; AX = C*D -> AX=20  
ADD AX, SI  
; Output: 6 + 20 = 26 -> AX = 001AH  
HLT  
Output: AX = 001AH (26 decimal)
```

## **12. Ex-12: (A + B) - (C + D)**

```
; Program: (A+B) - (C+D)
MOV AX, 2000H
MOV BX, 1000H
MOV CX, 0800H
MOV DX, 0100H
ADD AX, BX
ADD CX, DX
SUB AX, CX
; Output shown in AX
HLT
```

*Output: AX = 2800H - 0900H = 1F00H (depending on values)*

## **13. Ex-13: A - B \* B - C (follow operator precedence)**

```
; Program: A - B*B - C
; Compute B*B first, then A - (B*B) - C
MOV AX, 0000H
MOV BX, 0003H      ; B=3
MOV CX, 0002H      ; C=2
MOV DI, 000AH      ; A=10
MOV AX, BX
MUL BX            ; AX = B*B = 9
MOV BX, DI
SUB BX, AX        ; BX = A - B*B = 10 - 9 = 1
SUB BX, CX        ; BX = BX - C = 1 - 2 = -1 (wraps)
; Result in BX (signed)
HLT
```

*Output: BX = FFFFH (if -1 in 16-bit two's complement)*

## **14. Ex-14: Average of 3 numbers**

```
; Program: Average = (A+B+C)/3
MOV AX, 0009H
MOV BX, 0006H
MOV CX, 0003H
ADD AX, BX
ADD AX, CX
MOV BL, 03H
DIV BL
; Quotient in AL (and AH holds remainder)
HLT
```

*Output: Quotient AL = 0004H (4 decimal) remainder in AH*

## **15. Ex-15: Factorial (5!)**

```
; Program: Factorial of 5
MOV CX, 05H
MOV AX, 01H
FACT:
MUL CX
LOOP FACT
; Result in AX (for 5! = 120 -> 0078H)
HLT
```

*Output: AX = 0078H (120 decimal)*

## 16. Ex-16: Sum of N words in an array

```
; Program: Sum of word array
ARR DW 1000H,2000H,3000H,4000H,5000H
MOV SI, OFFSET ARR
MOV CX, 05
MOV AX, 0000H
SUMLOOP:
ADD AX, [SI]
ADD SI, 2
LOOP SUMLOOP
; Result in AX
HLT
```

*Output: AX = SUM of values (0x? depending on values)*

## 17. Ex-17: Find length of string (ending with \$)

```
; Program: Length of string until '$'
STR DB 'HELLO$'
MOV SI, OFFSET STR
MOV CX, 0
NEXT:
CMP BYTE PTR [SI], '$'
JE DONE
INC CX
INC SI
JMP NEXT
DONE:
; Length in CX = 5
HLT
```

*Output: CX = 0005H (5 characters)*

## 18. Ex-18: Move a byte string (REP MOVSB)

```
; Program: Move string from SRC to DEST
SRC DB 'HELLO'
DEST DB 5 DUP (?)
LEA SI, SRC
LEA DI, DEST
MOV CX, 05
REP MOVSB
; DEST now contains 'HELLO'
HLT
```

*Output: DEST = 'HELLO' in memory*

## 19. Ex-19: Bubble sort ascending (5 elements)

```
; Program: Bubble sort (ascending) for word array
NUMS DW 4321H,1234H,5678H,1111H,1000H
MOV CX, 05
DEC CX
MOV SI, OFFSET NUMS
OUTER:
MOV BX, CX
MOV SI, OFFSET NUMS
INNER:
MOV AX, [SI]
CMP AX, [SI+2]
JBE SKIP
XCHG AX, [SI+2]
MOV [SI], AX
SKIP:
ADD SI, 2
DEC BX
JNZ INNER
LOOP OUTER
; Sorted ascending
HLT
```

*Output: Array sorted ascending: 1000H,1111H,1234H,4321H,5678H*

## **20. Ex-20: Find largest of two numbers**

```
; Program: Largest of two  
MOV AX, 2000H  
MOV BX, 3000H  
CMP AX, BX  
JAE L1  
MOV AX, BX  
L1:  
; Largest now in AX = 3000H  
HLT
```

*Output: AX = 3000H*

## **21. Ex-21: Reverse digits of a number (decimal)**

```
; Program: Reverse digits (example 123 -> 321)  
; This is conceptual; requires byte/div loops and ascii conversion in full ALP  
; Simple: use DIV 10 to extract digits, build reversed number in BX  
; (Detailed ALP available in lab notes)  
HLT
```

*Output: Example: input 123 -> reversed 321 (in register)*

## **22. Ex-22: GCD of two numbers (Euclidean)**

```
; Program: GCD using repeated subtraction / mod  
; Conceptual ALP - use DIV and remainder until remainder = 0  
HLT
```

*Output: GCD in AX (example 48 and 18 -> GCD = 6)*

## **23. Ex-23: Check even or odd**

```
; Program: Check LSB  
MOV AX, 0005H  
TEST AX, 1  
JZ EVEN  
; if not zero -> odd  
; EVEN:  
; ...  
HLT
```

*Output: For AX=5 -> odd (jump not taken)*

## **24. Ex-24: Convert lowercase to uppercase (ASCII)**

```
; Program: Convert 'a' to 'A' (ASCII)  
MOV AL, 'a' ; 61h  
SUB AL, 20H ; convert to uppercase -> 'A'  
; AL = 'A'  
HLT
```

*Output: AL = 'A' (41h)*

## **Notes & Tips for Exams**

1. Always clear DX before DIV for 16-bit division (XOR DX, DX).
2. MUL is unsigned; use IMUL for signed multiplication (if needed).
3. Use LOOP with CX as counter; preserve CX if needed later.
4. Strings in many lab files use '\$' as terminator for INT 21h AH=09h display.
5. For multi-word results MUL gives DX:AX; check DX for overflow.