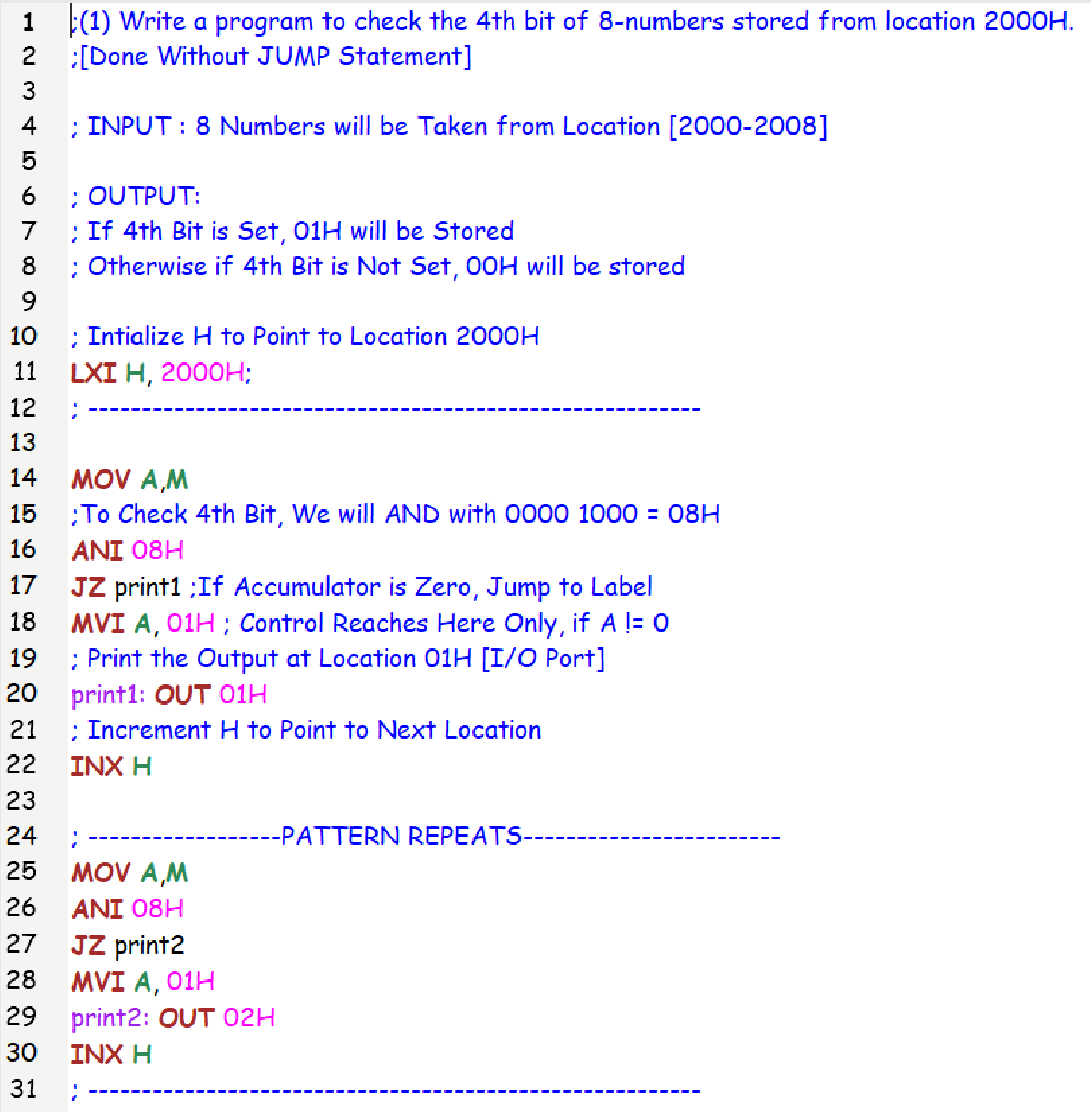
M.I.T. LAB Assignment – 04

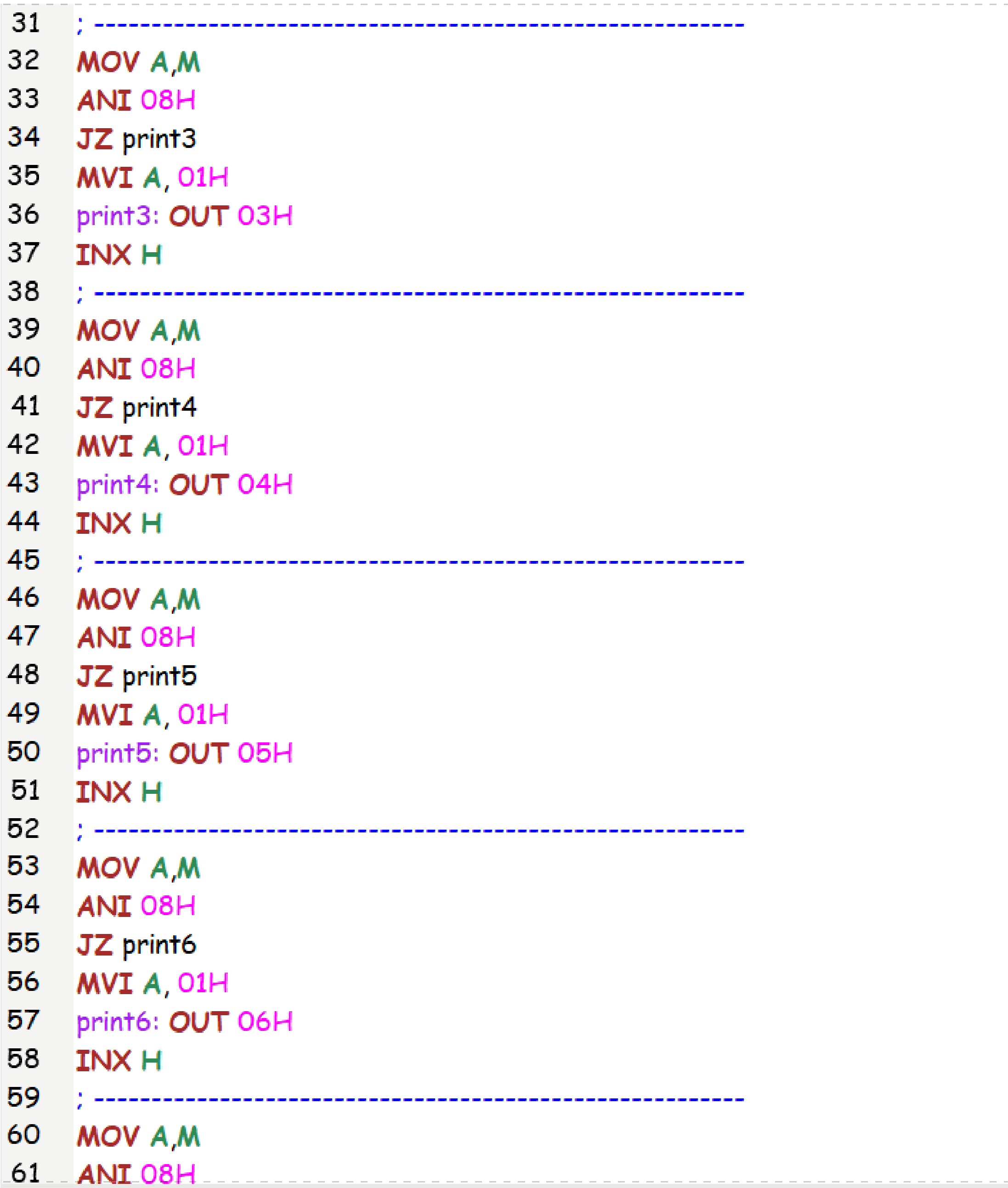
U19CS012

(1) Write a program to check the 4th bit of 8-numbers stored from location 2000H.

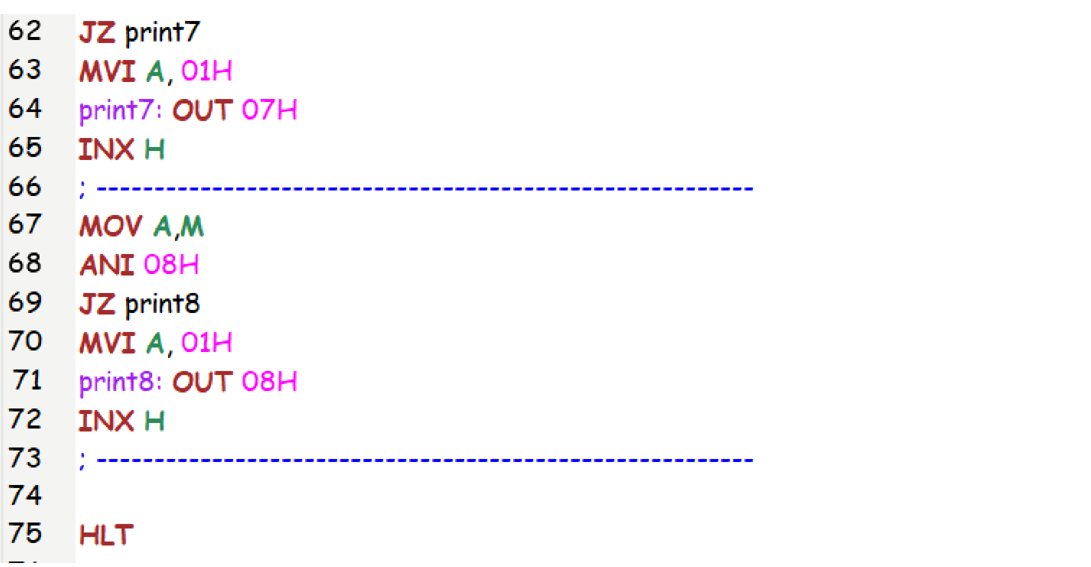
Notepad Code:



```
;(1) Write a program to check the 4th bit of 8-numbers stored from location 2000H.
;[Done Without JUMP Statement]

; INPUT : 8 Numbers will be Taken from Location [2000-2008]

; OUTPUT:
; If 4th Bit is Set, 01H will be Stored
; Otherwise if 4th Bit is Not Set, 00H will be stored

; Intialize H to Point to Location 2000H
LXI H, 2000H;
; -------------------------------------------------------

MOV A,M
;To Check 4th Bit, We will AND with 0000 1000 = 08H
ANI 08H
JZ print1 ;If Accumulator is Zero, Jump to Label
MVI A, 01H ; Control Reaches Here Only, if A != 0
; Print the Output at Location 01H [I/O Port]
print1: OUT 01H
; Increment H to Point to Next Location
INX H

; ------------------PATTERN REPEATS------------------------
MOV A,M
ANI 08H
JZ print2
MVI A, 01H
print2: OUT 02H
INX H
; -------------------------------------------------------
```



```
; --------------------------------------------------------
MOV A,M
ANI 08H
JZ print3
MVI A, 01H
print3: OUT 03H
INX H
; --------------------------------------------------------
MOV A,M
ANI 08H
JZ print4
MVI A, 01H
print4: OUT 04H
INX H
; --------------------------------------------------------
MOV A,M
ANI 08H
JZ print5
MVI A, 01H
print5: OUT 05H
INX H
; --------------------------------------------------------
MOV A,M
ANI 08H
JZ print6
MVI A, 01H
print6: OUT 06H
INX H
; --------------------------------------------------------
MOV A,M
ANI 08H
```

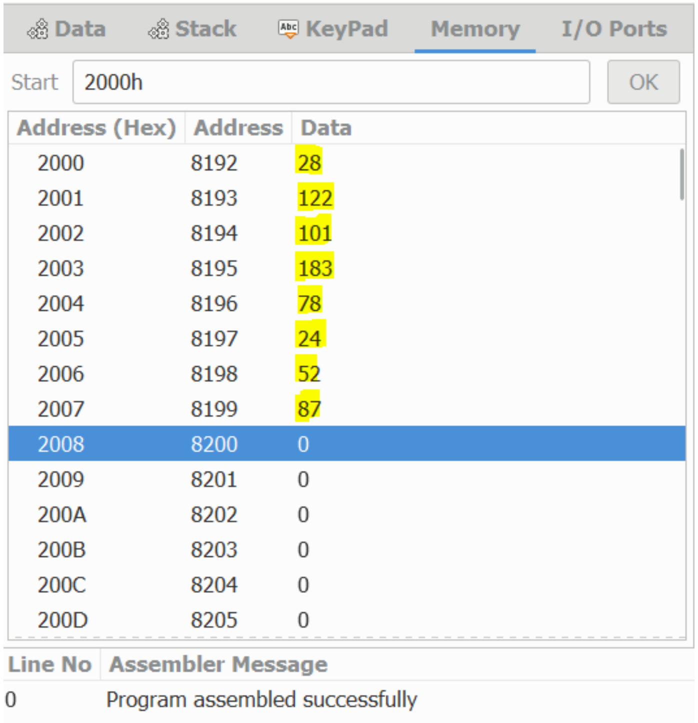


```
JZ print7
MVI A, 01H
print7: OUT 07H
INX H
; --------------------------------------------------------
MOV A,M
ANI 08H
JZ print8
MVI A, 01H
print8: OUT 08H
INX H
; --------------------------------------------------------

HLT
```

Registers and Memory:

Location	Hexadecimal	Decimal	Binary	Location	4th Bit	Output
2000H	1C	28	0001 1100	3000H	1	1
2001H	7A	122	0111 1010	3001H	1	1
2002H	65	101	0110 0101	3002H	0	0
2003H	B7	183	1011 0111	3003H	0	0
2004H	4E	78	0100 1110	3004H	1	1
2005H	18	24	0001 1000	3005H	1	1
2006H	34	52	0011 0100	3006H	0	0
2007H	57	87	0101 0111	3007H	0	0

INPUT

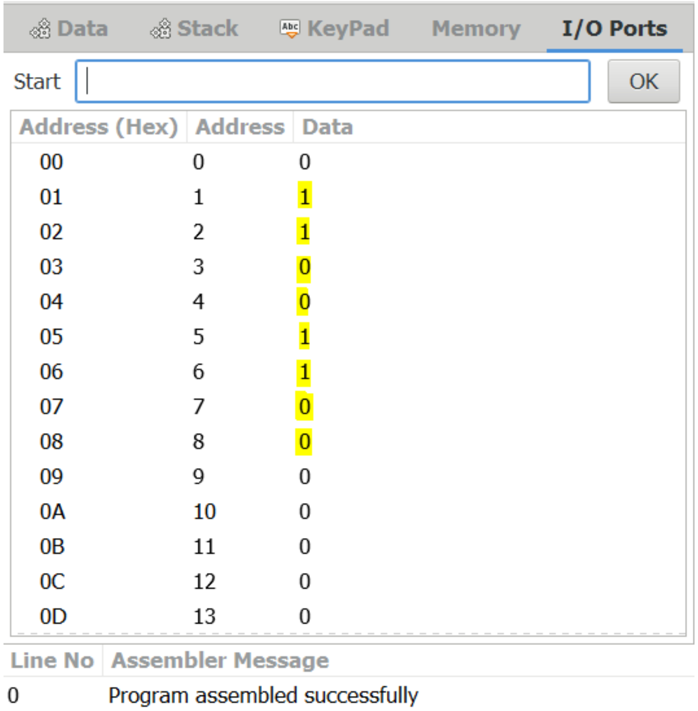


Start 2000h

Address (Hex)	Address	Data
2000	8192	28
2001	8193	122
2002	8194	101
2003	8195	183
2004	8196	78
2005	8197	24
2006	8198	52
2007	8199	87
2008	8200	0
2009	8201	0
200A	8202	0
200B	8203	0
200C	8204	0
200D	8205	0

Line No	Assembler Message
0	Program assembled successfully

OUTPUT



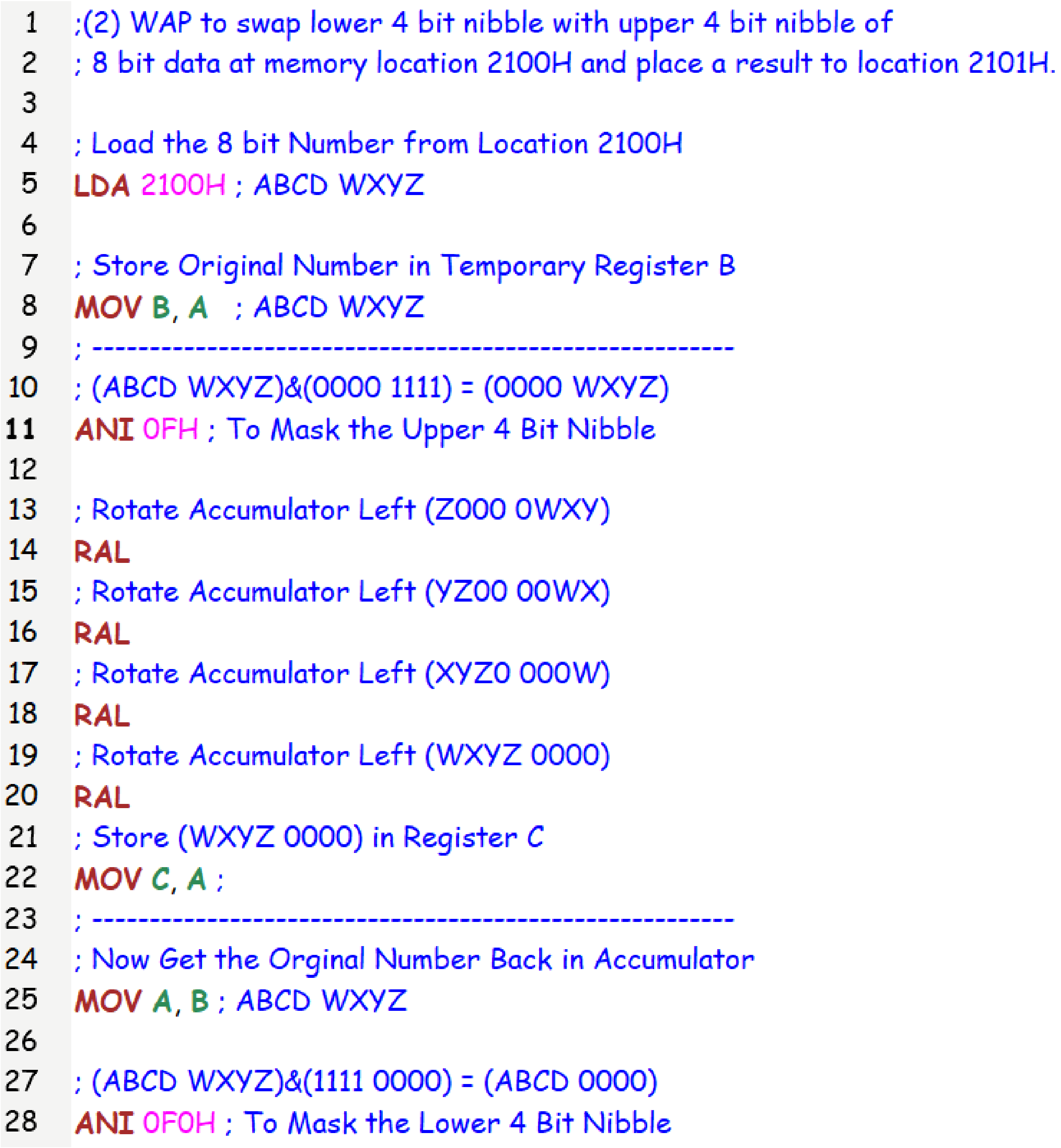
Start

Address (Hex)	Address	Data
00	0	0
01	1	1
02	2	1
03	3	0
04	4	0
05	5	1
06	6	1
07	7	0
08	8	0
09	9	0
0A	10	0
0B	11	0
0C	12	0
0D	13	0

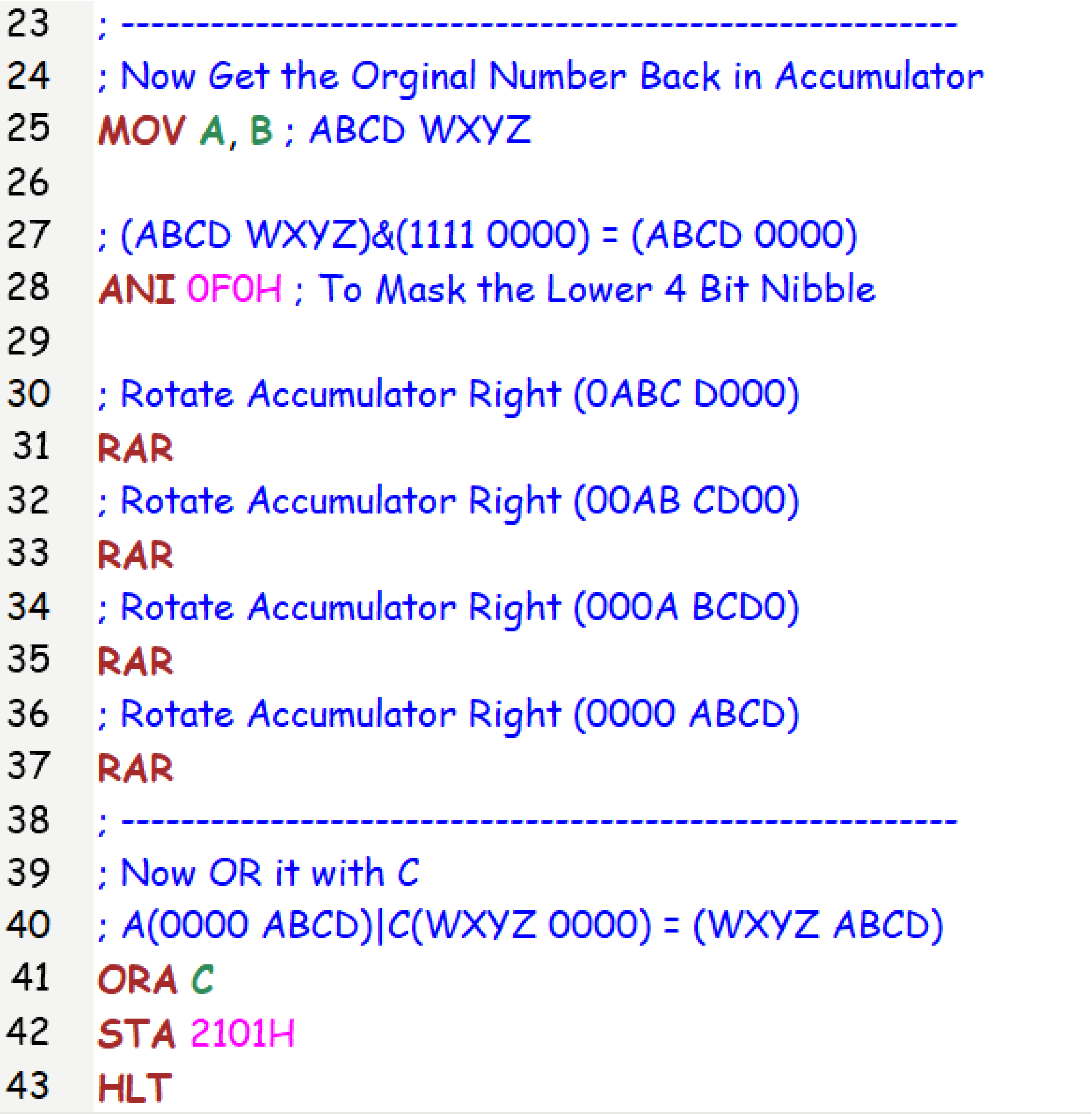
Line No	Assembler Message
0	Program assembled successfully

(2) Write a program to swap lower 4 bit nibble with upper 4 bit nibble of 8 bit data at memory location 2100H and place a result to location 2101H.

Notepad Code:



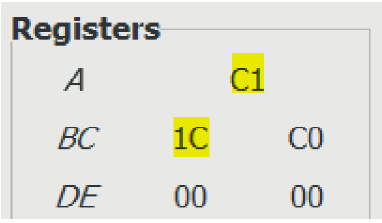
```
;(2) WAP to swap lower 4 bit nibble with upper 4 bit nibble of
; 8 bit data at memory location 2100H and place a result to location 2101H.

; Load the 8 bit Number from Location 2100H
LDA 2100H ; ABCD WXYZ

; Store Original Number in Temporary Register B
MOV B, A   ; ABCD WXYZ
; ----------------------------------------------------
; (ABCD WXYZ)&(0000 1111) = (0000 WXYZ)
ANI 0FH ; To Mask the Upper 4 Bit Nibble

; Rotate Accumulator Left (Z000 0WXY)
RAL
; Rotate Accumulator Left (YZ00 00WX)
RAL
; Rotate Accumulator Left (XYZ0 000W)
RAL
; Rotate Accumulator Left (WXYZ 0000)
RAL
; Store (WXYZ 0000) in Register C
MOV C, A ;
; ----------------------------------------------------
; Now Get the Orginal Number Back in Accumulator
MOV A, B ; ABCD WXYZ

; (ABCD WXYZ)&(1111 0000) = (ABCD 0000)
ANI 0F0H ; To Mask the Lower 4 Bit Nibble
```



```
; --------------------------------------------------------
; Now Get the Orginal Number Back in Accumulator
MOV A, B ; ABCD WXYZ

; (ABCD WXYZ)&(1111 0000) = (ABCD 0000)
ANI 0F0H ; To Mask the Lower 4 Bit Nibble

; Rotate Accumulator Right (0ABC D000)
RAR
; Rotate Accumulator Right (00AB CD00)
RAR
; Rotate Accumulator Right (000A BCD0)
RAR
; Rotate Accumulator Right (0000 ABCD)
RAR
; --------------------------------------------------------
; Now OR it with C
; A(0000 ABCD)|C(WXYZ 0000) = (WXYZ ABCD)
ORA C
STA 2101H
HLT
```

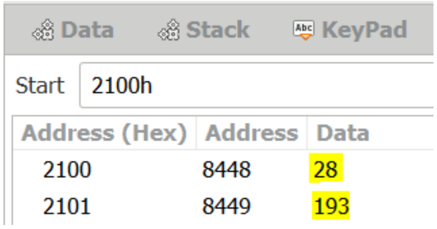
Registers and Memory:

Location	Hexadecimal	Decimal	Location	Hexadecimal	Output
2100H	1C	28	2101H	C1	193
2100H	7A	122	2101H	A7	167
2100H	65	101	2101H	56	86
2100H	B7	183	2101H	7B	123
2100H	4E	78	2101H	E4	228



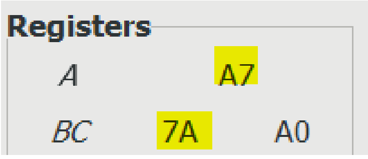
Registers

A	C1	
BC	1C	C0
DE	00	00



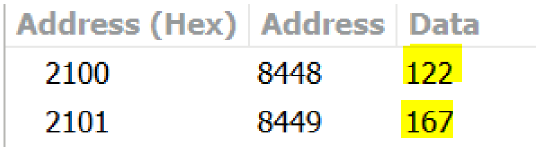
Start 2100h

Address (Hex)	Address	Data
2100	8448	28
2101	8449	193

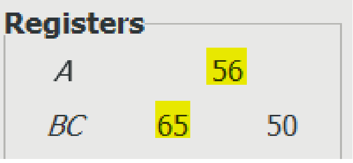


Registers

A	A7	
BC	7A	A0

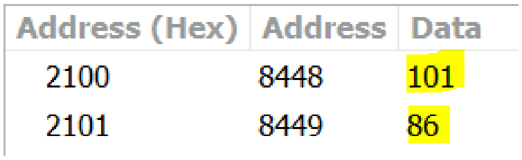


Address (Hex)	Address	Data
2100	8448	122
2101	8449	167

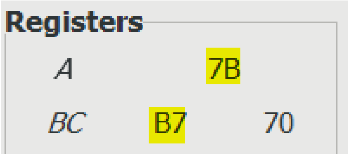


Registers

A	56	
BC	65	50

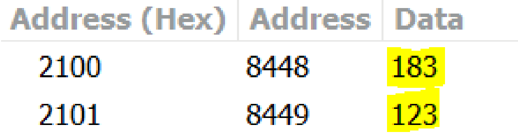


Address (Hex)	Address	Data
2100	8448	101
2101	8449	86

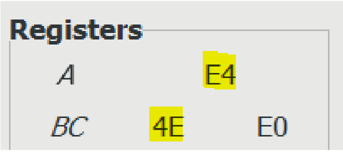


Registers

A	7B	
BC	B7	70

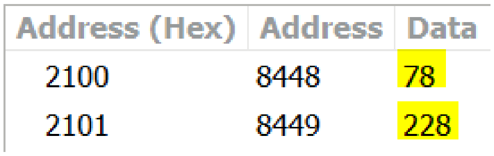


Address (Hex)	Address	Data
2100	8448	183
2101	8449	123



Registers

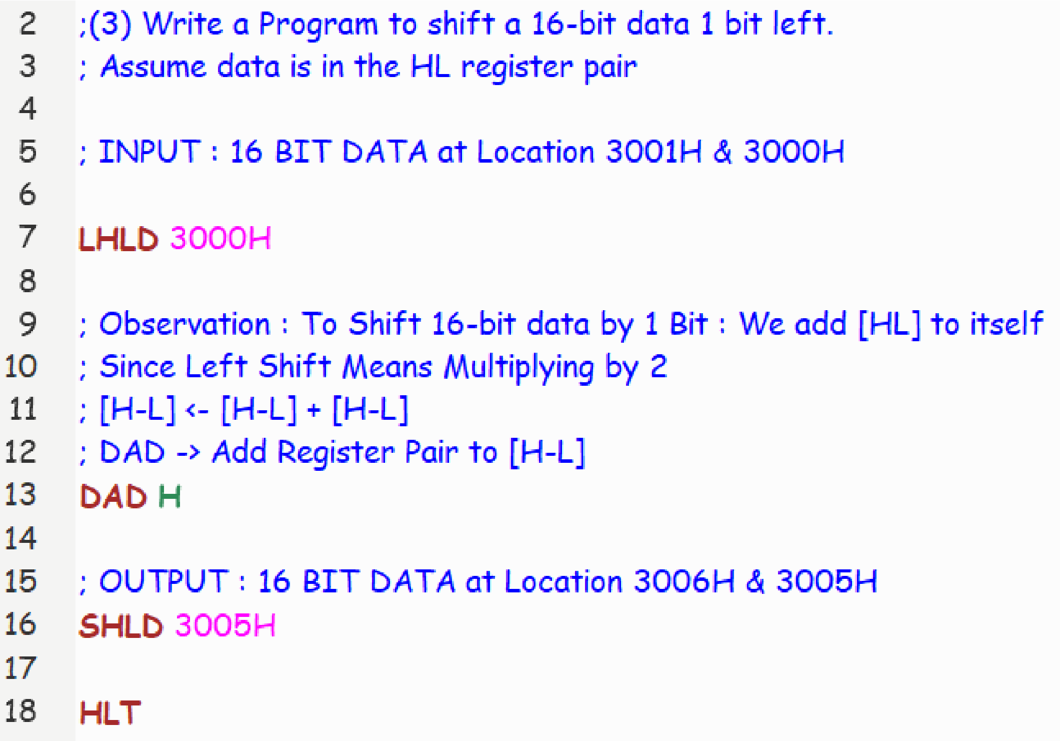
A	E4	
BC	4E	E0



Address (Hex)	Address	Data
2100	8448	78
2101	8449	228

(3) Write a Program to shift a 16-bit data 1 bit left. Assume data is in the HL register pair

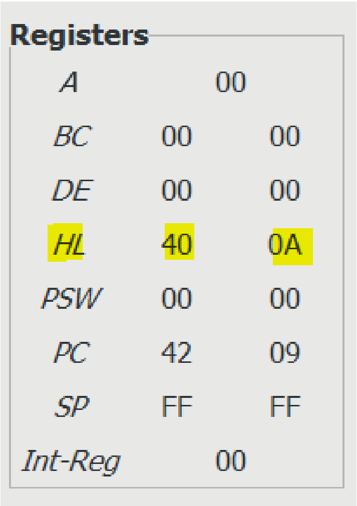
Notepad Code:



```
;(3) Write a Program to shift a 16-bit data 1 bit left.
; Assume data is in the HL register pair

; INPUT : 16 BIT DATA at Location 3001H & 3000H

LHLD 3000H

; Observation : To Shift 16-bit data by 1 Bit : We add [HL] to itself
; Since Left Shift Means Multiplying by 2
; [H-L] <- [H-L] + [H-L]
; DAD -> Add Register Pair to [H-L]
DAD H

; OUTPUT : 16 BIT DATA at Location 3006H & 3005H
SHLD 3005H

HLT
```

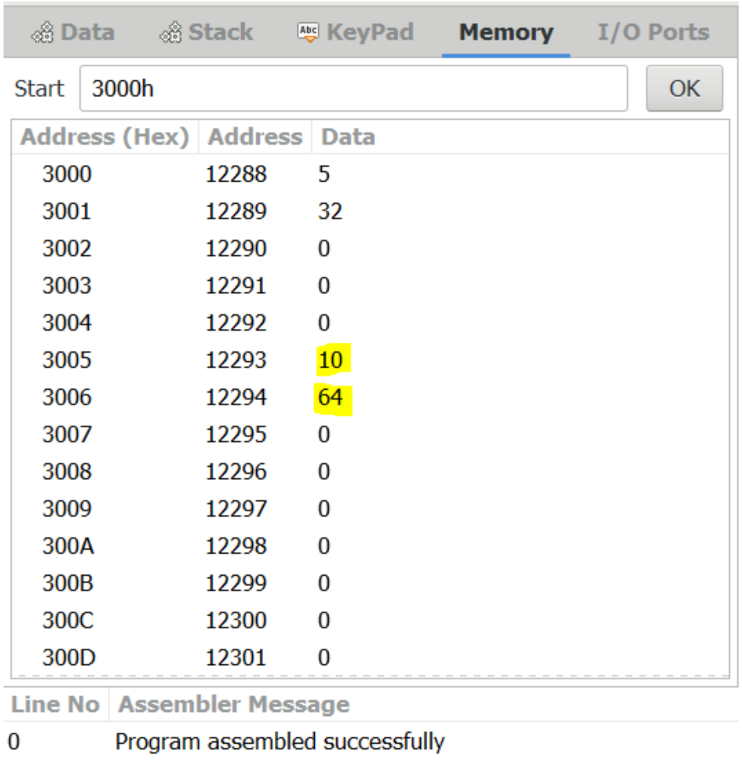
Registers and Memory:

INPUT	EXPECTED OUTPUT
(20 05)H	After Shifting Left, By 1 Bit
[3000H] = 05H = (5) [Decimal]	$(0100\ 0000\ 0000\ 1010)_2$ = (40 0A)H
[3001H] = 20H = (32) [Decimal]	[3005H] = 0AH = (10) [Decimal]
(2005)H = $(0010\ 0000\ 0000\ 0101)_2$	[3006H] = 40H = (64) [Decimal]



Registers

A	00	
BC	00	00
DE	00	00
HL	40	0A
PSW	00	00
PC	42	09
SP	FF	FF
Int-Reg	00	

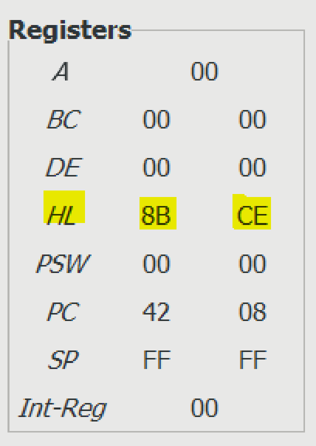


Start 3000h

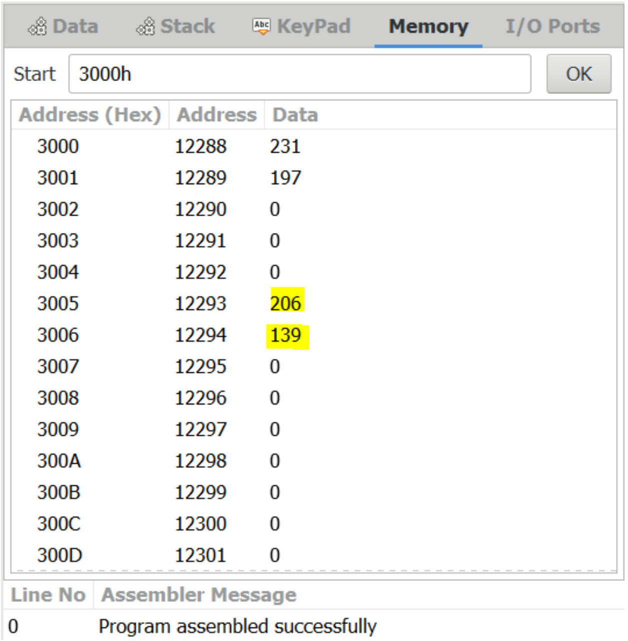
Address (Hex)	Address	Data
3000	12288	5
3001	12289	32
3002	12290	0
3003	12291	0
3004	12292	0
3005	12293	10
3006	12294	64
3007	12295	0
3008	12296	0
3009	12297	0
300A	12298	0
300B	12299	0
300C	12300	0
300D	12301	0

Line No	Assembler Message
0	Program assembled successfully

INPUT	EXPECTED OUTPUT
(C5 E7)H	After Shifting Left, By 1 Bit
[3000H] = E7H = (231) [Decimal]	$(1000\ 1011\ 1100\ 1110)_2$ = (8B CE)H
[3001H] = C5H = (197) [Decimal]	[3005H] = CEH = (206) [Decimal]
(2005)H = $(1100\ 0101\ 1110\ 0111)_2$	[3006H] = 8BH = (139) [Decimal]



```
Registers
 A          00
 BC    00   00
 DE    00   00
 HL    8B   CE
 PSW   00   00
 PC    42   08
 SP    FF   FF
 Int-Reg    00
```



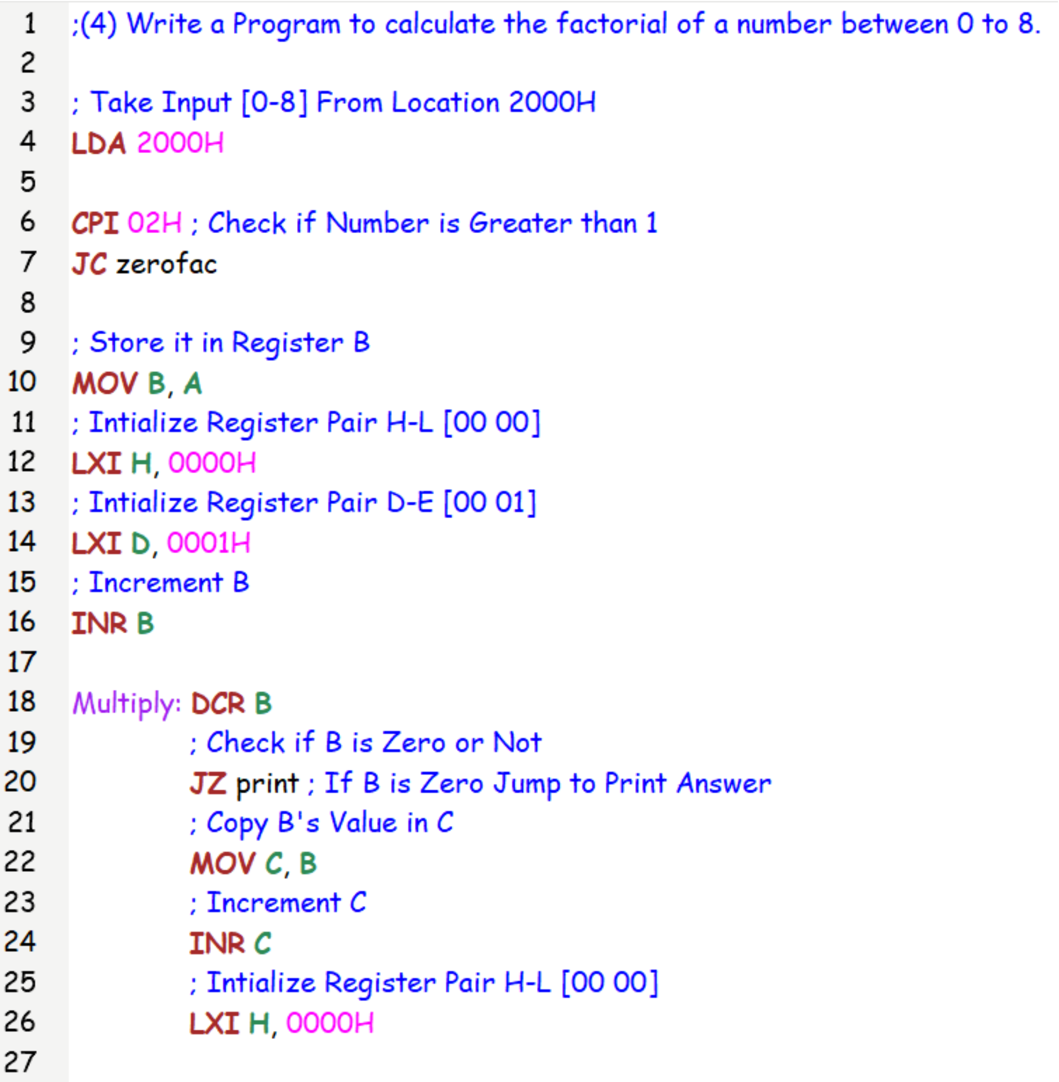
Start 3000h

Address (Hex)	Address	Data
3000	12288	231
3001	12289	197
3002	12290	0
3003	12291	0
3004	12292	0
3005	12293	206
3006	12294	139
3007	12295	0
3008	12296	0
3009	12297	0
300A	12298	0
300B	12299	0
300C	12300	0
300D	12301	0

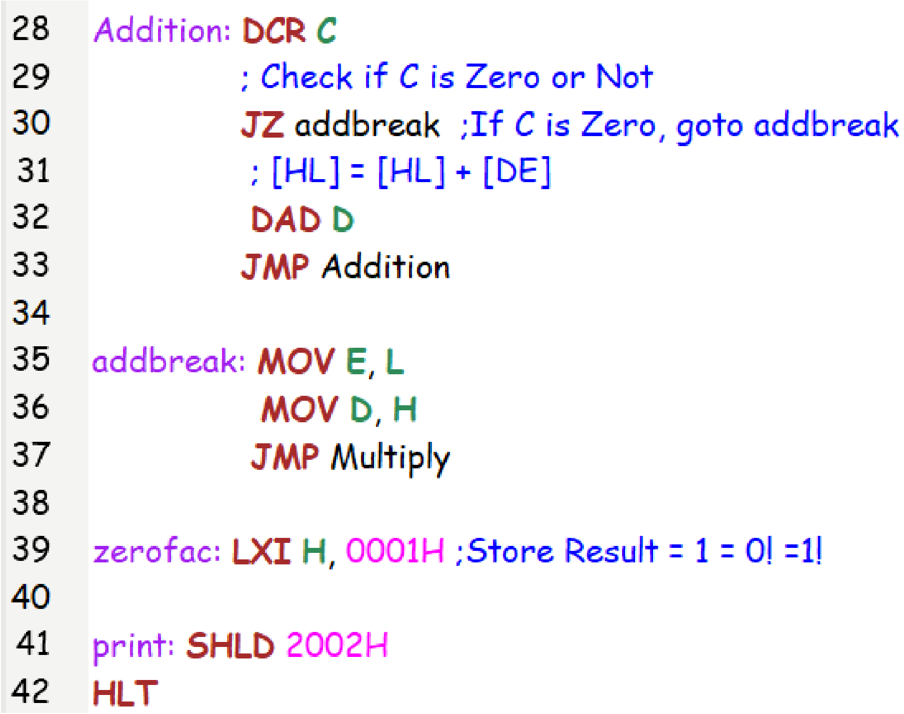
Line No	Assembler Message
0	Program assembled successfully

(4) Write a Program to calculate the factorial of a number between 0 to 8.

Notepad Code:



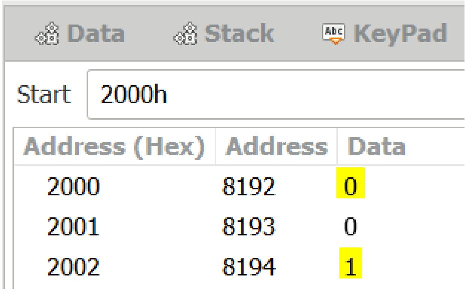
```
;(4) Write a Program to calculate the factorial of a number between 0 to 8.

; Take Input [0-8] From Location 2000H
LDA 2000H

CPI 02H ; Check if Number is Greater than 1
JC zerofac

; Store it in Register B
MOV B, A
; Intialize Register Pair H-L [00 00]
LXI H, 0000H
; Intialize Register Pair D-E [00 01]
LXI D, 0001H
; Increment B
INR B

Multiply: DCR B
          ; Check if B is Zero or Not
          JZ print ; If B is Zero Jump to Print Answer
          ; Copy B's Value in C
          MOV C, B
          ; Increment C
          INR C
          ; Intialize Register Pair H-L [00 00]
          LXI H, 0000H

```



```
Addition: DCR C
            ; Check if C is Zero or Not
            JZ addbreak  ;If C is Zero, goto addbreak
             ; [HL] = [HL] + [DE]
             DAD D
            JMP Addition

addbreak: MOV E, L
             MOV D, H
             JMP Multiply

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

print: SHLD 2002H
HLT
```

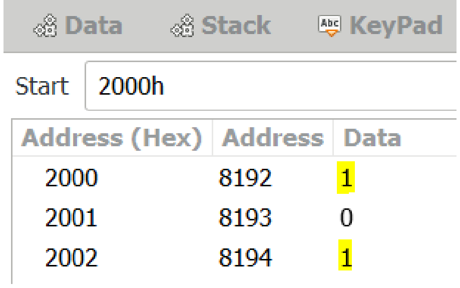
Registers and Memory:

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



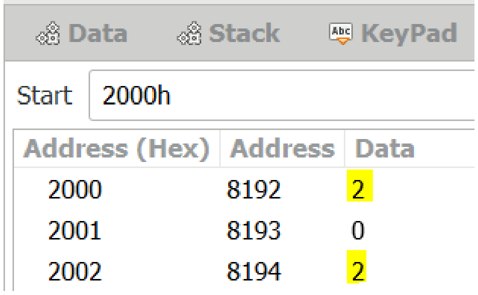
Start 2000h

Address (Hex)	Address	Data
2000	8192	0
2001	8193	0
2002	8194	1



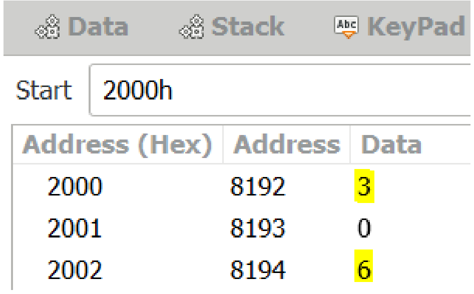
Start 2000h

Address (Hex)	Address	Data
2000	8192	1
2001	8193	0
2002	8194	1



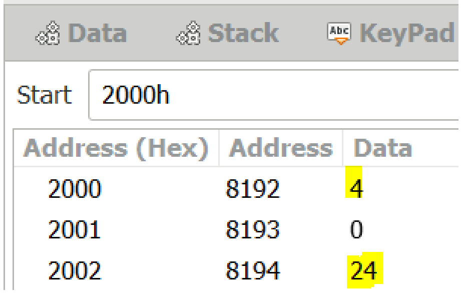
Start 2000h

Address (Hex)	Address	Data
2000	8192	2
2001	8193	0
2002	8194	2



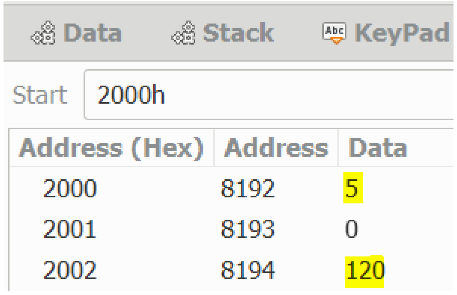
Start 2000h

Address (Hex)	Address	Data
2000	8192	3
2001	8193	0
2002	8194	6



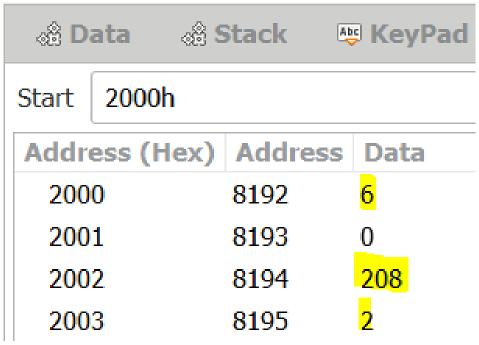
Start 2000h

Address (Hex)	Address	Data
2000	8192	4
2001	8193	0
2002	8194	24



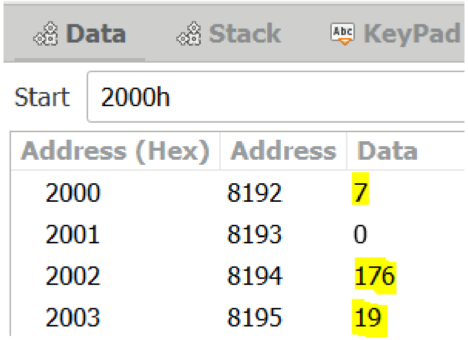
Start 2000h

Address (Hex)	Address	Data
2000	8192	5
2001	8193	0
2002	8194	120



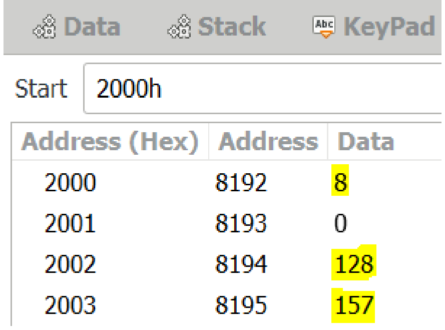
Start 2000h

Address (Hex)	Address	Data
2000	8192	6
2001	8193	0
2002	8194	208
2003	8195	2



Start 2000h

Address (Hex)	Address	Data
2000	8192	7
2001	8193	0
2002	8194	176
2003	8195	19



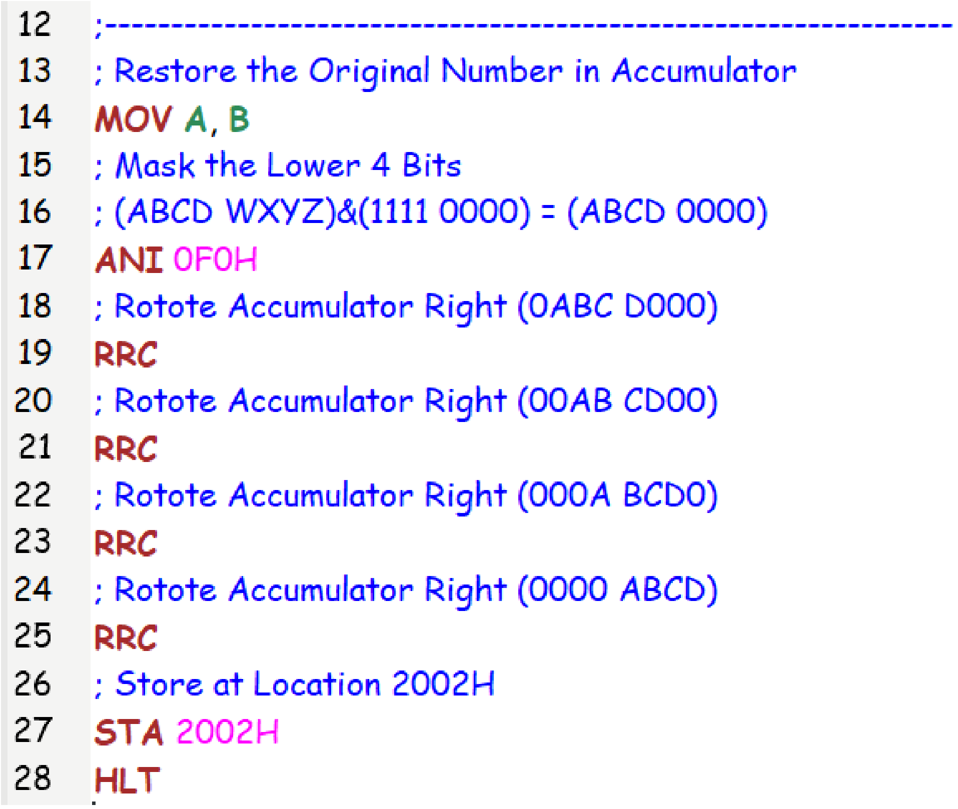
Start 2000h

Address (Hex)	Address	Data
2000	8192	8
2001	8193	0
2002	8194	128
2003	8195	157

(5) Write a program to Split 8 bit HEX data into two nibbles and store it in memory.

Notepad Code:

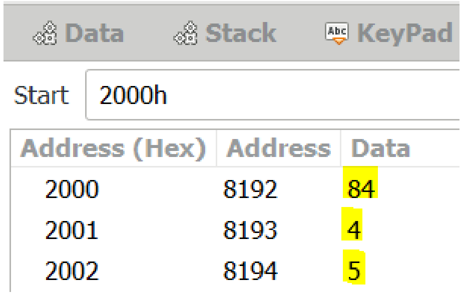
```
;(5) Write a program to Split 8 bit HEX data into two nibbles and store it in memory.

; Load A Number from Location 2000H [ABCD WXYZ]
LDA 2000H
; Store Original Number in Register B [ABCD WXYZ]
MOV B, A
; Mask the Upper 4 Bits
; (ABCD WXYZ)&(0000 1111) = (0000 WXYZ)
ANI 0FH
; Store at Location 2001H
STA 2001H
```



```
;-------------------------------------------------------------
; Restore the Original Number in Accumulator
MOV A, B
; Mask the Lower 4 Bits
; (ABCD WXYZ)&(1111 0000) = (ABCD 0000)
ANI 0F0H
; Rotote Accumulator Right (0ABC D000)
RRC
; Rotote Accumulator Right (00AB CD00)
RRC
; Rotote Accumulator Right (000A BCD0)
RRC
; Rotote Accumulator Right (0000 ABCD)
RRC
; Store at Location 2002H
STA 2002H
HLT
```

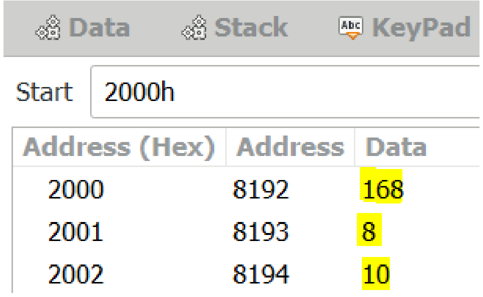
Registers and Memory:

INPUT [(2000)H]		OUTPUT1 [(2002)H]		OUTPUT2[(2001)H]	
Hex	Decimal	Hex	Decimal	Hex	Decimal
(54)H	84	(05)H	5	(04)H	4
(A8)H	168	(0A)H	10	(08)H	8
(DA)H	218	(0D)H	13	(0A)H	10



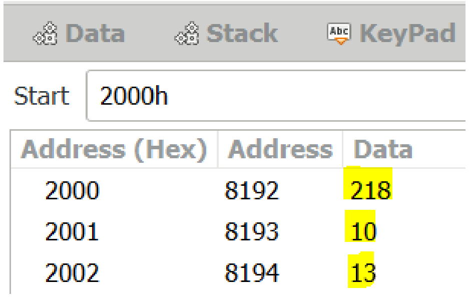
Start 2000h

Address (Hex)	Address	Data
2000	8192	84
2001	8193	4
2002	8194	5



Start 2000h

Address (Hex)	Address	Data
2000	8192	168
2001	8193	8
2002	8194	10



Start 2000h

Address (Hex)	Address	Data
2000	8192	218
2001	8193	10
2002	8194	13

SUBMITTED BY: BHAGYA VINOD RANA [*U19CS012*]