Practical No. 1

Aim: Store 8-bit data in memory

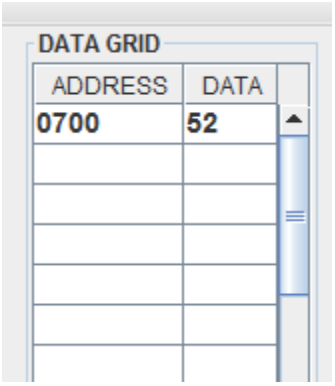
MVI A, 52H : "Store 32H in the accumulator"

STA 0700H : "Copy accumulator contents at address 4000H"

HLT : "Terminate program execution"

OUTPUT:

After executing this program we will get



Practical No. 2.

Aim: Exchange the contents of memory locations 0700H and 0701H.

LDA 0700H: "Get the contents of memory location 0700H into accumulator"

MOV B, A : "Save the contents into B register"

LDA 0701H : "Get the contents of memory location 0701Hinto accumulator"

STA 0700H : "Store the contents of accumulator at address 0700H"

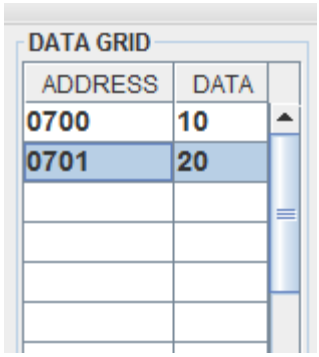
MOV A, B : "Get the saved contents back into A register"

STA 0701H : "Store the contents of accumulator at address 0701H"

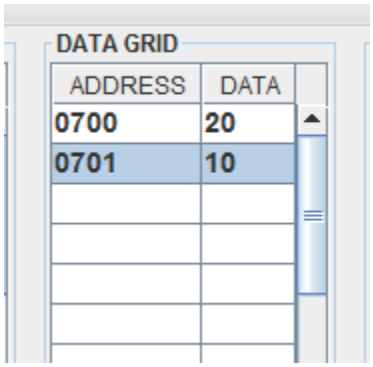
HLT

Output:

Before executing program type the following data in data grid



After executing this program we will get



Practical No. 3.

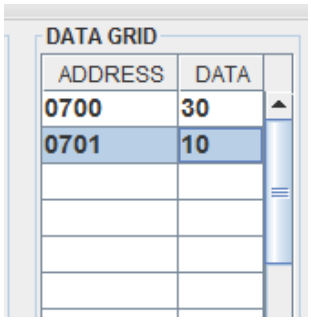
Aim: Subtract the contents of memory location 0701H from the memory location 0700H and place the result in memory location 0702H.

Source program:

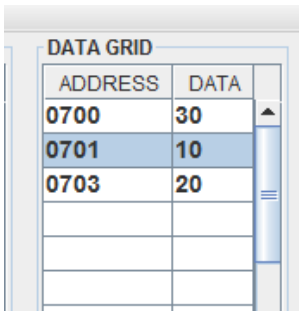
```
LDA 0701H       : "HL points 0700H"
MOV B, A        : "copy in reg. B"
LDA 0700H       : "HL points 0700H"
SUB B           : "Subtract second operand"
STA 0703H
HLT             : "Terminate program execution"
```

Output:

Before executing program type the following data in data grid



After executing this program we will get



Practical No. 4

Aim: Add the 16-bit number in memory locations 0700H and 0701H to the 16-bit number in memory locations 0702H and 0703H. Store the result in memory locations 0704H and 0705H with the most significant byte in memory location 4005H.

LHLD 0700H : Get first I6-bit number

XCHG : Save first I6-bit number in DE

LHLD 0702H : Get second I6-bit number in HL

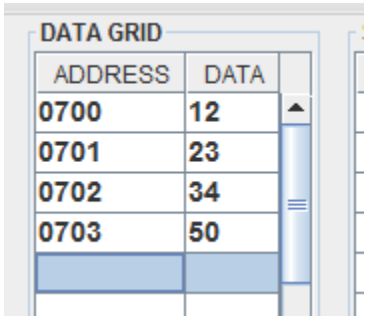
DAD D : Add DE and HL

SHLD 0704H : Store I6-bit result in memory locations 0704H and 0705H.

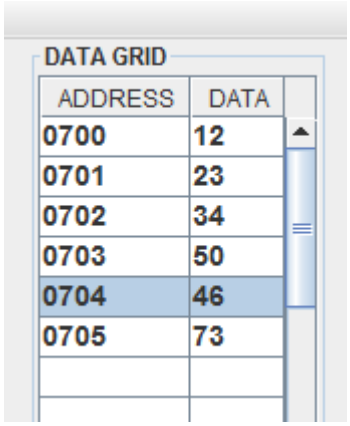
HLT : Terminate program execution

Output:

Before executing program type the following data in data grid



After executing this program we will get


DATA GRID
ADDRESS
DATA
0700
12
0701
23
0702
34
0703
50
0704
46
0705
73

Practical No. 5

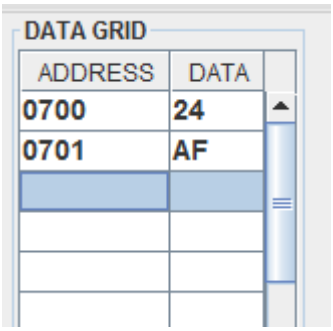
Aim: Add the contents of memory locations 0700H and 0701H and place the result in the memory locations 0702Hand 0703H.

Source program:

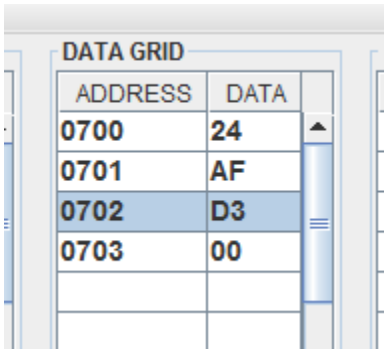
```
LXI H, 0700H      : "HL Points 0700H"
MOV A, M          : "Get first operand"
INX H             : "HL Points 0701H"
ADD M             : "Add second operand"
INX H             : "HL Points 0702H"
MOV M, A          : "Store the lower byte of result at 0702H"
MVIA, 00          : "Initialize higher byte result with 00H"
ADC A             : "Add carry in the high byte result"
INX H             : "HL Points 0703H"
MOV M, A          : "Store the higher byte of result at 0703H"
HLT               : "Terminate program execution"
```

Output:

Before executing program type the following data in data grid



After executing this program we will get



Practical No. 6

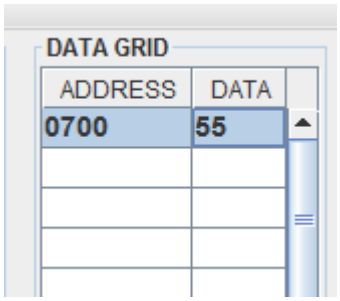
Aim: Find the l's complement of the number stored at memory location 0700H and store the complemented number at memory location 0701H.

Source program:

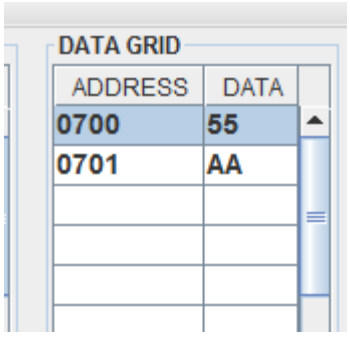
```
LDA 0700H      : "Get the number"
CMA            : "Complement number"
STA 0701H      : "Store the result"
HLT            : "Terminate program execution"
```

Output:

Before executing program type the following data in data grid



After executing this program we will get



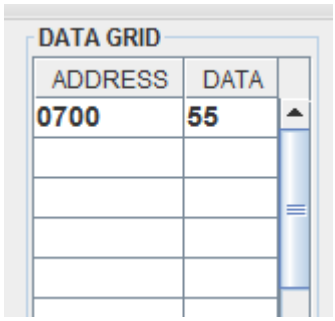
Practical No. 7

Aim: Find the 2's complement of the number stored at memory location 0700H and store the complemented number at memory location 0701H

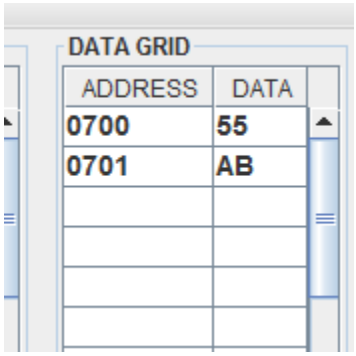
LDA 0700H	: "Get the number"
CMA	: "Complement number"
ADI 01H	:Add one to get 2's complement
STA 0701H	: "Store the result"
HLT	: "Terminate program execution"

Output:

Before executing program type the following data in data grid



After executing this program we will get



Practical No. 8.

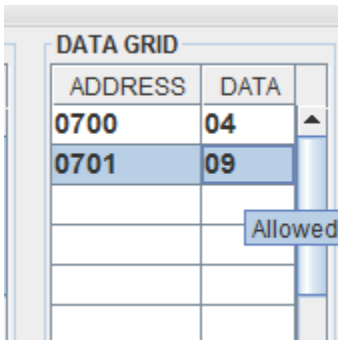
Aim: Pack the two unpacked BCD numbers stored in memory locations 0700H and 0701H and store result in memory location 0703H. Assume the least significant digit is stored at 0700H.

Source program:

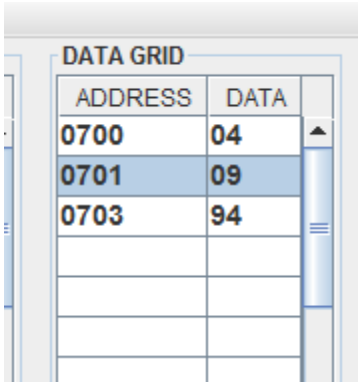
```
LDA 0701H     : "Get the Most significant BCD digit"
RLC
RLC
RLC
RLC           : "Adjust the position of the second digit (09 is changed to 90)"
ANI F0H       : "Make least significant BCD digit zero"
MOV C, A      : "store the partial result"
LDA 0700H     : "Get the lower BCD digit"
ADD C         : "Add lower BCD digit"
STA 0703H     : "Store the result"
HLT           : "Terminate program execution"
```

Output:

Before executing program type the following data in data grid



After executing this program we will get



Practical No. 9

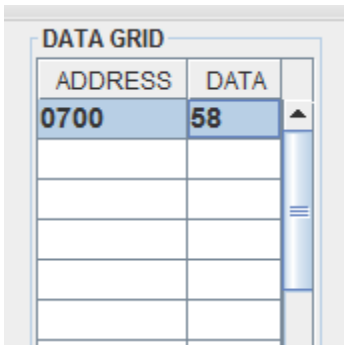
Aim: Two digit BCD number is stored in memory location 0700H. Unpack the BCD number and store the two digits in memory locations 0701H and 0702H such that memory location 0701H will have lower BCD digit.

Source program:

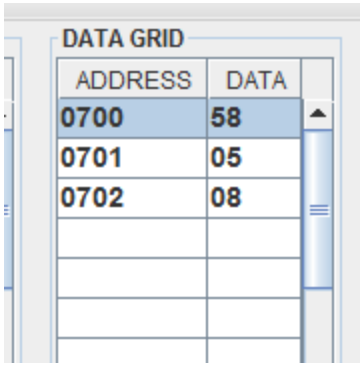
```
LDA 0700H      : "Get the packed BCD number"
ANI F0H        : "Mask lower nibble"
RRC
RRC
RRC
RRC            : "Adjust higher BCD digit as a lower digit"
STA 0701H      : "Store the partial result"
LDA 0700H      : "Get the original BCD number"
ANI 0FH        : "Mask higher nibble"
STA 0702H      : "Store the result"
HLT            : "Terminate program execution"
```

Output:

Before executing program type the following data in data grid



After executing this program we will get



Practical No. 10

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

Sample problem:

C = 58

Result C = 05

Source program 1:

```
MVI C, 58H
MOV A, C
RAR
RAR
RAR
RAR
MOV C, A
HLT
```

Practical No. 11.

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

STC : Setting carry Flag

SUB A : Setting Zero Flag

ADI 03H : Setting parity

SUI 01H : Setting Sign

Practical No. 12.

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

Sample problem

```
MVI D, 56H

MVI B, 00H

MVI C, 08H

MOV A, D

RAR

JNC 200C

INR B

DCR C

JNZ  2007H

HLT
```

Practical No. 13

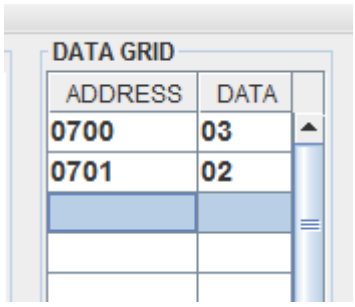
Aim: Multiply two 8-bit numbers stored in memory locations 0700H and 0701H by repetitive addition and store the result in memory locations 0702H and 0703H

Source program

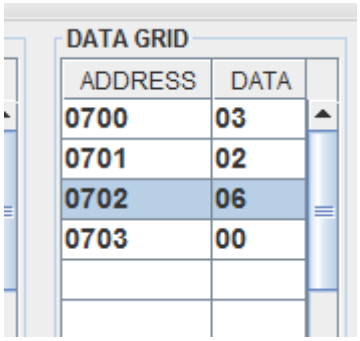
```
LDA 0700H
MOV E, A
MVI D, 00          : Get the first number in DE register pair
LDA 0701H
MOV C, A           : Initialize counter
LX I H, 0000 H     : Result = 0
DAD   D            : Result = result + first number
DCR   C            : Decrement count
JNZ   200DH        : If count  0 repeat
SHLD 0702H         : Store result
HLT                : Terminate program execution
```

Output:

Before executing program type the following data in data grid



After executing this program we will get


DATA GRID
ADDRESS
DATA
0700
03
0701
02
0702
06
0703
00

Practical No. 14.

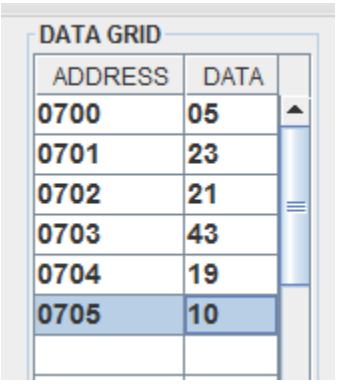
Aim: Find the largest number in a block of data. The length of the block is in memory location 0700H and the block itself starts from memory location 0701H. Store the maximum number in memory location 070FH. Assume that the numbers in the block are all 8 bit unsigned binary numbers.

Source program

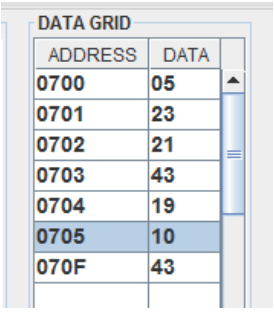
```
LDA 0700H
MOV C, A          : Initialize counter
XRA A             : Maximum = Minimum possible value = 0
LXI H, 0701H      : Initialize pointer
CMP M             : Is number> maximum
JNC 200DH         : Yes, replace maximum
MOV A, M
INX H
DCR C
JNZ 2008H
STA 070FH         : Store maximum number
HLT               : Terminate program execution
```

Output:

Before executing program type the following data in data grid



After executing this program we will get



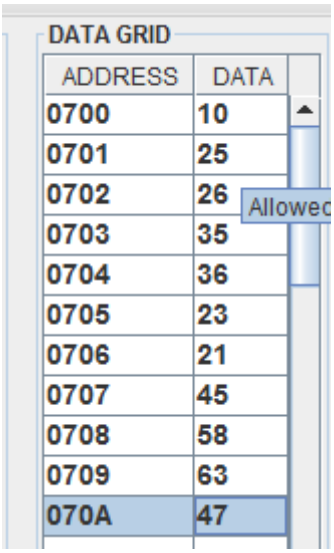
Practical No. 15.

Aim: Search the given byte in the list of 10 numbers stored in the consecutive memory locations and store the address of memory location in the memory locations 070EH AND 070FH . Assume byte is in the C register and starting address of the list is 0700H. If byte is not found store 00 at 2200H and 2201H.

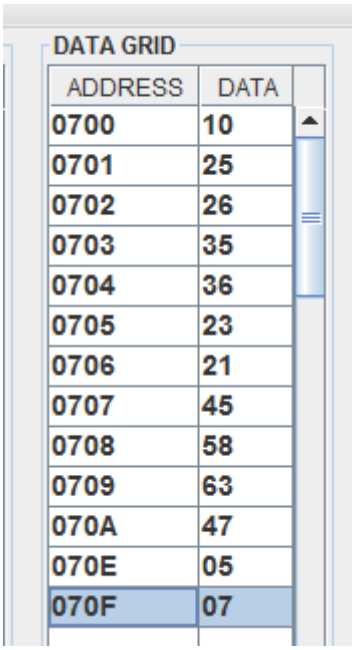
```
MVI   C     23H
LXI   H    0700H     :"Initialize memory pointer TO 0700H"
MVI   B     0AH     :"Initialize counter"
MOV   A    M        :"Get the number"
CMP    C             :"Compare with the given byte"
JZ  2018H            :"Go last if match occurs"
INX  H 23H           :"Increment memory pointer"
DCR    B             :"Decrement counter"
JNZ  2005H           :"If not zero, repeat"
LXI   H    0000H
SHLD  070EH       :"Store 00 at 070EH and 070FH"
JMP  201BH
SHLD  070EH       :"Store memory address"
HLT                  :"Stop"
```

Output:

Before executing program type the following data in data grid



After executing this program we will get



Practical No. 16.

Aim: Write an assembly language program to generate fibonacci number.

```
MVI D, 10H      : 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]
  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 2007H       : if count  0 go to BACK
HLT             : Stop.
```

Practical No. 17.

Aim: Transfer ten bytes of data from one memory to another memory block. Source memory block starts from memory location 0700H where as destination memory block starts from memory location 0710H

Source Program:

```
        LXI H, 0700H      : Initialize memory pointer
      LXI D, 0710H
      MVI C, 0AH
        LOOP: MOV A,M

        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
```

Practical No. 18.

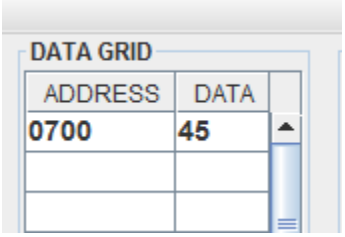
Aim: Write a simple program to Split a HEX data into two nibbles and store it in memory

Source Program:

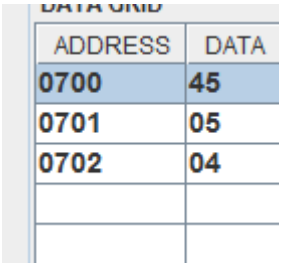
```
LXI H, 0700H        : Set pointer data for array
MOV B,M             : Get the data in B-reg
MOV A,B             : Copy the data to A-reg
ANI OFH             : Mask the upper nibble
INX H               : Increment address as 4201
MOV M,A             : Store the lower nibble in memory
MOV A,B             : 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
```

Output:

Before executing program type the following data in data grid



After executing this program we will get



Practical No. 19.

Aim: Add two 4 digit BCD numbers in HL and DE register pairs and store result in memory locations, 0700H and 0701H. 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

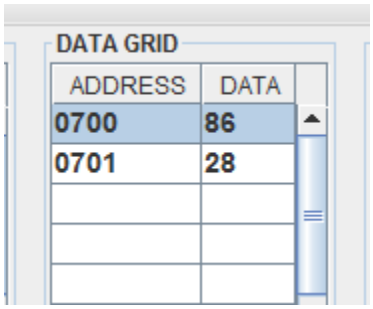
Lower nibble of addition is greater than 9, so add 6.
7D + 06 = 83
Result = 8367

Source program

```
LXI H, 2629H
LXI D, 4738H
MOV A, L        : Get lower 2 digits of no. 1
ADD E           : Add two lower digits
DAA             : Adjust result to valid BCD
STA 0700H       : 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 0701H       : Store partial result
HLT             : Terminate program execution.
```

Output:

After executing this program we will get



Practical No. 20

Aim: Subtract the BCD number stored in E register from the number stored in the D register

Source Program:

```
MVI E, 30H
MVI D, 25H
    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
```