M.I.T. LAB Assignment – 05

U19CS012

(1) The following block of data is stored in memory locations from 3055H to 305AH. WAP to transfer the block of data in reverse order at same memory location.

DATA (HEX): 22, A5, B2, 99, 7F, 37

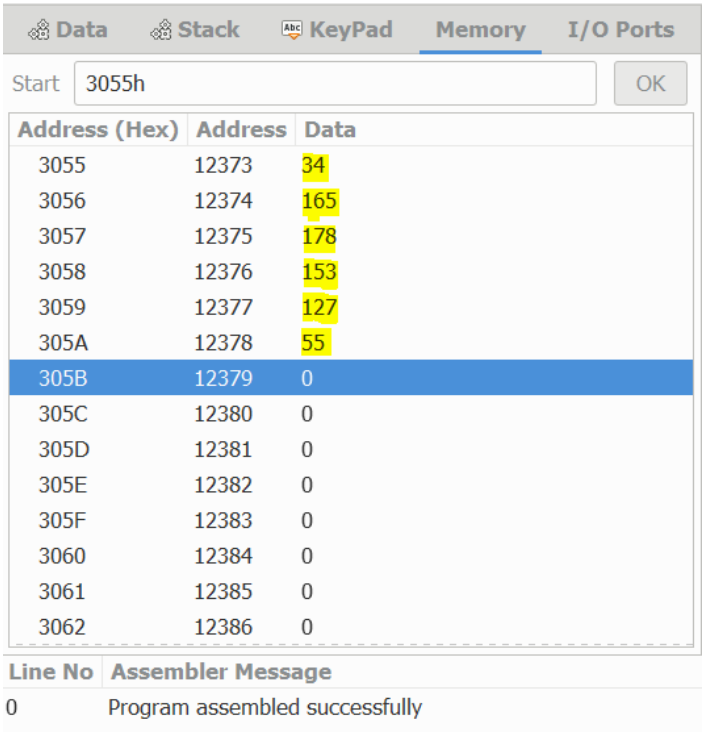
Notepad Code:

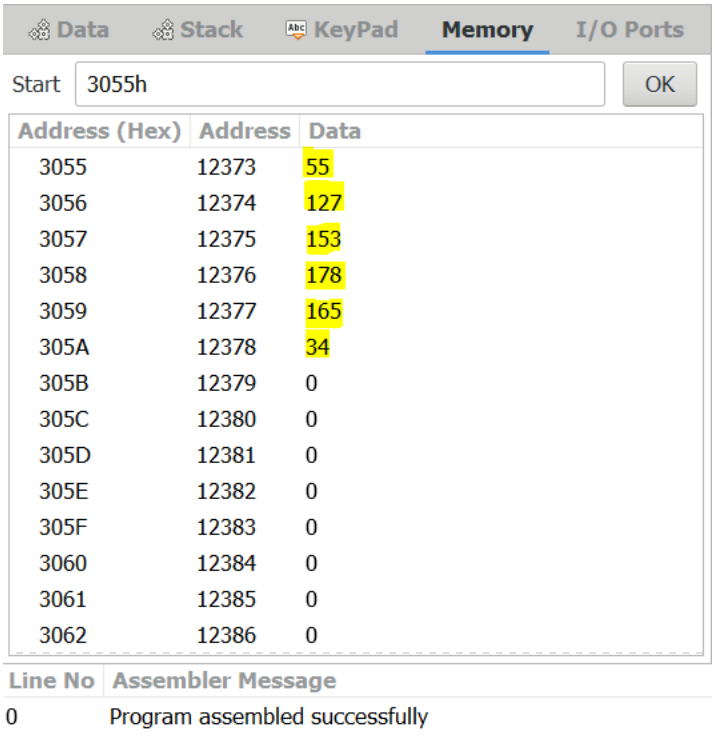
```
; Point HL to First Location
LXI B, 3055H
; Intialise the Counter to 6 [Since 6 Numbers]
MVI H, 06H

; COPY IT TO TEMOPARY LOCATION SERIALLY

; Point DE to Location 2000H
LXI D, 2000H
LOOP: LDAX B ; A <- (BC)
    STAX D ; (DE) <- A
    INX B ; i++
    INX D ; j++
    DCR H ;Decrease the Counter
    JNZ LOOP

; COPY FROM TEMOPARY LOCATION back to Same Location [3055-305A]
; IN REVERSE ORDER

DCX D ; DE Pair is pointing to Last Location
LXI B, 3055H
MVI H, 06H

LOOP2: LDAX D ; A <- (DE)
    STAX B ; (BC) <- A
    DCX D  ; j--
    INX B  ; i++
    DCR H  ; Decrement Counter
    JNZ LOOP2
HLT
```

Test Case:

Input Data: 34, 165, 178, 153, 127, 55 [Decimal]

Output Data: 55, 127, 153, 178, 165, 36 [Decimal]





(2) Find the square of the given numbers from memory location 6100H and store the result from memory location 7000H.

```
4   ; INPUT
5   ; Location 60FFH : 'n' : Number of Elements
6   ; Followed by 'n' Numbers from : (6100)H to (6100+n)H
7   ; LIMITATION: Only Numbers between [0-15] Accepted
8
9   ; OUTPUT
10  ; Square of Numbers are Stored from Location : (7000)H to (7000+n)H
```

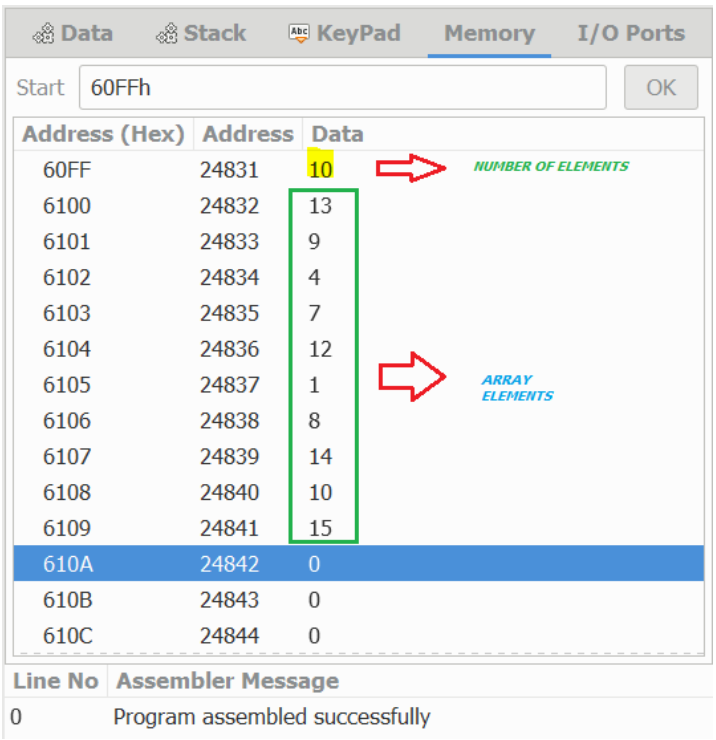
Notepad Code:

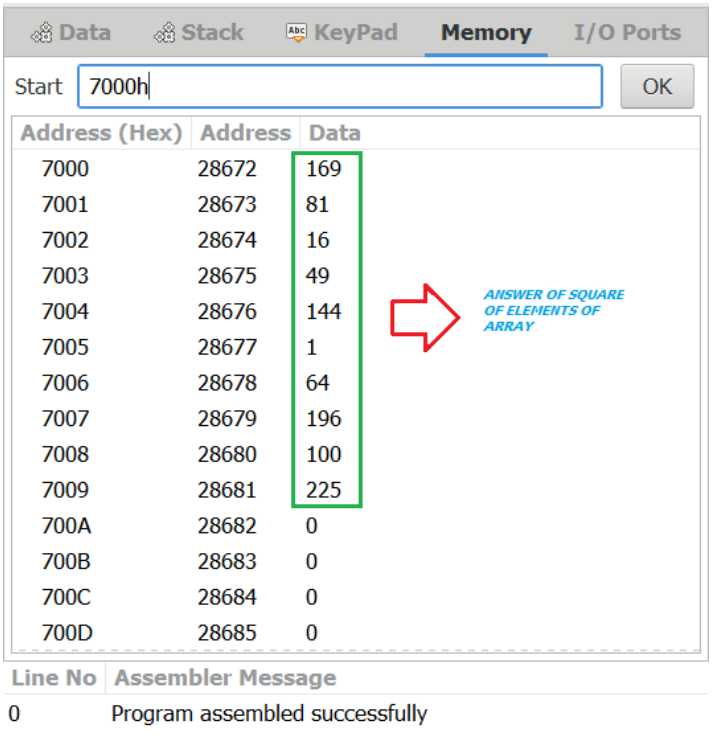
```
12  LDA 60FFH ; Input 'n'
13  ; Store 'n' in Register C
14  MOV C, A
15
16  ; Load the First Number
17  LXI  H, 6100H
18  ; Load the Place where Square of Numbers need to be stored
19  LXI  D, 7000H
20
21  LOOP: MVI A, 00H ; Intialise Accumulator
22        MOV B, M ; Copy the Number to Register 'B'
23
24  ; Squaring : Adding the Number 'n' , n times
25  SQUARE: ADD M
26            DCR B
27            ; Until B Reaches 0
28            JNZ SQUARE
29
30  ; Store the Result in DE Register Pair
31  STAX D
32  INX H   ; Next Number in Array
33  INX D   ; Next Destination Location
34
35  DCR C
36  JNZ LOOP ; Until All Elements of Array are Traversed
37
38  HLT
```

Test Case:

Input Data: 13, 9, 4, 7, 12, 1, 8, 14, 10, 15 [Decimal]

Output Data: 169, 81, 16, 49, 144, 1, 64, 196, 100,225 [Decimal]





(3) WAP to find Factorial of a given number using Call and Subroutine.

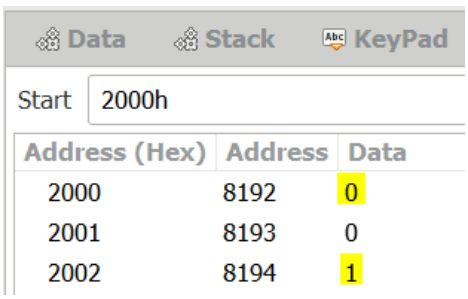
```
; Input   : Enter a Number 'n' at Location 2000H
; Output  : Output the answer of 'n!' at Location 2002H
```

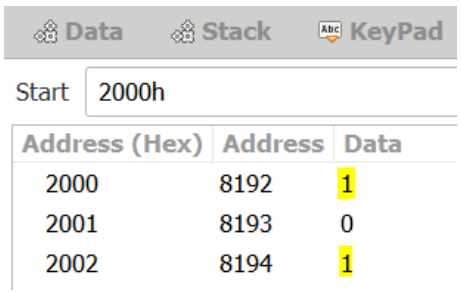
Notepad Code:

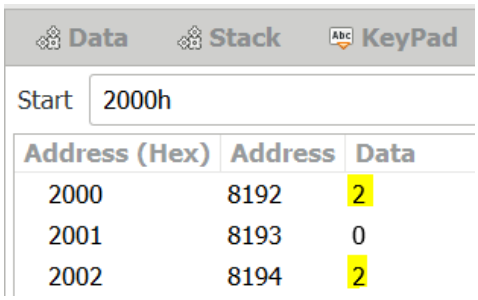
```
; Load 'n' From Location 2000H
LDA 2000H
; Check if Number is Less than 2 [0!=1!=1]
CPI 02H
JC zerofac

MOV B, A   ; Copy 'n' to B = Counter
LXI D, 01H ; DE Register Pair Stores Our Factorial Answer

LOOP: CALL factorial ; Call the Sub-Routine [Function]
        DCR B
        JNZ LOOP
; After the Loop Ends, HL Rp Will have Answer
SHLD 2002H
HLT

zerofac: LXI H, 0001H ; Store Result = 1 = 0! = 1!
          SHLD 2002H
HLT

; Factorial Function [Sub-Routine] ; [DE] <- [DE] * [B]
; Multiply Reg B [Value] with "DE" Rp and Store the Result Back in "DE" Rp

factorial: MOV A, B
           LXI H, 00H
            UP: DAD D
                DCR A
                JNZ UP
            ; Exhange HL Rp <-> DE Rp
            XCHG
RET ; Return after Function Call
```

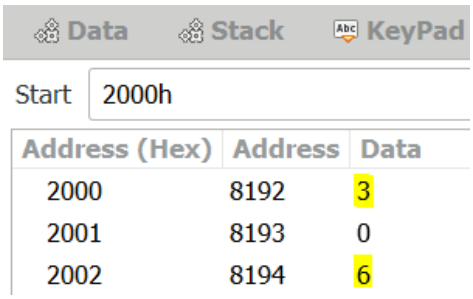
Test Case:

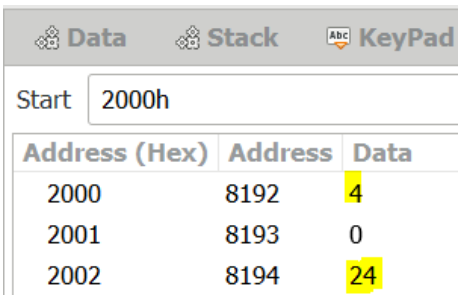
Number (n)	Factorial (n!) [Decimal]	Factorial (n!) [Hexa-Decimal]
0	[0 1]	(00 01)H
1	[0 1]	(00 01)H
2	[0 2]	(00 02)H
3	[0 6]	(00 06)H
4	[0 24]	(00 18)H
5	[0 120]	(00 78)H
6	[2 208]	(02 D0)H
7	[19 176]	(13 B0)H
8	[157 128]	(9D 80)H

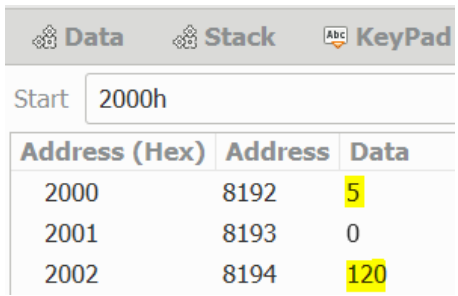


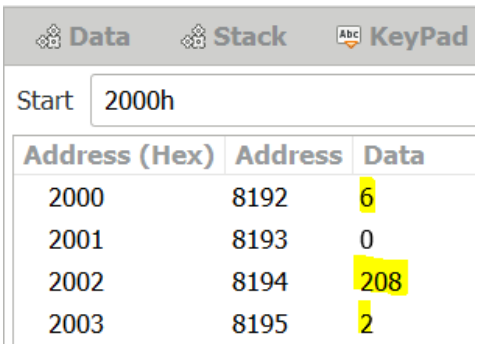


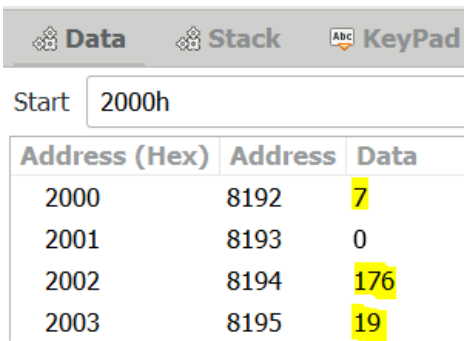


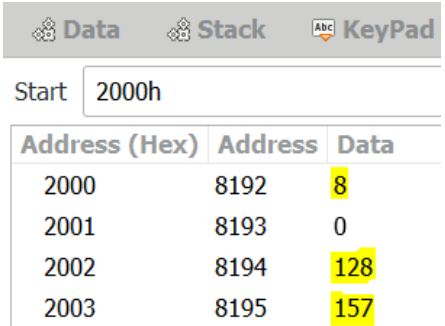












(4) WAP for Fibonacci Series using Call and Subroutine.

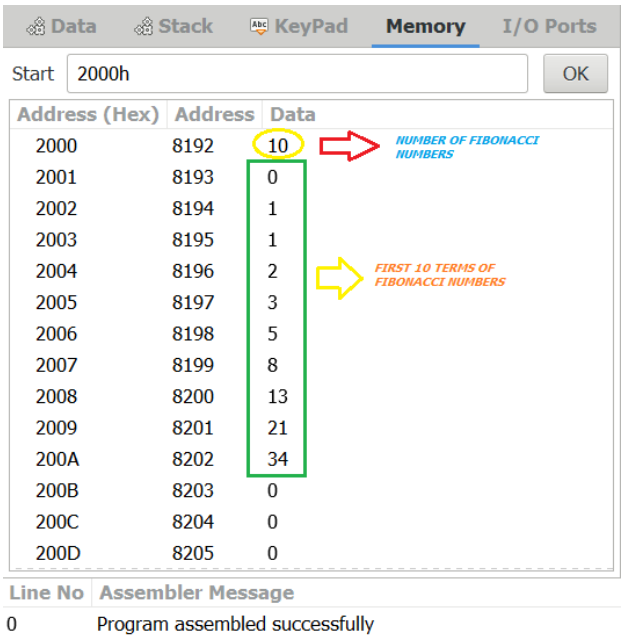
```
; Input (Location 2000H)   : Limit 'n' Upto Which Fibonaaci Number is to be found
; Output (Location 2001H Onwards) : [0, 1, 1, 2, 3, 5, 8, 13, 21, … ]
; LIMITATION:   n <= 14
```

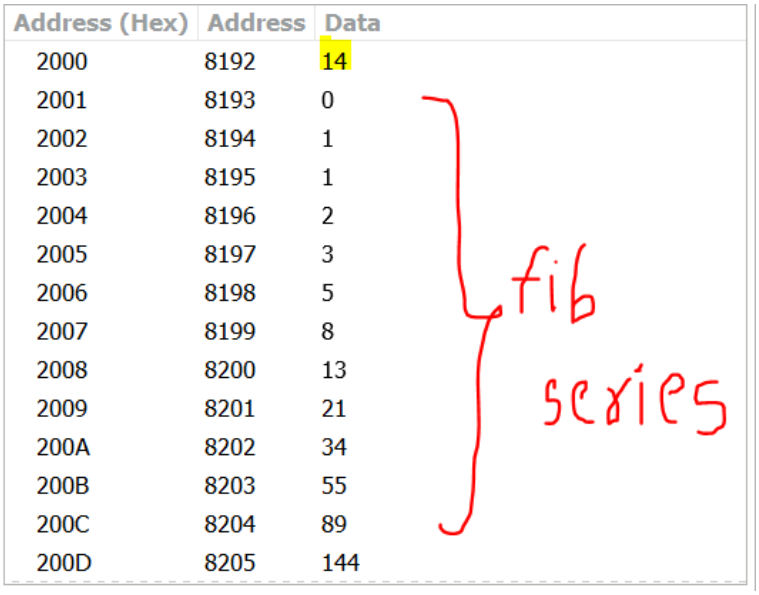
Notepad Code:

```
LDA 2000H       ; Input the Limit [n] Upto Which Fibonacci Number Need to be Printed
DCR A           ; First Fibonacci is 0
MOV E,A         ; Copy 'n-1' to E
CALL fibonacci  ; CALL the Sub-Routine
HLT

fibonacci: LXI H, 2002H      ; HL = 3000H
           MVI M, 01H        ; M = 01 [2nd Element]
           MOV B, M          ; B = 01
           DCR E             ; E--
           JZ return         ; IF Z==1 => Return
           INX H             ; HL Points Next Location
           MVI M, 01H        ; M = 01 [3rd Element]
           MOV C, M          ; C = M
           DCR E             ; E--
           MVI A, 00H        ; A = 00H
LOOP:      ADD B             ; A = A + B
           ADD C             ; A = A + C
           ; Sum of Previous Two Terms
           INX H             ; HL Points Next Location
           MOV M, A          ; M = A
           MOV B, C          ; B = C [Prev]
           MOV C, A          ; C = A [Curr]
           MVI A, 00H        ; A = 00H
           DCR E             ; E--
           JNZ LOOP          ; IF Z != 0 => LOOP
return: RET                  ; End of Sub-Routine
```

Test Case:

First 10 & 14 Numbers of Fibonacci Series





(5) WAP to find Multiplication of Two 8-Bit Numbers using Call and Subroutine.

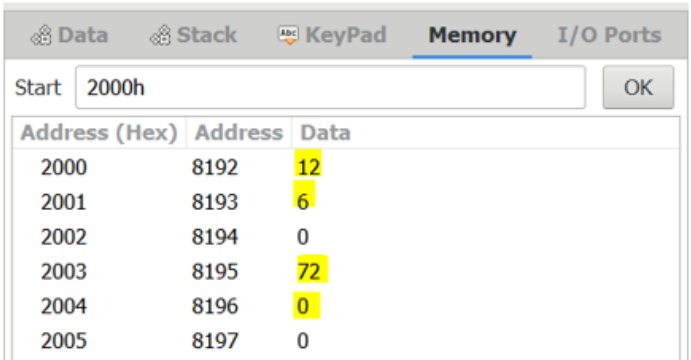
```
; INPUT: Enter Two Numbers to Multiply at Location 2000H & 2001H
; OUTPUT: Result is Shown at Two Location (2004)H [AA] & (2003)H [BB]  [AA BB]H
```

Notepad Code:

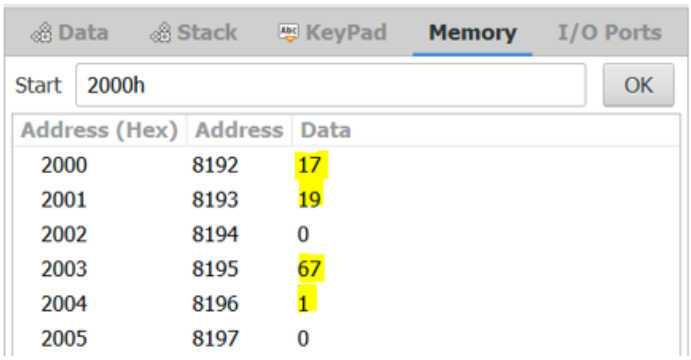
```
7   LXI H, 2000H
8   MOV B, M     ; First Number = Register B
9   INX H
10  MOV C, M     ; Second Number = Register C
11
12  MVI E, 00H
13
14  CALL MULTIPLY
15
16  INX H        ; Extra Space
17  ; Final Result
18  INX H        ; 2003H
19  MOV M, A
20  INX H
21  MOV M, E     ; 2004H
22  HLT
23
24  ; B*C = Adding 'B', C times to Accumulator
25  MULTIPLY: MVI A, 00H
26            LOOP: ADD B
27                  JNC SKIP ; If No Carry, Jump to SKIP
28                  INR E    ; Add it to Carry
29            SKIP: DCR C    ; Decrement the Counter
30                  JNZ LOOP
31            RET      ; Return from Sub-Routine
```

Test Case:

(1) 12 * 6 = 72



(2) 17*19 = 323 = (01 43)H = (01 67)



SUBMITTED BY:

BHAGYA VINOD RANA

[U19CS012]