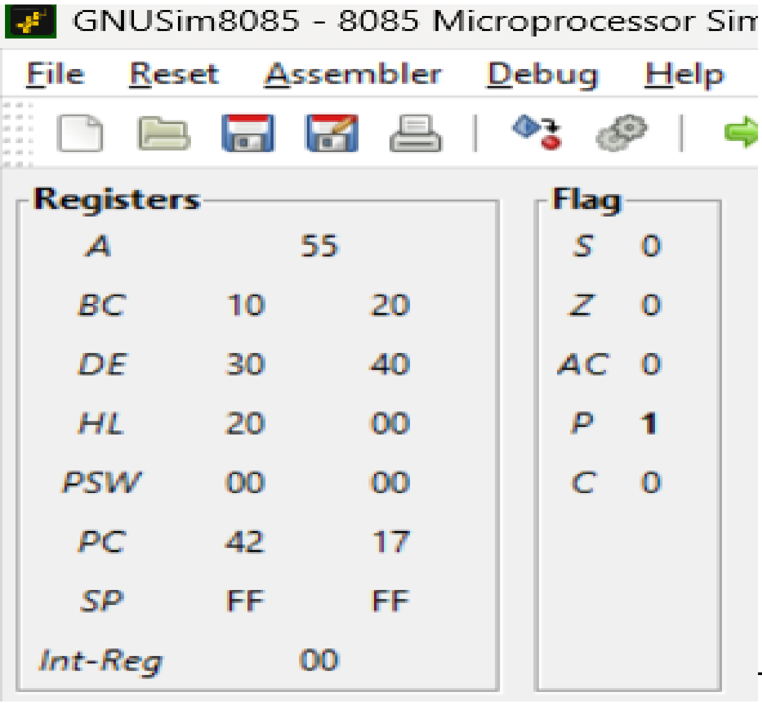
PRACTICAL-1

Aim: Write a program to execute all data transfer, arithmetic and logical, shift and all rotate instructions.

Program:

```
; Initial Instructions
MVI A, 50H      ; Load A with 50H (this will be overwritten)
MVI B, 10H      ; Load B with 10H
MVI C, 20H      ; Load C with 20H
MVI D, 30H      ; Load D with 30H
MVI E, 40H      ; Load E with 40H
; Data Transfer Instructions
MVI A, 55H      ; Overwrites A with 55H
LXI H, 2000H    ; Load HL pair with memory address 2000H
MOV M, A        ; Store the value of A (55H) into memory location 2000H
LDA 2000H       ; Load the value from memory location 2000H into A
STA 3000H       ; Store the value of A into memory location 3000H
HLT             ; Halt the execution
```

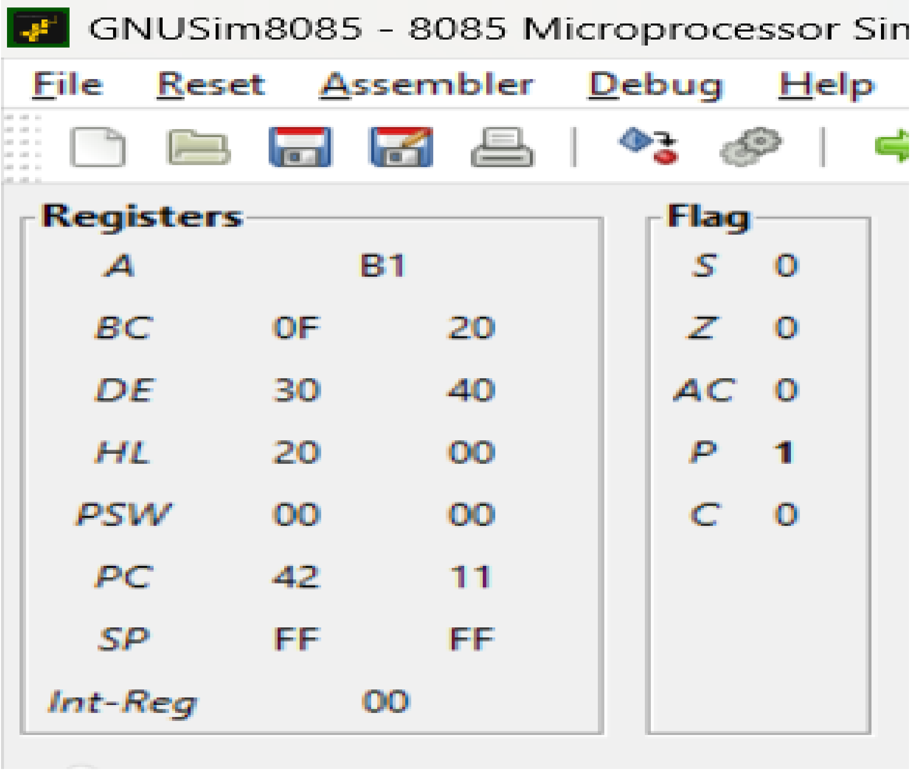
Output:



```
Registers                     Flag
A          55                 S   0
BC      10      20            Z   0
DE      30      40            AC  0
HL      20      00            P   1
PSW     00      00            C   0
PC      42      17
SP      FF      FF
Int-Reg    00
```

```
;Arithmetic Instructions
ADD B          ; A = A + B  ->  50H + 10H = 60H
SUB C          ; A = A - C  ->  60H - 20H = 40H
ADD D          ; A = A + D  ->  40H + 30H = 70H
ADD E          ; A = A + E  ->  70H + 40H = B0H
INR A          ; Increment A by 1  ->  B0H + 1 = B1H
DCR B          ; Decrement B by 1  ->  B - 1 = 0FH
HLT            ; Halt the execution
```

<u>Output:</u>

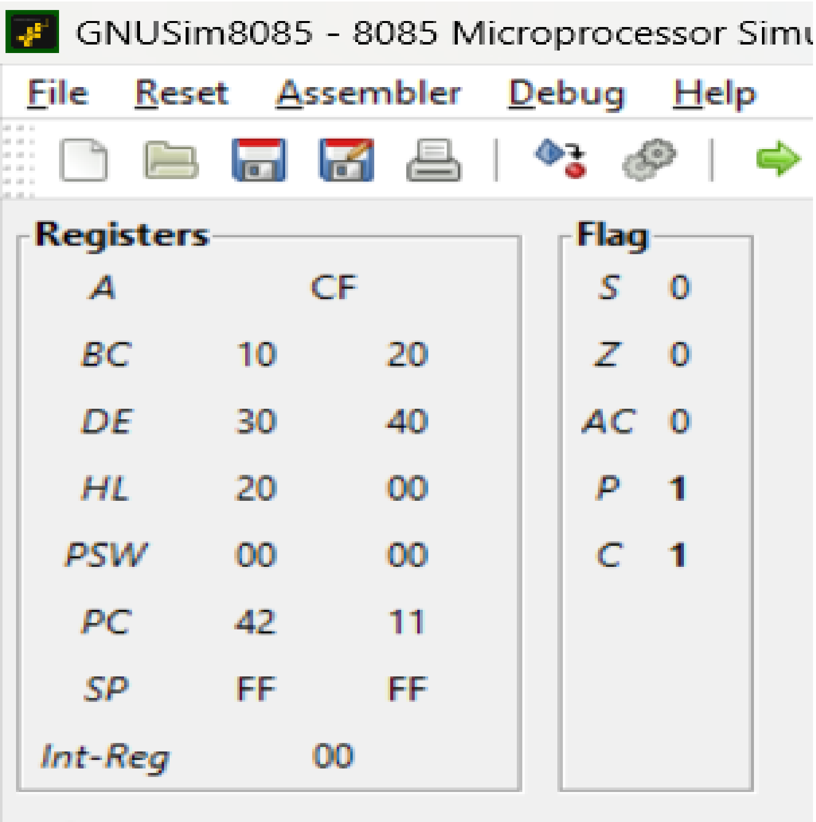


```
Registers                              Flag
  A              B1                      S    0
  BC        0F         20                Z    0
  DE        30         40                AC   0
  HL        20         00                P    1
  PSW       00         00                C    0
  PC        42         11
  SP        FF         FF
  Int-Reg        00
```

;Logical Instructions

```
ANA B          ; A = A & B  ->  50H & 10H = 10H  (Bitwise AND)
XRA C          ; A = A ^ C  ->  10H ^ 20H = 30H  (Bitwise XOR)
ORA D          ; A = A | D  ->  30H | 30H = 30H  (Bitwise OR)
CMA           ; A = ~A     ->  Complement A (NOT operation) -> CFH
CMC           ; Complement Carry Flag
STC          ; Set Carry Flag
HLT          ; Halt the execution
```

<u>Output:</u>



Registers

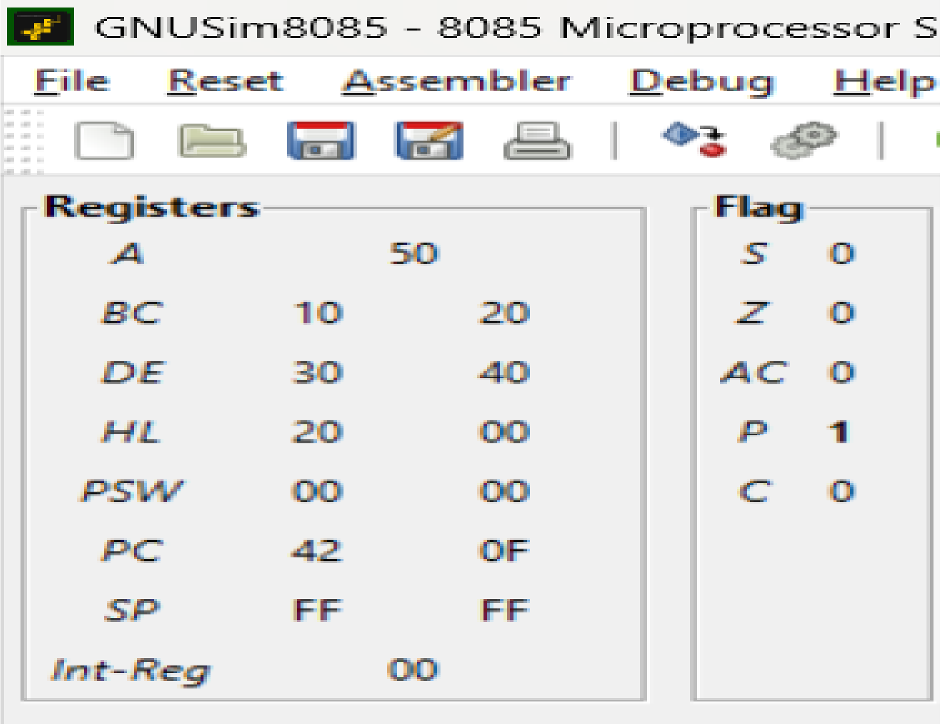
A	CF	
BC	10	20
DE	30	40
HL	20	00
PSW	00	00
PC	42	11
SP	FF	FF
Int-Reg	00	

Flag

S	0
Z	0
AC	0
P	1
C	1

```
; Shfit and Rotate Instructions
RLC             ; Rotate A left through carry (Circular Left Shift)
RRC             ; Rotate A right through carry (Circular Right Shift)
RAL             ; Rotate A left (Through Carry)
RAR             ; Rotate A right (Through Carry)
HLT             ; Halt the execution
```

Output:



Registers

A	50	
BC	10	20
DE	30	40
HL	20	00
PSW	00	00
PC	42	0F
SP	FF	FF
Int-Reg	00	

Flag

S	0
Z	0
AC	0
P	1
C	0

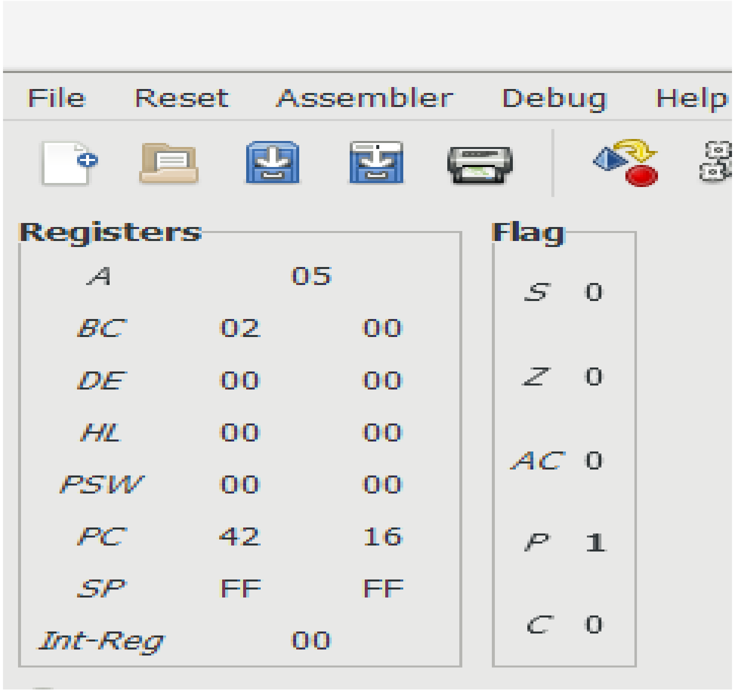
PRACTICAL-2

Aim: a) Write a program to add the content of location4000h and 4001h and store answer at 4002h

Program:

```
MVI A, 02H              ; Load immediate value 02H into accumulator A
STA 4000H               ; Store the value of A into memory location 4000H
MVI A, 03H               ; Load immediate value 03H into accumulator A
STA 4001H                ; Store the value of A into memory location 4001H
LDA 4000H                ; Load the value from memory location 4000H into A
MOV B, A                 ; Move the value of A into register B
LDA 4001H                ; Load the value from memory location 4001H into A
ADD B                    ; Add the value of B (4000H) to A (4001H)
STA 4002H               ; Store the result in memory location 4002H
HLT                     ; Halt the program
```

Output:



Registers

A	05	
BC	02	00
DE	00	00
HL	00	00
PSW	00	00
PC	42	16
SP	FF	FF
Int-Reg	00	

Flag

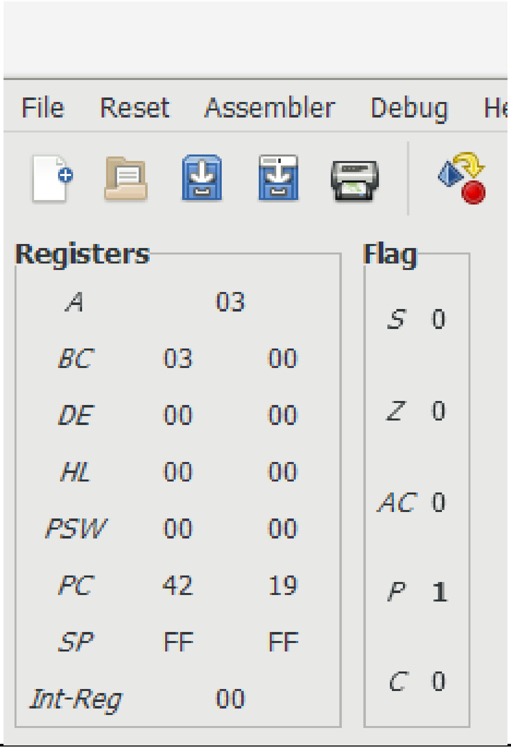
S	0
Z	0
AC	0
P	1
C	0

Aim: b) Write an 8085 assembly language program for exchanging two 8- bit numbers stored in memory locations 2050h and 2051h.

Program:

```
MVI A, 03H          ; Load immediate value 03H into accumulator A
STA 2050H           ; Store 03H into memory location 2050H
MVI A, 04H          ; Load immediate value 04H into accumulator A
STA 2051H           ; Store 04H into memory location 2051H
LDA 2050H           ; Load the value from memory location 2050H into A
MOV B, A            ; Copy the value of A (2050H) into register B
LDA 2051H           ; Load the value from memory location 2051H into A
STA 2050H           ; Store the value of A (2051H) into 2050H
MOV A, B             ; Move the old value of 2050H (stored in B) back to A
STA 2051H           ; Store the swapped value into 2051H
HLT                 ; Halt the program
```

Output:

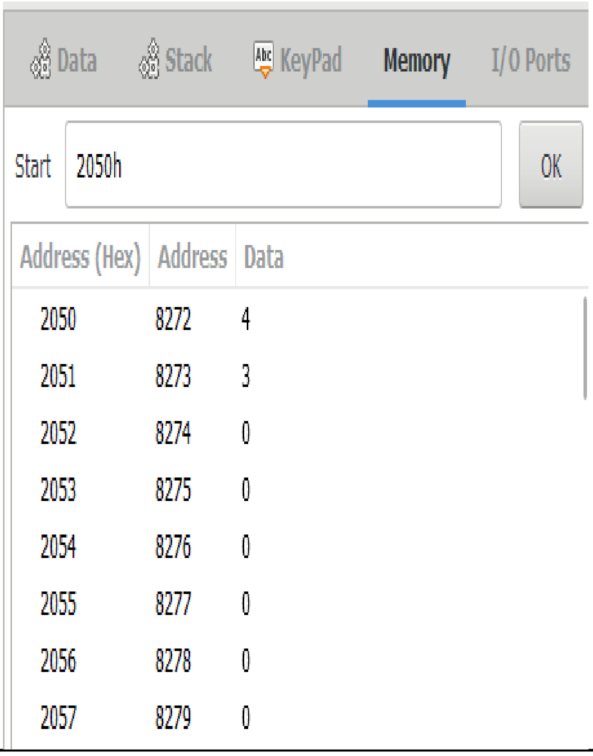


Registers

A	03	
BC	03	00
DE	00	00
HL	00	00
PSW	00	00
PC	42	19
SP	FF	FF
Int-Reg	00	

Flag

S	0
Z	0
AC	0
P	1
C	0



Start 2050h

Address (Hex)	Address	Data
2050	8272	4
2051	8273	3
2052	8274	0
2053	8275	0
2054	8276	0
2055	8277	0
2056	8278	0
2057	8279	0

PRACTICAL-3

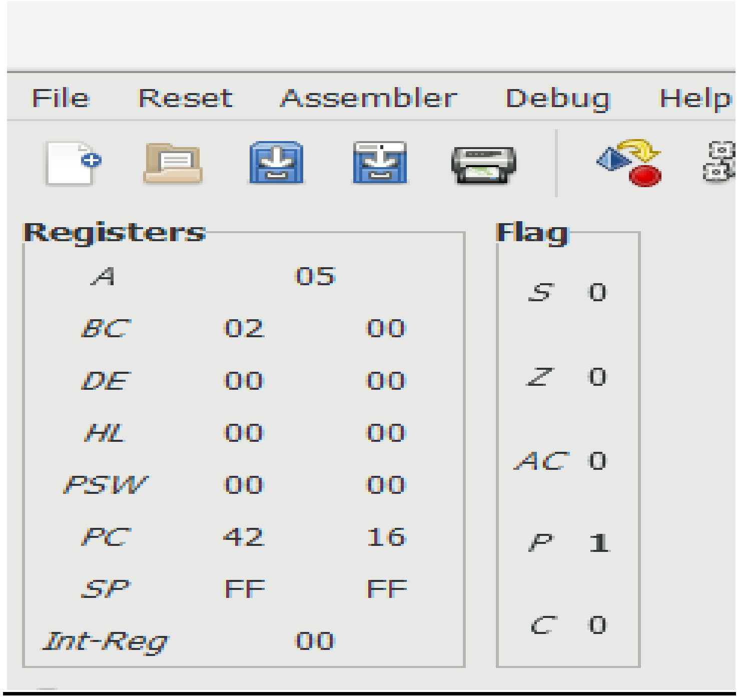
Aim: a) Write the program to add, subtract two 8 bit and16 bit numbers without using 16 bit instructions.

8-bit addition

Program:

```
MVI A, 02H      ; Load immediate value 02H into accumulator A
STA 4000H       ; Store the value of A into memory location 4000H
MVI A, 03H      ; Load immediate value 03H into accumulator A
STA 4001H       ; Store the value of A into memory location 4001H
LDA 4000H       ; Load the value from memory location 4000H into A
MOV B, A        ; Move the value of A into register B
LDA 4001H       ; Load the value from memory location 4001H into A
ADD B           ; Add the value of B (4000H) to A (4001H)
STA 4002H       ; Store the result in memory location 4002H
HLT             ; Halt the program
```

Output:



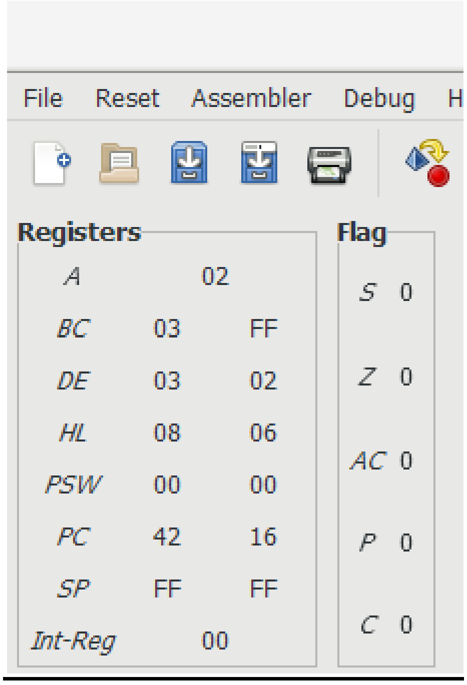
```
Registers                          Flag
   A            05                   S   0
   BC     02          00             Z   0
   DE     00          00             AC  0
   HL     00          00             P   1
   PSW    00          00             C   0
   PC     42          16
   SP     FF          FF
   Int-Reg      00
```

8-bit subtraction

Program:

```
MVI A,03h      ; Load immediate value 03h into accumulator
STA 4000h      ; Store the value (03h) at memory location 4000h
MVI A,05h      ; Load immediate value 05h into accumulator
STA 4001h      ; Store the value (05h) at memory location 4001h
LDA 4000h      ; Load the value from memory location 4000h (03h) into accumulator
MOV B,A        ; Copy accumulator (03h) into register B
LDA 4001h      ; Load the value from memory location 4001h (05h) into accumulator
SUB B        ; Subtract register B (03h) from accumulator (05h); result is 02h
STA 4002h      ; Store the result (02h) at memory location 4002h
HLT          ; Halt the execution of the program
```

Output:



Registers

A	02	
BC	03	FF
DE	03	02
HL	08	06
PSW	00	00
PC	42	16
SP	FF	FF
Int-Reg	00	

Flag

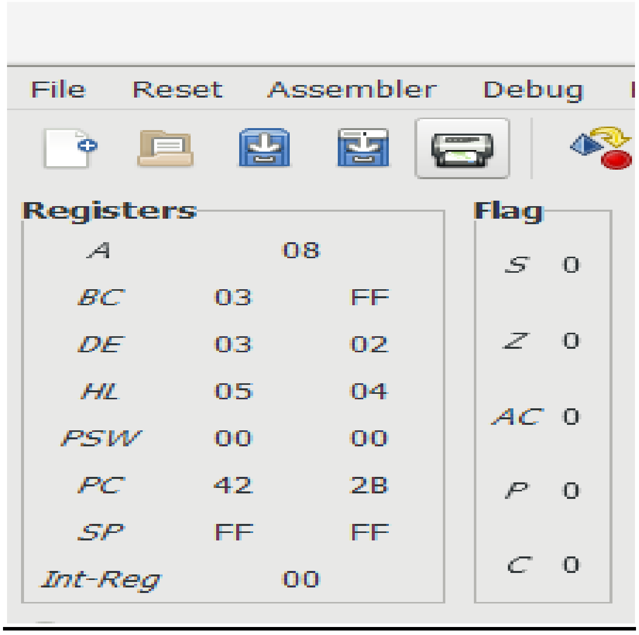
S	0
Z	0
AC	0
P	0
C	0

16 bit-addition

Program:

```
MVI A,02h     ; Load immediate value 02h into accumulator
STA 2000h     ; Store 02h at memory location 2000h
MVI A,03h     ; Load immediate value 03h into accumulator
STA 2001h     ; Store 03h at memory location 2001h
MVI A,04h     ; Load immediate value 04h into accumulator
STA 2002h     ; Store 04h at memory location 2002h
MVI A,05h     ; Load immediate value 05h into accumulator
STA 2003h     ; Store 05h at memory location 2003h
MVI A,06h     ; Load immediate value 06h into accumulator
STA 2004h     ; Store 06h at memory location 2004h
LHLD 2000h     ; Load 16-bit number from memory locations 2000h (L) and 2001h (H) into HL pair
XCHG         ; Exchange HL with DE; now DE holds first 16-bit number
LHLD 2002h     ; Load 16-bit number from memory locations 2002h (L) and 2003h (H) into HL pair
MOV A,H       ; Move the high byte of second number (H) into accumulator
ADD H        ; Add the high byte (H) of first number (which is now in H via XCHG) to A
STA 2005h     ; Store the resulting high byte sum at memory location 2005h
MOV A,L       ; Move the low byte of second number (L) into accumulator
ADC L        ; Add the low byte (L) of first number with carry to A
STA 2006h     ; Store the resulting low byte sum at memory location 2006h
HLT         ; Halt program execution
```

Output:

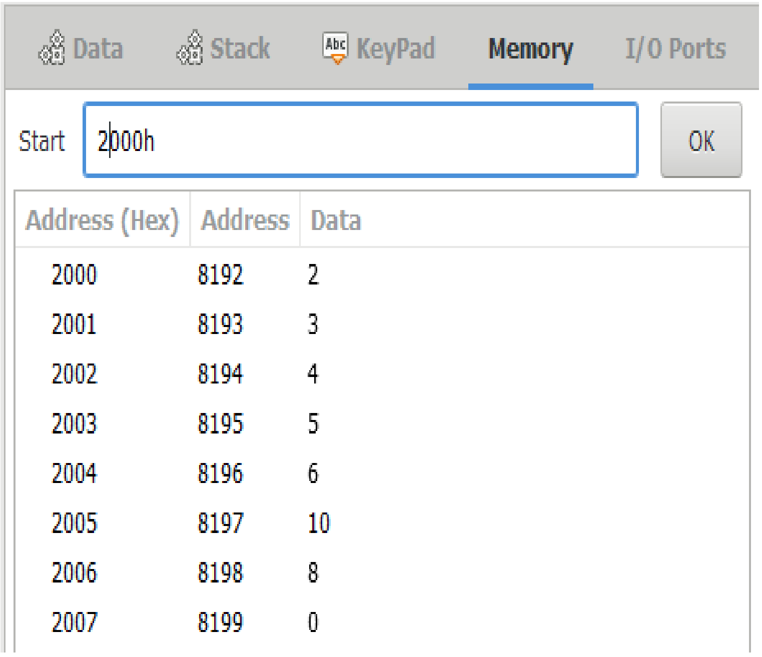


Registers

A	08	
BC	03	FF
DE	03	02
HL	05	04
PSW	00	00
PC	42	2B
SP	FF	FF
Int-Reg	00	

Flag

S	0
Z	0
AC	0
P	0
C	0



Start 2000h

Address (Hex)	Address	Data
2000	8192	2
2001	8193	3
2002	8194	4
2003	8195	5
2004	8196	6
2005	8197	10
2006	8198	8
2007	8199	0

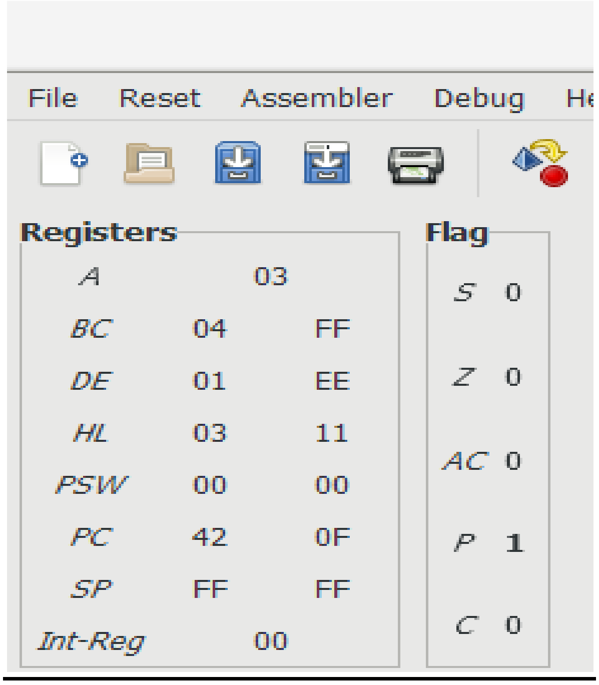
p

16-bit subtraction:

Program:

```
MVI B,04h       ; Load immediate value 04h into register B
MVI C,0FFh      ; Load immediate value 0FFh into register C
MVI D,01h       ; Load immediate value 01h into register D
MVI E,0EEh      ; Load immediate value 0EEh into register E
MOV A,C         ; Copy the value from C (0FFh) into accumulator A
SUB E           ; Subtract the value in E (0EEh) from A: 0FFh - 0EEh = 11h
MOV L,A         ; Move the result (11h) into register L (low byte)
MOV A,B         ; Copy the value from B (04h) into accumulator A
SUB D           ; Subtract the value in D (01h) from A: 04h - 01h = 03h
MOV H,A         ; Move the result (03h) into register H (high byte)
HLT             ; Halt program execution
```

Output:



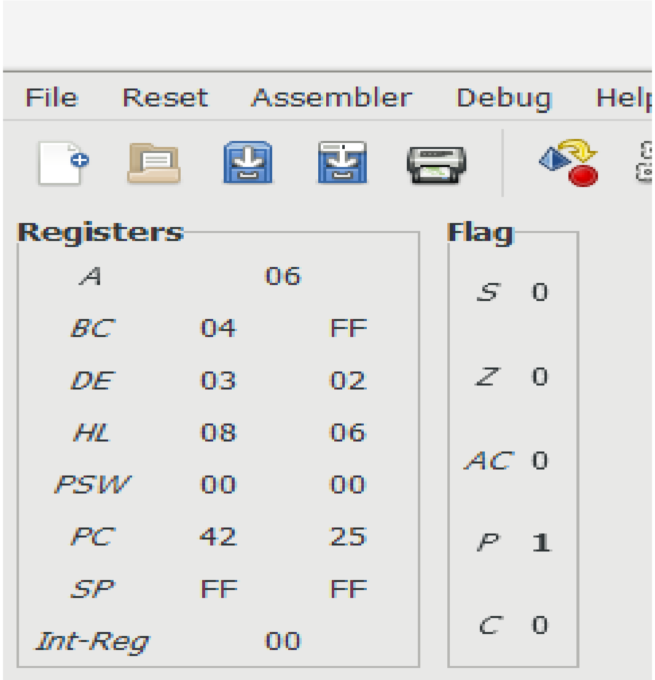
```
Registers                      Flag
A              03              S   0
BC       04         FF
DE       01         EE         Z   0
HL       03         11
PSW      00         00         AC  0
PC       42         0F
SP       FF         FF         P   1
Int-Reg        00              C   0
```

b) Write an 8085 assembly language program to add two16-bit number stored in memory.

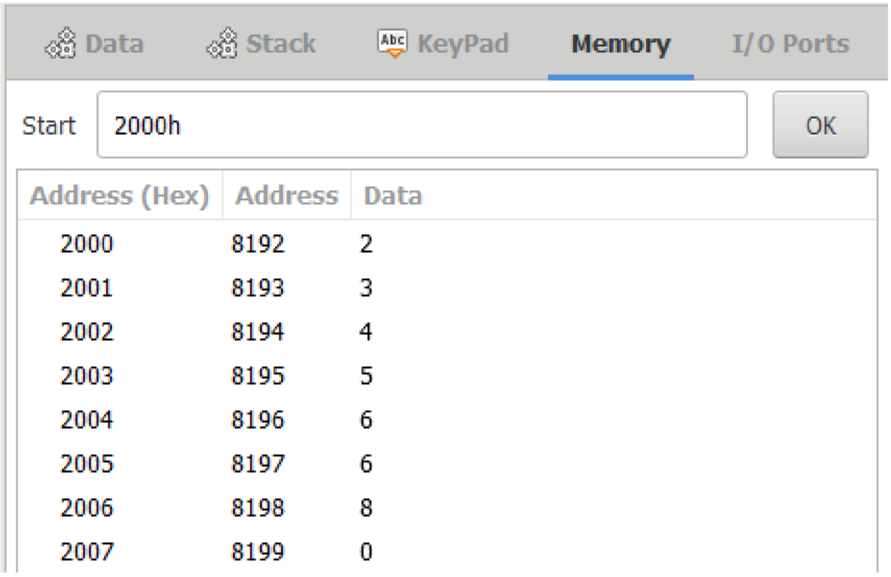
<u>Program:</u>

```
MVI A,02h      ; Load immediate value 02h into accumulator
STA 2000h      ; Store 02h at memory location 2000h (lower byte of first 16-bit number)
MVI A,03h      ; Load immediate value 03h into accumulator
STA 2001h      ; Store 03h at memory location 2001h (higher byte of first 16-bit number)
MVI A,04h      ; Load immediate value 04h into accumulator
STA 2002h      ; Store 04h at memory location 2002h (lower byte of second 16-bit number)
MVI A,05h      ; Load immediate value 05h into accumulator
STA 2003h      ; Store 05h at memory location 2003h (higher byte of second 16-bit number)
MVI A,06h      ; Load immediate value 06h into accumulator
STA 2004h      ; Store 06h at memory location 2004h (extra data, if needed)
LHLD 2000h      ; Load 16-bit value from 2000h (L) and 2001h (H) into HL pair
XCHG           ; Exchange HL with DE so that DE now holds the first 16-bit number
LHLD 2002h      ; Load 16-bit value from 2002h (L) and 2003h (H) into HL pair
DAD D          ; Add the 16-bit number in DE (first number) to HL (second number)
SHLD 2005h      ; Store the resulting 16-bit sum into memory locations 2005h (L) and 2006h (H)
HLT           ; Halt execution
```

Output:



```
Registers                          Flag
   A              06                 S   0
   BC       04          FF
   DE       03          02           Z   0
   HL       08          06
   PSW      00          00           AC  0
   PC       42          25
   SP       FF          FF           P   1
   Int-Reg        00                 C   0
```



Start 2000h

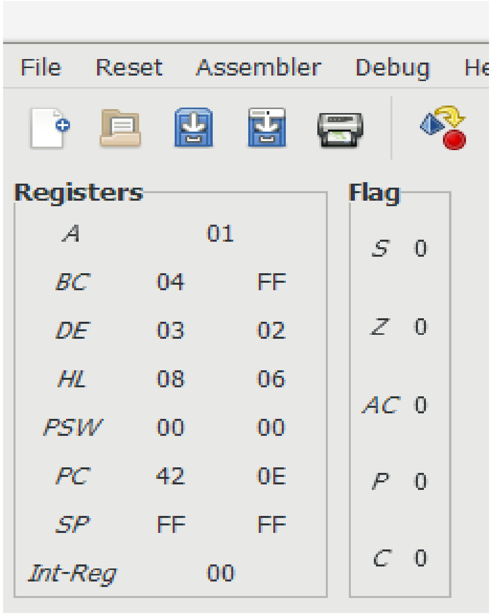
Address (Hex)	Address	Data
2000	8192	2
2001	8193	3
2002	8194	4
2003	8195	5
2004	8196	6
2005	8197	6
2006	8198	8
2007	8199	0

c) Write a program to find 2’s compliment of given number.

Program:

```
MVI A,0255    ; Load immediate value 0255h into accumulator
STA 5000h     ; Store 0255h at memory location 5000h
CMA           ; Complement the accumulator (bitwise NOT of A)
STA 5001h     ; Store the complemented value at memory location 5001h
INR A         ; Increment the accumulator by 1
STA 5002h     ; Store the incremented value at memory location 5002h
HLT           ; Halt program execution
```

Output:

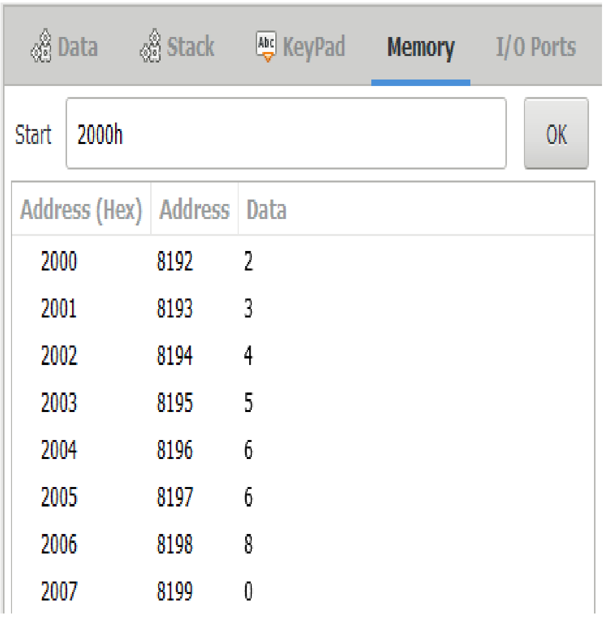


Registers

A	01	
BC	04	FF
DE	03	02
HL	08	06
PSW	00	00
PC	42	0E
SP	FF	FF
Int-Reg	00	

Flag

S	0
Z	0
AC	0
P	0
C	0



Start 2000h

Address (Hex)	Address	Data
2000	8192	2
2001	8193	3
2002	8194	4
2003	8195	5
2004	8196	6
2005	8197	6
2006	8198	8
2007	8199	0