Somnath Chattaraj
IT-A3-067 ASM Lab Assignment-1

---

1. Write an Assembly Language Program to add two 8-bit integer numbers. The numbers can be stored in the data segment.

```
.MODEL SMALL
.STACK 100H

.DATA
   NUM1 DB 0AH
   NUM2 DB 05H
   RESULT DB ?

.CODE
MAIN PROC
   MOV AX, @DATA
   MOV DS, AX

   MOV AL, NUM1
   ADD AL, NUM2
   MOV RESULT, AL

   MOV AH, 4CH
   INT 21H

MAIN ENDP
END MAIN
```

---

2. Write an 8086 Assembly Language Program to subtract two 8-bit signed integers. The numbers can be stored in the data segment.

```
.MODEL SMALL
.STACK 100H

.DATA
   NUM1 DB -10
   NUM2 DB 5
   RESULT DB ?

.CODE
MAIN PROC
```

```
    MOV AX, @DATA
    MOV DS, AX

    MOV AL, NUM1
    SUB AL, NUM2
    MOV RESULT, AL

    MOV AH, 4CH
    INT 21H
MAIN ENDP
END MAIN
```

---

3. Write an 8086 Assembly Language Program to find out a maximum of three numbers.

```
.MODEL SMALL
.STACK 100H

.DATA
    NUM1 DB 10
    NUM2 DB 20
    NUM3 DB 15
    MAX DB ?

.CODE
MAIN PROC
    MOV AX, @DATA
    MOV DS, AX

    MOV AL, NUM1
    CMP AL, NUM2
    JL L1
    MOV AL, NUM2
L1: CMP AL, NUM3
    JL L2
    MOV AL, NUM3
L2: MOV MAX, AL

    MOV AH, 4CH
    INT 21H
MAIN ENDP
END MAIN
```

---

4. Write an Assembly Language Program to check whether an input number is even or odd.

```
.MODEL SMALL
.STACK 100H

.DATA
    NUM DB 5
    RESULT DB ?

.CODE
MAIN PROC
    MOV AX, @DATA
    MOV DS, AX

    MOV AL, NUM
    AND AL, 1
    JZ EVEN
    MOV RESULT, 1
    JMP DONE
EVEN: MOV RESULT, 0
DONE: MOV AH, 4CH
    INT 21H
MAIN ENDP
END MAIN
```

---

5. Write an Assembly Language Program to check whether an input number is Armstrong or not.

```
.MODEL SMALL
.STACK 100H

.DATA
    NUM DB 153
    RESULT DB ?

.CODE
MAIN PROC
    MOV AX, @DATA
    MOV DS, AX

    MOV AL, NUM
    MOV BL, AL
    MUL BL
    MUL BL
    CMP AX, NUM
    JNE NOT_ARM
    MOV RESULT, 1
    JMP DONE
NOT_ARM: MOV RESULT, 0
DONE: MOV AH, 4CH
    INT 21H
MAIN ENDP
END MAIN
```

---

6. Write an Assembly Language Program to check whether an input number is Prime or not.

```
.MODEL SMALL
.STACK 100H

.DATA
    NUM DB 7
    FLAG DB 1

.CODE
MAIN PROC
    MOV AX, @DATA
    MOV DS, AX

    MOV AL, NUM
    MOV CX, 2
PRIME_LOOP:
    CMP CX, AL
    JGE DONE
    MOV AH, AL
    DIV CL
    CMP AH, 0
    JE NOT_PRIME
    INC CX
    JMP PRIME_LOOP
NOT_PRIME: MOV FLAG, 0
DONE: MOV AH, 4CH
    INT 21H
MAIN ENDP
END MAIN
```

---

7. Write an Assembly Language Program to swap two numbers using stack.

```
.MODEL SMALL
.STACK 100H

.DATA
   NUM1 DB 10
   NUM2 DB 20

.CODE
MAIN PROC
   MOV AX, @DATA
   MOV DS, AX

   MOV AL, NUM1
   PUSH AX
   MOV AL, NUM2
   PUSH AX

   POP AX
   MOV NUM1, AL
   POP AX
   MOV NUM2, AL

   MOV AH, 4CH
   INT 21H
MAIN ENDP
END MAIN
```

---

8. Write an Assembly Language Program to accept a string and print it.

```
.MODEL SMALL
.STACK 100H

.DATA
   MSG DB 'Enter string: $'
   STR DB 10 DUP('$')

.CODE
MAIN PROC
   MOV AX, @DATA
   MOV DS, AX

   MOV AH, 09H
   LEA DX, MSG
   INT 21H

   MOV AH, 0AH
   LEA DX, STR
   INT 21H

   MOV AH, 09H
   LEA DX, STR+1
   INT 21H

   MOV AH, 4CH
   INT 21H
MAIN ENDP
END MAIN
```

---

9. Write an Assembly Language Program to find the length of a given string.

```
.MODEL SMALL
.STACK 100H

.DATA
   STR DB 'Hello$', 0
   LEN DB 0

.CODE
MAIN PROC
   MOV AX, @DATA
   MOV DS, AX

   LEA SI, STR
   MOV CX, 0
LEN_LOOP:
   LODSB
   CMP AL, '$'
   JE DONE
   INC CX
   JMP LEN_LOOP
DONE:
   MOV LEN, CL

   MOV AH, 4CH
   INT 21H
MAIN ENDP
END MAIN
```

---

10. Write an Assembly Language Program to reverse a string using stack and display the result.

```
.MODEL SMALL
.STACK 100H
```

```
.DATA
   STR DB 'Hello$', 0
   REVSTR DB 10 DUP(?)

.CODE
MAIN PROC
   MOV AX, @DATA
   MOV DS, AX

   LEA SI, STR
   MOV CX, 0
REV_LOOP:
   LODSB
   CMP AL, '$'
   JE DONE_REV
   PUSH AX
   INC CX
   JMP REV_LOOP
DONE_REV:
   LEA DI, REVSTR
PRINT_REV:
   POP AX
   MOV [DI], AL
   INC DI
   LOOP PRINT_REV

   MOV AH, 09H
   LEA DX, REVSTR
   INT 21H

   MOV AH, 4CH
   INT 21H
MAIN ENDP
END MAIN
```

---

11. Write an 8086 Assembly Language Program which will ask for a number and the no. will be taken from keyboard. Print the number in decimal, binary and hexadecimal format.

```
.MODEL SMALL
.STACK 100H

.DATA
   MSG DB 'Enter number: $'
   NUM DB 0
   BINSTR DB 9 DUP(?)
```

```
    HEXSTR DB 5 DUP(?)

.CODE
MAIN PROC
    MOV AX, @DATA
    MOV DS, AX

    MOV AH, 09H
    LEA DX, MSG
    INT 21H

    MOV AH, 01H
    INT 21H
    SUB AL, '0'
    MOV NUM, AL

    MOV AH, 4CH
    INT 21H
MAIN ENDP
END MAIN
```