8085 Assembly Language Programs & Explanations

1. Statement: Store the data byte 32H into memory location 4000H.

Program 1:

```
MVI A, 32H      : Store 32H in the accumulator
STA 4000H       : Copy accumulator contents at address 4000H
HLT             : Terminate program execution
```

Program 2:

```
LXI H           : Load HL with 4000H
MVI M           : Store 32H in memory location pointed by HL register pair
(4000H)
HLT             : Terminate program execution
```

2. Statement: Exchange the contents of memory locations 2000H and 4000H

Program 1:

```
    LDA 2000H     : Get the contents of memory location 2000H into
accumulator
    MOV B, A      : Save the contents into B register
    LDA 4000H     : Get the contents of memory location 4000Hinto
accumulator
    STA 2000H     : Store the contents of accumulator at address 2000H
    MOV A, B      : Get the saved contents back into A register
    STA 4000H     : Store the contents of accumulator at address 4000H
```

Program 2:

```
    LXI H 2000H         : Initialize HL register pair as a pointer to
memory location 2000H.
    LXI D 4000H         : Initialize DE register pair as a pointer to
memory location 4000H.
    MOV B, M            : Get the contents of memory location 2000H into B
register.
    LDAX D              : Get the contents of memory location 4000H into A
register.
    MOV M, A            : Store the contents of A register into memory
location 2000H.
    MOV A, B            : Copy the contents of B register into accumulator.
    STAX D              : Store the contents of A register into memory location
4000H.
    HLT                 : Terminate program execution.
```

3.Sample problem

(4000H) = 14H
(4001H) = 89H
Result = 14H + 89H = 9DH

Source program

```
LXI H 4000H          : HL points 4000H
MOV A, M           : Get first operand
INX H                : HL points 4001H
ADD M                : Add second operand
INX H                : HL points 4002H
MOV M, A           : Store result at 4002H
HLT                : Terminate program execution
```

4.Statement: Subtract the contents of memory location 4001H from the memory location 2000H and place the result in memory location 4002H.

Program - 4: Subtract two 8-bit numbers

Sample problem:

(4000H) = 51H
(4001H) = 19H
Result = 51H - 19H = 38H

Source program:

```
LXI H, 4000H          : HL points 4000H
MOV A, M            : Get first operand
INX H                 : HL points 4001H
SUB    M              : Subtract second operand
INX H                 : HL points 4002H
MOV M, A            : Store result at 4002H.
HLT                 : Terminate program execution
```

5.Statement: Add the 16-bit number in memory locations 4000H and 4001H to the 16-bit number in memory locations 4002H and 4003H. The most significant eight bits of the two numbers to be added are in memory locations 4001H and 4003H. Store the result in memory locations 4004H and 4005H with the most significant byte in memory location 4005H.

Program - 5.a: Add two 16-bit numbers - Source Program 1

Sample problem:

```
(4000H) = 15H
(4001H) = 1CH
(4002H) = B7H
(4003H) = 5AH
Result = 1C15 + 5AB7H = 76CCH
(4004H) = CCH
(4005H) = 76H
```

Source Program 1:

```
LHLD 4000H          : Get first I6-bit number in HL
XCHG                : Save first I6-bit number in DE
LHLD 4002H          : Get second I6-bit number in HL
MOV A, E            : Get lower byte of the first number
ADD L               : Add lower byte of the second number
MOV L, A            : Store result in L register
MOV A, D            : Get higher byte of the first number
ADC H               : Add higher byte of the second number with CARRY
MOV H, A            : Store result in H register
SHLD 4004H          : Store I6-bit result in memory locations 4004H and
4005H.
HLT                 : Terminate program execution
```

6.Statement: Add the contents of memory locations 40001H and 4001H and place the result in the memory locations 4002Hand 4003H.

Sample problem:

```
    (4000H) = 7FH
    (400lH) = 89H
Result  = 7FH + 89H = lO8H
    (4002H) = 08H
    (4003H) = 0lH
```

Source program:

```
    LXI H, 4000H        :HL Points 4000H
    MOV A, M            :Get first operand
    INX H               :HL Points 4001H
    ADD M               :Add second operand
    INX H               :HL Points 4002H
    MOV M, A            :Store the lower byte of result at 4002H
    MVIA, 00            :Initialize higher byte result with 00H
```

```
ADC A                    :Add carry in the high byte result
INX H                    :HL Points 4003H
MOV M, A                 :Store the higher byte of result at 4003H
HLT                      :Terminate program execution
```

7.Statement: **Subtract the 16-bit number in memory locations 4002H and 4003H from the 16-bit number in memory locations 4000H and 4001H. The most significant eight bits of the two numbers are in memory locations 4001H and 4003H. Store the result in memory locations 4004H and 4005H with the most significant byte in memory location 4005H.**

Sample problem

```
(4000H) = 19H
(400IH) = 6AH
(4004H) = I5H (4003H) = 5CH
Result    = 6A19H - 5C15H = OE04H
(4004H) = 04H
(4005H) = OEH
```

Source program:

```
LHLD 4000H           : Get first 16-bit number in HL
XCHG                 : Save first 16-bit number in DE
LHLD 4002H           : Get second 16-bit number in HL
MOV A, E             : Get lower byte of the first number
SUB L                : Subtract lower byte of the second number
MOV L, A             : Store the result in L register
MOV A, D             : Get higher byte of the first number
SBB H                : Subtract higher byte of second number with borrow
MOV H, A             : Store I6-bit result in memory locations 4004H and
4005H.
SHLD 4004H           : Store I6-bit result in memory locations 4004H and
4005H.
HLT                  : Terminate program execution
```

8.Statement: **Find the l's complement of the number stored at memory location 4400H and store the complemented number at memory location 4300H.**

Sample problem:

```
(4400H) = 55H
```

Result = (4300B) = AAB

Source program:

```
LDA 4400B          : Get the number
CMA                : Complement number
STA 4300H          : Store the result
HLT                : Terminate program execution
```

9.Statement: Find the 2's complement of the number stored at memory location 4200H and store the complemented number at memory location 4300H.

Sample problem:

(4200H) = 55H

Result = (4300H) = AAH + 1 = ABH

Source program:

```
LDA 4200H          : Get the number
CMA                : Complement the number
ADI, 01 H          : Add one in the number
STA 4300H          : Store the result
HLT                : Terminate program execution
```

10.Statement: Pack the two unpacked BCD numbers stored in memory locations 4200H and 4201H and store result in memory location 4300H. Assume the least significant digit is stored at 4200H.

Sample problem:

(4200H) = 04

(4201H) = 09

Result = (4300H) = 94

Source program

```
LDA 4201H          : Get the Most significant BCD digit
RLC
RLC
RLC
RLC                : Adjust the position of the second digit (09 is changed to
90)
```

```
ANI FOH          : Make least significant BCD digit zero
MOV C, A         : store the partial result
LDA 4200H          : Get the lower BCD digit
ADD C              : Add lower BCD digit
STA 4300H          : Store the result
HLT              : Terminate program execution
```

11.Statement: Two digit BCD number is stored in memory location 4200H. Unpack the BCD number and store the two digits in memory locations 4300H and 4301H such that memory location 4300H will have lower BCD digit.

Sample problem

(4200H) = 58
Result = (4300H) = 08 and
(4301H) = 05

Source program

```
LDA 4200H            : Get the packed BCD number
ANI FOH            : Mask lower nibble
RRC
RRC
RRC
RRC                : Adjust higher BCD digit as a lower digit
STA 4301H            : Store the partial result
LDA 4200H            : .Get the original BCD number
ANI OFH            : Mask higher nibble
STA 4201H            : Store the result
HLT                : Terminate program execution
```

12.Statement:Read the program given below and state the contents of all registers after the execution of each instruction in sequence.

Main program:

```
4000H      LXI SP, 27FFH
4003H      LXI H, 2000H
4006H      LXI B, 1020H
4009H      CALL SUB
400CH      HLT
```

Subroutine program:

```
4100H        SUB: PUSH B
4101H        PUSH H
4102H        LXI B, 4080H
4105H        LXI H, 4090H
4108H        SHLD 2200H
4109H        DAD B
410CH        POP H
410DH        POP B
410EH        RET
```

13.Statement:Write a program to shift an eight bit data four bits right. Assume that data is in register C.

Source program:

```
MOV A, C
RAR
RAR
RAR
RAR
MOV C, A
HLT
```

14.Statement: Program to shift a 16-bit data 1 bit left. Assume data is in the HL register pair

Source program:

```
DAD H      :      Adds HL data with HL data
```

15.Statement: Write a set of instructions to alter the contents of flag register in 8085.

```
PUSH PSW          : Save flags on stack
POP H             : Retrieve flags in 'L'
MOV A, L          : Flags in accumulator
CMA               : Complement accumulator
MOV L, A          : Accumulator in 'L'
```

```
PUSH H          : Save on stack
POP PSW          : Back to flag register
HLT             :Terminate program execution
```

16.Statement: Calculate the sum of series of numbers. The length of the series is in memory location 4200H and the series begins from memory location 4201H.
a. Consider the sum to be 8 bit number. So, ignore carries. Store the sum at memory location 4300H.
b. Consider the sum to be 16 bit number. Store the sum at memory locations 4300H and 4301H

a. Sample problem

```
4200H  = 04H
4201H  = 10H
4202H  = 45H
4203H  = 33H
4204H  = 22H
Result = 10 +41 + 30 + 12 =  H
4300H  =  H
```

Source program:

```
LDA 4200H
MOV C, A          : Initialize counter
SUB A             : sum = 0
LXI H, 420lH        : Initialize pointer
BACK:    ADD M          : SUM = SUM + data
INX H             : increment pointer
DCR C             : Decrement counter
JNZ BACK          : if counter  0 repeat
STA 4300H         : Store sum
HLT               : Terminate program execution
```

b. Sample problem

```
4200H =  04H
420lH  = 9AH
4202H = 52H
4203H = 89H
4204H = 3EH
Result = 9AH + 52H + 89H + 3EH = H
4300H = B3H Lower byte
4301H = 0lH Higher byte
```

Source program:

```
        LDA 4200H
        MOV C, A          : Initialize counter
        LXI H, 4201H        : Initialize pointer
        SUB A             :Sum low = 0
        MOV B, A          : Sum high = 0
BACK:   ADD M               : Sum = sum + data
        JNC SKIP
        INR B             : Add carry to MSB of SUM
SKIP:   INX H             : Increment pointer
        DCR C             : Decrement counter
        JNZ BACK          : Check if counter 0 repeat
        STA 4300H         : Store lower byte
        MOV A, B
        STA 4301H         : Store higher byte
        HLT               :Terminate program execution
```

17.Statement: Multiply two 8-bit numbers stored in memory locations 2200H and 2201H by repetitive addition and store the result in memory locations 2300H and 2301H.

Sample problem:

```
(2200H) = 03H
(2201H) = B2H
        Result = B2H + B2H + B2H = 216H
               = 216H
(2300H) = 16H
(2301H) = 02H
```

Source program

```
        LDA 2200H
        MOV E, A
        MVI D, 00         : Get the first number in DE register pair
        LDA 2201H
        MOV C, A          : Initialize counter
        LX I H, 0000 H    : Result = 0
BACK:   DAD     D           : Result = result + first number
        DCR     C         : Decrement count
        JNZ     BACK          : If count  0 repeat
        SHLD 2300H          : Store result
        HLT               : Terminate program execution
```

18.Statement:Divide 16 bit number stored in memory locations 2200H and 2201H by the 8 bit number stored at memory location 2202H. Store the quotient in memory locations 2300H and 2301H and remainder in memory locations 2302H and 2303H.

Sample problem

(2200H) = 60H
(2201H) = A0H
(2202H) = l2H
Result = A060H/12H = 8E8H Quotient and 10H remainder
(2300H) = E8H
(2301H) = 08H
(2302H= 10H
(2303H) 00H

Source program

```
    LHLD 2200H          : Get the dividend
    LDA 2202H           : Get the divisor
    MOV C, A
    LXI D, 0000H        : Quotient = 0
BACK: MOV A, L
    SUB C               : Subtract divisor
    MOV L, A            : Save partial result
    JNC SKIP            : if CY  1 jump
    DCR H               : Subtract borrow of previous subtraction
SKIP: INX D             : Increment quotient
    MOV A, H
    CPI, 00             : Check if dividend < divisor
    JNZ BACK            : if no repeat
    MOV A, L
    CMP C
    JNC BACK
    SHLD 2302H          : Store the remainder
    XCHG
    SHLD 2300H          : Store the quotient
    HLT                 : Terminate program execution
```

19.Statement:Find the number of negative elements (most significant bit 1) in a block of data. The length of the block is in memory location 2200H and the block itself begins in memory location 2201H. Store the number of negative elements in memory location 2300H

Sample problem

(2200H) = 04H

(2201H) = 56H
(2202H) = A9H
(2203H) = 73H
(2204H) = 82H

Result = 02 since 2202H and 2204H contain numbers with a MSB of 1.

Source program

```
        LDA 2200H
        MOV C, A          : Initialize count
        MVI B, 00         : Negative number = 0
        LXI H, 2201H      : Initialize pointer
BACK:   MOV A, M          : Get the number
        ANI 80H           : Check for MSB
        JZ SKIP           : If MSB = 1
        INR B             : Increment negative number count
SKIP:   INX H             : Increment pointer
        DCR C             : Decrement count
        JNZ BACK          : If count  0 repeat
        MOV A, B
        STA 2300H         : Store the result
        HLT               : Terminate program execution
```

20.Statement:Find the largest number in a block of data. The length of the block is in memory location 2200H and the block itself starts from memory location 2201H.
Store the maximum number in memory location 2300H. Assume that the numbers in the block are all 8 bit unsigned binary numbers.

Sample problem

(2200H) = 04
(2201H) = 34H
(2202H) = A9H
(2203H) = 78H
(2204H) =56H
Result = (2202H) = A9H

Source program

```
        LDA 2200H
        MOV C, A          : Initialize counter
        XRA A             : Maximum = Minimum possible value = 0
        LXI H, 2201H      : Initialize pointer
BACK:   CMP M             : Is number> maximum
        JNC SKIP          : Yes, replace maximum
```

```
        MOV A, M
SKIP:   INX H
        DCR C
        JNZ BACK
        STA 2300H          : Store maximum number
        HLT                : Terminate program execution
```

21.Statement:Write a program to count number of l's in the contents of D register and store the count in the B register.

Source program:

```
        MVI B, 00H
        MVI C, 08H
        MOV A, D
BACK:   RAR
        JNC SKIP
        INR B
SKIP:   DCR C
        JNZ BACK
        HLT
```

22.Statement:Write a program to sort given 10 numbers from memory location 2200H in the ascending order.

Source program:

```
        MVI B, 09          : Initialize counter
        START              : LXI H, 2200H: Initialize memory pointer
        MVI C, 09H         : Initialize counter 2
BACK:   MOV A, M           : Get the number
        INX H              : Increment memory pointer
        CMP M              : Compare number with next number
        JC SKIP            : If less, don't interchange
        JZ SKIP            : If equal, don't interchange
        MOV D, M
        MOV M, A
        DCX H
        MOV M, D
        INX H              : Interchange two numbers
SKIP:   DCR C              : Decrement counter 2
        JNZ BACK           : If not zero, repeat
        DCR B              : Decrement counter 1
        JNZ START
        HLT                : Terminate program execution
```

23.Statement:Calculate the sum of series of even numbers from the list of numbers. The length of the list is in memory location 2200H and the series itself begins from memory location 2201H. Assume the sum to be 8 bit number so you can ignore carries and store the sum at memory location 2*Sample problem:*

```
2200H= 4H
2201H= 20H
2202H= l5H
2203H= l3H
2204H= 22H
Result 22l0H= 20 + 22 = 42H
          = 42H
```

Source program:

```
        LDA 2200H
        MOV C, A          : Initialize      counter
        MVI B, 00H          : sum = 0
        LXI H, 2201H          : Initialize pointer
BACK: MOV A, M          : Get      the      number
        ANI      0lH          : Mask      Bit l      to      Bit7
        JNZ SKIP          : Don't add if number is ODD
        MOV A, B          : Get      the      sum
        ADD      M          : SUM      = SUM      + data
        MOV B, A          : Store      result      in      B      register
SKIP: INX H                 : increment pointer
        DCR      C          : Decrement      counter
        JNZ      BACK          : if counter  0 repeat
        STA      2210H          : store      sum
        HLT                 : Terminate program execution
```

24.Statement:Calculate the sum of series of odd numbers from the list of numbers. The length of the list is in memory location 2200H and the series itself begins from memory location 2201H. Assume the sum to be 16-bit. Store the sum at memory locations 2300H and 2301H.

Sample problem:

```
2200H = 4H
2201H= 9AH
2202H= 52H
2203H= 89H
2204H= 3FH
    Result = 89H + 3FH = C8H
        2300H= H Lower byte
        2301H = H Higher byte
```

Source program

```
        LDA 2200H
        MOV C, A          : Initialize counter
        LXI H, 2201H          : Initialize pointer
        MVI E, 00         : Sum low = 0
        MOV D, E          : Sum high = 0
BACK:   MOV A, M          : Get the number
        ANI 0lH           : Mask Bit 1 to Bit7
        JZ SKIP           : Don't add if number is even
        MOV A, E          : Get the lower byte of sum
        ADD M             : Sum = sum + data
        MOV E, A          : Store result in E register
        JNC SKIP
        INR D                 : Add carry to MSB of SUM
SKIP:   INX H                 : Increment pointer
        DCR C             : Decrement
```

25.Statement:Find the square of the given numbers from memory location 6100H and store the result from memory location 7000H

Source Program:

```
        LXI H, 6200H      : Initialize lookup table pointer
        LXI D, 6100H          : Initialize source memory pointer
        LXI B, 7000H      : Initialize destination memory pointer
BACK:   LDAX D            : Get the number
        MOV L, A          : A point to the square
        MOV A, M          : Get the square
        STAX B            : Store the result at destination memory location
        INX D             : Increment source memory pointer
        INX B                 : Increment destination memory pointer
        MOV A, C
        CPI 05H           : Check for last number
        JNZ BACK          : If not repeat
        HLT               : Terminate program execution
```

26.Statement: Search the given byte in the list of 50 numbers stored in the consecutive memory locations and store the address of memory location in the memory locations 2200H and 2201H. Assume byte is in the C register and starting address of the list is 2000H. If byte is not found store 00 at 2200H and 2201H.

Source program:

```
        LX I H, 2000H       : Initialize memory pointer 52H
        MVI B, 52H            : Initialize counter
BACK: MOV A, M              : Get the number
        CMP C             : Compare with the given byte
        JZ LAST             : Go last if match occurs
        INX H             : Increment memory pointer
        DCR B                 : Decrement counter
        JNZ B           : I f not zero, repeat
        LXI H, 0000H
        SHLD 2200H
        JMP END           : Store 00 at 2200H and 2201H
LAST: SHLD 2200H              : Store memory address
END:  HLT                   : Stop
```

27.Statement: **Two decimal numbers six digits each, are stored in BCD package form. Each number occupies a sequence of byte in the memory. The starting address of first number is 6000H Write an assembly language program that adds these two numbers and stores the sum in the same format starting from memory location 6200H**

Source Program:

```
        LXI H, 6000H            : Initialize pointer l to first number
        LXI D, 6l00H          : Initialize pointer2 to second number
        LXI B, 6200H      : Initialize pointer3 to result
        STC
        CMC                   : Carry = 0
BACK: LDAX D                  : Get the digit
        ADD M             : Add two digits
        DAA                   : Adjust for decimal
        STAX.B            : Store the result
        INX H                 : Increment pointer 1
        INX D             : Increment pointer2
        INX B                 : Increment result pointer
        MOV A, L
        CPI 06H           : Check for last digit
        JNZ BACK              : If not last digit repeat
        HLT                 : Terminate program execution
```

28.Statement: **Add 2 arrays having ten 8-bit numbers each and generate a third array of result. It is necessary to add the first element of array 1 with the first**

element of array-2 and so on. The starting addresses of array l, array2 and array3 are 2200H, 2300H and 2400H, respectively.

Source Program:

```
        LXI H, 2200H        : Initialize memory pointer 1
        LXI B, 2300H        : Initialize memory pointer 2
        LXI D, 2400H        : Initialize result pointer
BACK:   LDAX B              : Get the number from array 2
        ADD M               : Add it with number in array 1
        STAX D              : Store the addition in array 3
        INX H               : Increment pointer 1
        INX B               : Increment pointer2
        INX D               : Increment result pointer
        MOV A, L
        CPI 0AH             : Check pointer 1 for last number
        JNZ BACK            : If not, repeat
        HLT                 : Stop
```

29.Statement: Write an assembly language program to separate even numbers from the given list of 50 numbers and store them in the another list starting from 2300H. Assume starting address of 50 number list is 2200H

Source Program:

```
        LXI H, 2200H        : Initialize memory pointer l
        LXI D, 2300H        : Initialize memory pointer2
        MVI C, 32H          : Initialize counter
BACK:   MOV A, M            : Get the number
        ANI 0lH             : Check for even number
        JNZ SKIP            : If ODD, don't store
        MOV A, M            : Get the number
        STAX D              : Store the number in result list
        INX D               : Increment pointer 2
SKIP:   INX H               : Increment pointer l
        DCR C               : Decrement counter
        JNZ BACK            : If not zero, repeat
        HLT                 : Stop
```

30.Statement: Write assembly language program with proper comments for the following:

A block of data consisting of 256 bytes is stored in memory starting at 3000H. This block is to be shifted (relocated) in memory from 3050H onwards. Do not shift the block or part of the block anywhere else in the memory.

Source Program:

Two blocks (3000 - 30FF and 3050 - 314F) are overlapping. Therefore it is necessary to transfer last byte first and first byte last.

```
        MVI C, FFH              : Initialize counter
        LX I H, 30FFH           : Initialize source memory pointer 3l4FH
        LXI D, 314FH            : Initialize destination memory pointer
BACK:   MOV A, M                : Get byte from source memory block
        STAX D                  : Store byte in the destination memory block
        DCX H                   : Decrement source memory pointer
        DCX                     : Decrement destination memory pointer
        DCR C                   : Decrement counter
        JNZ BACK                : If counter  0 repeat
        HLT                     : Stop execution
```

31.Statement: Add even parity to a string of 7-bit ASCII characters. The length of the string is in memory location 2040H and the string itself begins in memory location 2041H. Place even parity in the most significant bit of each character.

Source Program:

```
        LXI H, 2040H
        MOV C ,M                : Counter for character
REPEAT: INX H                   : Memory pointer to character
        MOV A,M                 : Character in accumulator
        ORA A                   : ORing with itself to check parity.
        JPO PAREVEN             : If odd parity place
        ORI 80H                 even parity in D7 (80).
PAREVEN:MOV M , A               : Store converted even parity character.
        DCR C                   : Decrement counter.
        JNZ REPEAT              : If not zero go for next character.
        HLT
```

32.Statement: A list of 50 numbers is stored in memory, starting at 6000H. Find number of negative, zero and positive numbers from this list and store these results in memory locations 7000H, 7001H, and 7002H respectively

Source Program:

```
        LXI H, 6000H          : Initialize memory pointer
        MVI C, 00H            : Initialize number counter
        MVI B, 00H            : Initialize negative number counter
        MVI E, 00H            : Initialize zero number counter
BEGIN:MOV A, M                : Get the number
        CPI 00H               : If number = 0
        JZ ZERONUM            : Goto zeronum
        ANI     80H           : If MSB of number = 1i.e. if
        JNZ NEGNUM            number is negative goto NEGNUM
        INR D                 : otherwise increment positive number counter
        JMP LAST
ZERONUM:INR E                 : Increment zero number counter
        JMP LAST
NEGNUM:INR B                  : Increment negative number counter
LAST:INX H                    : Increment memory pointer
        INR C                 : Increment number counter
        MOV A, C
        CPI 32H               : If number counter = 5010 then
        JNZ BEGIN             : Store      otherwise check next number
        LXI H, 7000           : Initialize memory pointer.
        MOV M, B              : Store      negative number.
        INX H
        MOV M, E              : Store      zero number.
        INX H
        MOV M, D              : Store positive number.
        HLT                   : Terminate execution
```

33.Statement:Write an 8085 assembly language program to insert a string of four characters from the tenth location in the given array of 50 characters

Solution:

Step 1: Move bytes from location 10 till the end of array by four bytes downwards.

Step 2: Insert four bytes at locations 10, 11, 12 and 13.

Source Program:

```
        LXI H, 2l31H          : Initialize pointer at the last location of array.
        LXI D, 2l35H          : Initialize another pointer to point the last
location of array after insertion.
AGAIN: MOV A, M               : Get the character
```

```
        STAX D          : Store at the new location
        DCX D           : Decrement destination pointer
        DCX H           : Decrement source pointer
        MOV A, L        : [check whether desired
        CPI 05H         bytes are shifted or not]
        JNZ AGAIN       : if not repeat the process
        INX H           : adjust the memory pointer
        LXI D, 2200H    : Initialize the memory pointer to point the string to
be inserted
REPE: LDAX D            : Get the character
        MOV M, A        : Store it in the array
        INX D           : Increment source pointer
        INX H           : Increment destination pointer
        MOV A, E        : [Check whether the 4 bytes
        CPI 04          are inserted]
        JNZ REPE        : if not repeat the process
        HLT             : stop
```

34.Statement:Write an 8085 assembly language program to delete a string of 4 characters from the tenth location in the given array of 50 characters.

Solution: Shift bytes from location 14 till the end of array upwards by 4 characters i.e. from location 10 onwards.

Source Program:

```
LXI H, 2l0DH        :Initialize source memory pointer at the 14thlocation
of the array.
LXI D, 2l09H        : Initialize destn memory pointer at the 10th location
of the array.
MOV A, M        : Get the character
STAX D          : Store character at new location
INX D           : Increment destination pointer
INX H           : Increment source pointer
MOV A, L        : [check whether desired
CPI 32H         bytes are shifted or not]
JNZ REPE        : if not repeat the process
HLT             : stop
```

35.Statement:Multiply the 8-bit unsigned number in memory location 2200H by the 8-bit unsigned number in memory location 2201H. Store the 8 least significant bits of the result in memory location 2300H and the 8 most significant bits in memory location 2301H.

Sample problem:

```
        (2200)       = 1100 (0CH)
        (2201)       = 0101 (05H)
    Multiplicand      = 1100 (1210)
    Multiplier      =  0101 (510)
    Result            = 12 x 5 = (6010)
```

Source program

```
        LXI H, 2200          : Initialize the memory pointer
        MOV E, M           : Get multiplicand
        MVI D, 00H           : Extend to 16-bits
        INX H                : Increment memory pointer
        MOV A, M           : Get multiplier
        LXI H, 0000          : Product = 0
        MVI B, 08H           : Initialize counter with count 8
    MULT: DAD H                : Product = product x 2
        RAL
        JNC SKIP           : Is carry from multiplier 1 ?
        DAD D                : Yes, Product =Product + Multiplicand
    SKIP:  DCR B               : Is counter = zero
        JNZ MULT           : no, repeat
        SHLD 2300H           : Store the result
        HLT                : End of program
```

36.Statement:Divide the 16-bit unsigned number in memory locations 2200H and 2201H (most significant bits in 2201H) by the B-bit unsigned number in memory location 2300H store the quotient in memory location 2400H and remainder in 2401H

Assumption: The most significant bits of both the divisor and dividend are zero.

Source program

```
    MVI E, 00          : Quotient = 0
    LHLD     2200H           : Get dividend
    LDA 2300           : Get divisor
    MOV B, A           : Store      divisor
    MVI C, 08          : Count = 8
NEXT: DAD H            : Dividend = Dividend x 2
    MOV A, E
    RLC
    MOV E, A           : Quotient = Quotient x 2
```

```
      MOV A, H
      SUB B              : Is most significant byte of Dividend > divisor
      JC SKIP          : No, go to Next step
      MOV H, A          : Yes, subtract divisor
      INR E              : and Quotient = Quotient + 1
SKIP:DCR C               : Count = Count - 1
      JNZ NEXT          : Is count =0 repeat
      MOV A, E
      STA 2401H          : Store Quotient
      Mov A, H
      STA 2410H          : Store remainder
      HLT               : End of program
```

37.DAA instruction is not present. Write a sub routine which will perform the same task as DAA.

Sample Problem:

Execution of DAA instruction:
1. If the value of the low order four bits (03-00) in the accumulator is greater than 9 or if auxiliary carry flag is set, the instruction adds 6 '(06) to the low-order four bits.
2. If the value of the high-order four bits (07-04) in the accumulator is greater than 9 or if carry flag is set, the instruction adds 6(06) to the high-order four bits.

Source Program:

```
      LXI SP, 27FFH      : Initialize stack pointer
      MOV E, A          : Store the contents of accumulator
      ANI 0FH          : Mask upper nibble
      CPI 0A H          : Check if number is greater than 9
      JC SKIP          : if no go to skip
      MOV A, E          : Get the number
      ADI 06H          : Add 6 in the number
      JMP SECOND            : Go for second check
SKIP: PUSH PSW             : Store accumulator and flag contents in stack
      POP B               : Get the contents of accumulator in B register and
flag register contents in                    C register
      MOV A, C          : Get flag register contents in accumulator
      ANI 10H           : Check for bit 4
      JZ SECOND            : if zero, go for second check
      MOV A, E          : Get the number
      ADI 06          : Add 6 in the number
SECOND: MOV E, A           : Store the contents of accumulator
      ANI FOH          : Mask lower nibble
      RRC
      RRC
      RRC
```

```
      RRC                 : Rotate number 4 bit right
      CPI 0AH             : Check if number is greater than 9
      JC SKIPl            : if no go to skip 1
      MOV A, E            : Get the number
      ADI 60 H            : Add 60 H in the number
      JMP LAST            : Go to last
SKIP1: JNC LAST                 : if carry flag = 0 go to last
      MOV A, E            : Get the number
      ADI 60 H            : Add 60 H in the number
LAST: HLT
```

38.tement:To test RAM by writing '1' and reading it back and later writing '0' (zero) and reading it back. RAM addresses to be checked are 40FFH to 40FFH. In case of any error, it is indicated by writing 01H at port 10H

Source Program:

```
      LXI H, 4000H        : Initialize memory pointer
BACK: MVI M, FFH                : Writing '1' into RAM
      MOV A, M            : Reading data       from RAM
      CPI FFH             : Check for ERROR
      JNZ ERROR                 : If yes go to ERROR
      INX H                     : Increment memory pointer
      MOV A, H
      CPI SOH             : Check for last check
      JNZ BACK            : If not last, repeat
      LXI H, 4000H              : Initialize memory pointer
BACKI: MVI M, OOH               : Writing '0' into RAM
      MOV A, M            : Reading data from RAM
      CPI OOH             : Check for       ERROR
      INX H                     : Increment memory pointer
      MOV A, H
      CPI SOH             : Check for last check
      JNZ BACKI                 : If not last, repeat
      HLT                 : Stop Execution
```

39.tement:Write an assembly language program to generate fibonacci number

Source Program:

```
      MVI D, COUNT        : Initialize counter
      MVI B, 00           : Initialize variable to store previous number
      MVI C, 01           : Initialize variable to store current number
```

```
    MOV A, B          :[Add two numbers]
BACK: ADD C               :[Add two numbers]
    MOV B, C          : Current number is now previous number
    MOV C, A          : Save result as a new current number
    DCR D               : Decrement count
    JNZ BACK          : if count  0 go to BACK
    HLT               : Stop
```

40.tement: **Write a program to generate a delay of 0.4 sec if the crystal frequency is 5 MHz**

Calculation: In 8085, the operating frequency is half of the crystal frequency,
ie.Operating frequency = 5/2 = 2.5 MHz
Time for one T -state =
Number of T-states required =
= 1 x 106

Source Program:

```
LXI B, count        : 16 - bit count
BACK: DCX B         : Decrement count
MOV A, C
ORA B               : Logically OR Band C
JNZ BACK            : If result is not zero repeat
```

41.tement: **Arrange an array of 8 bit unsigned no in descending order**

Source Program:

```
START:MVI B, 00       ; Flag = 0
    LXI H, 4150       ; Count = length of array
    MOV C, M
    DCR C             ; No. of pair = count -1
    INX H             ; Point to start of array
LOOP:MOV A, M         ; Get kth element
    INX H
    CMP M             ; Compare to (K+1) th element
    JNC LOOP 1        ; No interchange if kth >= (k+1) th
    MOV D, M          ; Interchange if out of order
    MOV M, A          ;
    DCR H
    MOV M, D
    INX H
    MVI B, 01H        ; Flag=1
LOOP 1:DCR C          ; count down
    JNZ LOOP          ;
    DCR B             ; is flag = 1?
```

```
        JZ START            ; do another sort, if yes
        HLT                 ; If flag = 0, step execution
```

42.tement: Transfer ten bytes of data from one memory to another memory block. Source memory block starts from memory location 2200H where as destination memory block starts from memory location 2300H

Source Program:

```
        LXI H, 4150     : Initialize memory pointer
        MVI B, 08         : count for 8-bit
        MVI A, 54
    LOOP : RRC
        JC LOOP1
        MVI M, 00         : store zero it no carry
        JMP COMMON
    LOOP2: MVI M, 01     : store one if there is a carry
COMMON:  INX H
        DCR B      : check for carry
        JNZ LOOP
        HLT          : Terminate the program
```

43.tement: Program to calculate the factorial of a number between 0 to 8

Source program

```
    LXI SP, 27FFH      ; Initialize stack pointer
    LDA 2200H          ; Get the number
    CPI 02H          ; Check if number is greater than 1
    JC LAST
    MVI D, 00H         ; Load number as a result
    MOV E, A
    DCR A
    MOV C,A          ; Load counter one less than number
    CALL FACTO         ; Call subroutine FACTO
    XCHG             ; Get the result in HL
    SHLD 2201H         ; Store result in the memory
    JMP END
LAST: LXI H, 000lH         ; Store result = 01
END: SHLD 2201H
    HLT
```

44.tement:Write a program to find the Square Root of an 8 bit binary number. The binary number is stored in memory location 4200H and store the square root in 4201H.

Source Program:

```
        LDA 4200H       : Get the given data(Y) in A register
        MOV B,A         : Save the data in B register
        MVI C,02H       : Call the divisor(02H) in C register
        CALL DIV        : Call division subroutine to get initial value(X)
in D-reg
   REP: MOV E,D         : Save the initial value in E-reg
        MOV A,B         : Get the dividend(Y) in A-reg
        MOV C,D         : Get the divisor(X) in C-reg
        CALL DIV        : Call division subroutine to get initial
value(Y/X) in D-reg
        MOV A, D        : Move Y/X in A-reg
        ADD E           : Get the((Y/X) + X) in A-reg
        MVI C, 02H      : Get the divisor(02H) in C-reg
        CALL DIV        : Call division subroutine to get ((Y/X) + X)/2
in D-reg.This is XNEW
        MOV A, E        : Get Xin A-reg
        CMP D           : Compare X and XNEW
        JNZ REP         : If XNEW is not equal to X, then repeat
        STA 4201H       : Save the square root in memory
        HLT             : Terminate program execution
```

45.tement:Write a simple program to Split a HEX data into two nibbles and store it in memory

Source Program:

```
   LXI H, 4200H         : Set pointer data for array
   MOV B,M              : Get the data in B-reg
   MOV A,B              : Copy the data to A-reg
   ANI OFH              : Mask the upper nibble
   INX H                : Increment address as 4201
   MOV M,A              : Store the lower nibble in memory
   MOV A,B              : Get the data in A-reg
   ANI FOH              : Bring the upper nibble to lower nibble position
   RRC
   RRC
   RRC
   RRC
   INX H
   MOV M,A              : Store the upper nibble in memory
   HLT                  : Terminate program execution
```

46.tement: Add two 4 digit BCD numbers in HL and DE register pairs and store result in memory locations, 2300H and 2301H. Ignore carry after 16 bit.

Sample Problem:

```
(HL) =3629
(DE) =4738
     Step 1 : 29 + 38 = 61 and auxiliary carry flag = 1
     :.add      06
     61 + 06 = 67
     Step 2 :  36 + 47 + 0 (carry of LSB) = 7D

Lower nibble of addition is greater than 9, so add 6.
     7D + 06 = 83
     Result = 8367
```

Source program

```
MOV A, L          : Get lower 2 digits of no. 1
ADD E               : Add two lower digits
DAA               : Adjust result to valid BCD
STA 2300H           : Store partial result
MOV A, H          : Get most significant 2 digits of number
ADC D               : Add two most significant digits
DAA               : Adjust result to valid BCD
STA 2301H           : Store partial result
HLT               : Terminate program execution
```

47.tement: Subtract the BCD number stored in E register from the number stored in the D register.

Source Program:

```
MVI A,99H
SUB E          : Find the 99's complement of subtrahend
INR A          : Find 100's complement of subtrahend
ADD D        : Add minuend to 100's complement of subtrahend
DAA          : Adjust for BCD
HLT          : Terminate program execution
```

48.tement: Write an assembly language program to multiply 2 BCD numbers

Source Program:

```
        MVI C, Multiplier       : Load BCD multiplier
        MVI B, 00               : Initialize counter
        LXI H, 0000H            : Result = 0000
        MVI E, multiplicand     : Load multiplicand
        MVI D, 00H              : Extend to 16-bits
BACK:   DAD D                   : Result Result + Multiplicand
        MOV A, L                : Get the lower byte of the result
        ADI, 00H
        DAA                     : Adjust the lower byte of result to BCD.
        MOV L, A                : Store the lower byte of result
        MOV A, H                : Get the higher byte of the result
        ACI, 00H
        DAA                     : Adjust the higher byte of the result to BCD
        MOV H, A                : Store the higher byte of result.
        MOV A, B                : [Increment
        ADI 01H                 : counter
        DAA                     : adjust it to BCD and
        MOV B,A                 : store it]
        CMP C                   : Compare if count = multiplier
        JNZ BACK                : if not equal repeat
        HLT                     : Stop
```