# INSTITUTE OF COMPUTER TECHNOLOGY

# B-TECH COMPUTER SCIENCE ENGINEERING 2025-26

# SUBJECT: MICROCONTROLLER & APPLICATION

NAME: Rahul Prajapati

ENRLL. NO: 23162171020

BRANCH: CYBER SECURITY

BATCH: 52

## PRACTICAL_3

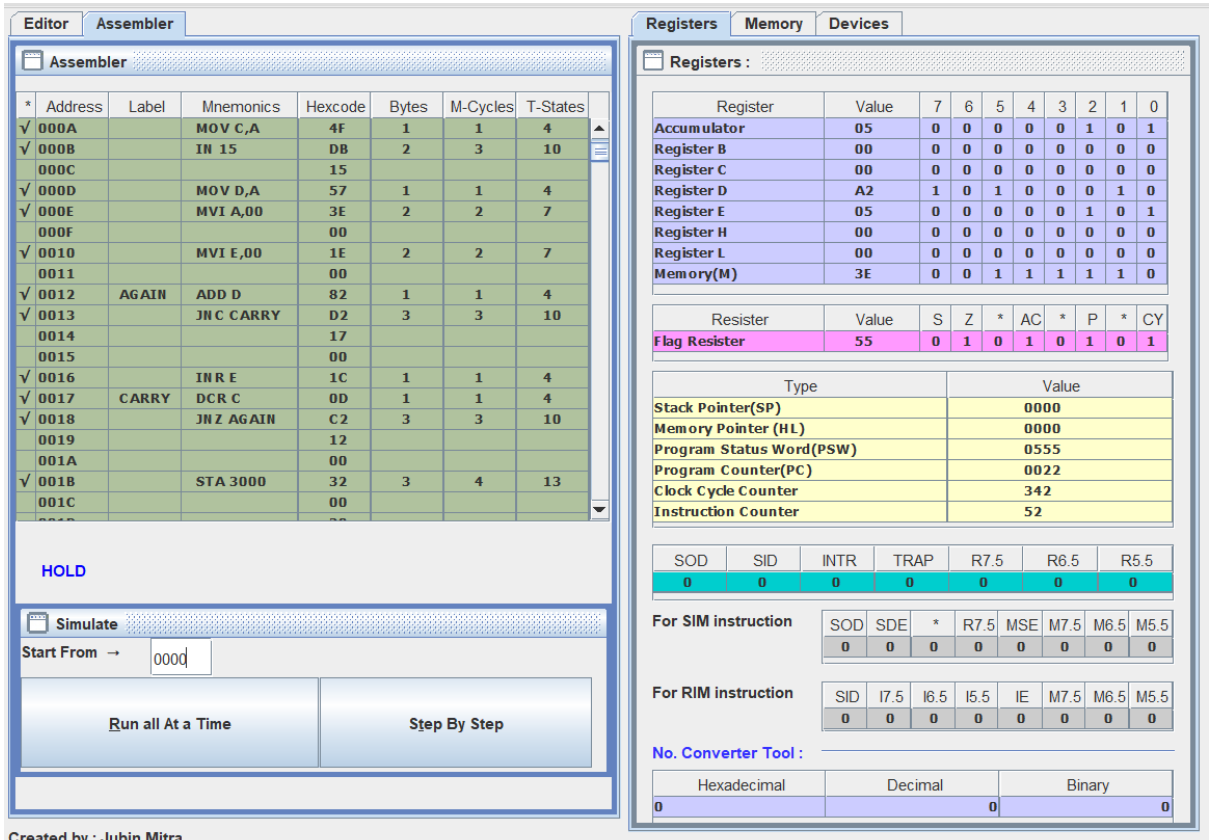
**Aim:-** Learning Programs using Branch Instructions like JMP, JZ, JNZ, JC etc.

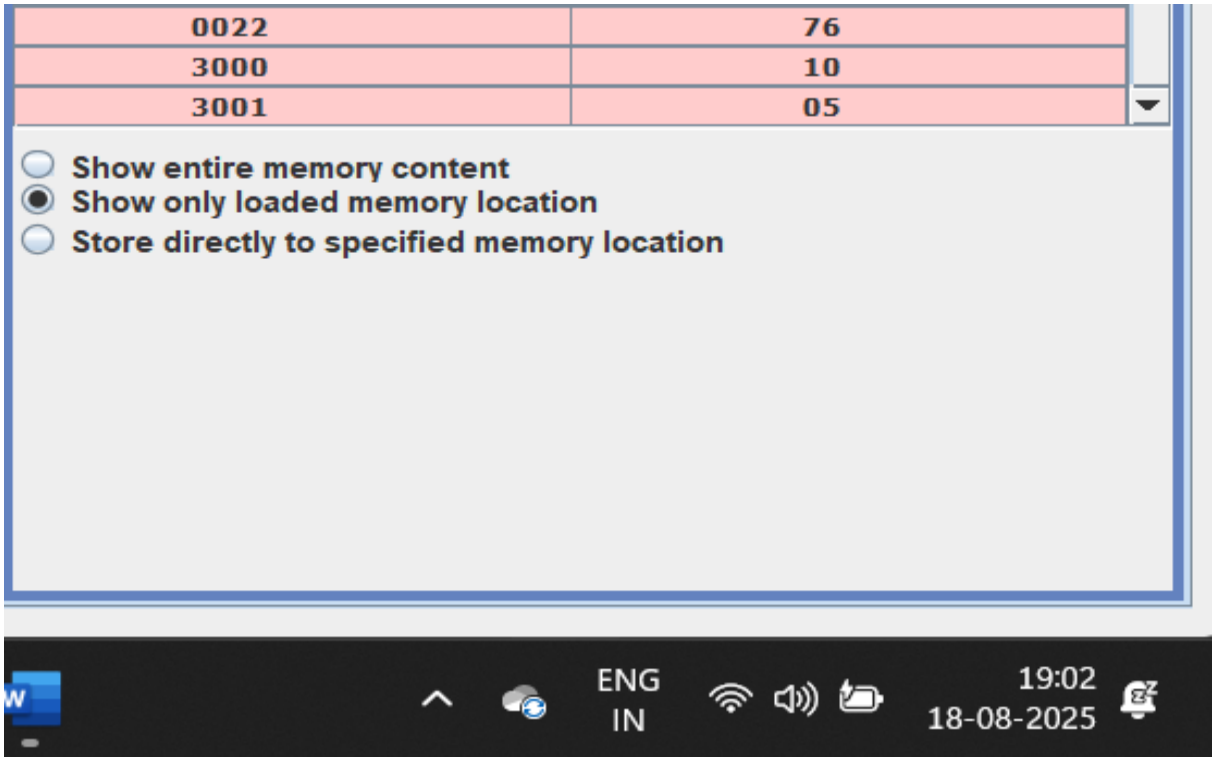
1. Write a Program to multiply two 8-bit numbers given as input on input ports 15H and 16H. Save the lower by of the result on memory location 3000H and higher byte of the result on 3001H.

```
            MVI A,A2H // MOVE A2H IN ACCUMULATOR
            OUT 15H   // STORE ACCUMULATOR DATA IN MEMORY
ADDRESS 15H
            MVI A,08H // MOVE 08H IN ACCUMULATOR
            OUT 16    // STORE ACCUMULATOR DATA IN MEMORY
ADDRESS 15H
            IN 16H    // FETCH DATA FROM ADDRESS 16H
            MOV C,A   // MOVE ACCUMULATOR DATA INTO REGISTER
C FOR COUNTER
            IN 15H    // FETCH DATA FROM ADDRESS 15H
            MOV D,A   // MOVE ACCUMULATOR DATA INTO REGISTER
D & ITS ACT AS A MULTIPLICAND
            MVI A,00H // ACCUMULATOR HOLD CURRENT SUM
            MVI E,00H // ITS HOLD CARRY SUM

AGAIN:      ADD D     // AGAIN IS LABEL WHICH IS ACT AS A LOOP
AND ADD D ISTRUCTION IN ADD DATA D INTO REGISTER A
            JNC CARRY             // IT WILL JUMP IF THEIR WILL
BE NO CARRY
            INR E     // INCREMENT HIGHER BYTE

CARRY:      DCR C     // DECREMENT COUNTER
            JNZ AGAIN             // IT WILL JUMP ON AGAIN WHEN
ZERO FLAG IS RESET
            STA 3000H // STORE LOWERBYTE RESULT AT 3000H
            MOV A,E   // MOVE HIGHER BYTE IN ACCUMULATOR
            STA 3001H // STORE HIGHER BYTE AT 3001H
            HLT       // HOLD
```


Editor
Assembler
Assembler
* Address Label Mnemonics Hexcode Bytes M-Cycles T-States
√ 000A MOV C,A 4F 1 1 4
√ 000B IN 15 DB 2 3 10
000C 15
√ 000D MOV D,A 57 1 1 4
√ 000E MVI A,00 3E 2 2 7
000F 00
√ 0010 MVI E,00 1E 2 2 7
0011 00
√ 0012 AGAIN ADD D 82 1 1 4
√ 0013 JNC CARRY D2 3 3 10
0014 17
0015 00
√ 0016 INR E 1C 1 1 4
√ 0017 CARRY DCR C 0D 1 1 4
√ 0018 JNZ AGAIN C2 3 3 10
0019 12
001A 00
√ 001B STA 3000 32 3 4 13
001C 00
HOLD
Simulate
Start From → 0000
Run all At a Time
Step By Step
Registers
Memory
Devices
Registers :
Register Value 7 6 5 4 3 2 1 0
Accumulator 05 0 0 0 0 0 1 0 1
Register B 00 0 0 0 0 0 0 0 0
Register C 00 0 0 0 0 0 0 0 0
Register D A2 1 0 1 0 0 0 1 0
Register E 05 0 0 0 0 0 1 0 1
Register H 00 0 0 0 0 0 0 0 0
Register L 00 0 0 0 0 0 0 0 0
Memory(M) 3E 0 0 1 1 1 1 1 0
Resister Value S Z * AC * P * CY
Flag Resister 55 0 1 0 1 0 1 0 1
Type Value
Stack Pointer(SP) 0000
Memory Pointer (HL) 0000
Program Status Word(PSW) 0555
Program Counter(PC) 0022
Clock Cycle Counter 342
Instruction Counter 52
SOD SID INTR TRAP R7.5 R6.5 R5.5
0 0 0 0 0 0 0
For SIM instruction
SOD SDE * R7.5 MSE M7.5 M6.5 M5.5
0 0 0 0 0 0 0 0
For RIM instruction
SID I7.5 I6.5 I5.5 IE M7.5 M6.5 M5.5
0 0 0 0 0 0 0 0
No. Converter Tool :
Hexadecimal Decimal Binary
0 0 0
Created by : Jubin Mitra
ENG
IN
19:00
18-08-2025


0022 76
3000 10
3001 05
Show entire memory content
Show only loaded memory location
Store directly to specified memory location
ENG
IN
19:02
18-08-2025

2. Write a Program to count positive and negative numbers out of 20 numbers stored In memory. (Assume appropriate memory location in your program and load 20 different bytes in memory using assembler directives). Display the count of positive nos on outpu port 30 H and negative numbers on 40 H.

Assembler | Disassembler

```
# ORG 5000H
# DB
F1H,52H,28H,B1H,A9H,08H,13H,36H,26H,88H,FFH,E4H,22H,31H,65H,44
# ORG 5100H
            LXI H,5000H // LOAD THE ADDRESS 5000H INTO HL PAIR
            MVI C,14H // STORE 14H IN REGISTER C
            MVI D,00H // STORE 00H IN REGISTER D FOR
                      //POSITIVE COUNT
            MVI E,00H // STORE 14H IN REGISTER E FOR
                      //NEGATIVE COUNT
NEXT:       MOV A,M   // MOVE MEMORY DATA INTO ACCUMULATOR
            ANI 80H   // PERFORM AND OPERATION TO
                      //IDENTIFY MSB 1 OR 0
            JZ POSITIVE // IF SIGN BIT IS 0 THEN JUMP ON
                        // POSITIVE
NEG:        INR E     // INCREMENT NEGATIVE COUNTER
            JMP CONTINUE        // JUMP ON CONTINUE
POSITIVE:   INR D     // INCREMENT POSITIVE COUNTER
CONTINUE:   INX H     // INCREMENT HL PAIR
            DCR C     // DECREMENT MAIN COUNTER
            JNZ NEXT  // JUMP ON NEXT IF ZERO FLAG IS NOT
                      // SET TO 0
            MOV A,D   // MOVE POSITIVE COUNTS TO
                      //ACCUMULATOR
            OUT 30H   // STORE ACCUMULATOR DATA INTO
                      // MEMORY ADDRESS 30H
            MOV A,E // MOVE NEGATIVE COUNTS TO ACCUMULATOR
            OUT 40H   // STORE ACCUMULATOR DATA INTO
                      // MEMORY ADDRESS 40H
```

Editor | Assembler

Assembler

| * | Address | Label | Mnemonics | Hexcode | Bytes | M-Cycles | T-States |
|---|---|---|---|---|---|---|---|
| | 5115 | | | 14 | | | |
| √ | 5116 | | MVI D,00 | 16 | 2 | 2 | 7 |
| | 5117 | | | 00 | | | |
| √ | 5118 | | MVI E,00 | 1E | 2 | 2 | 7 |
| | 5119 | | | 00 | | | |
| √ | 511A | NEXT | MOV A,M | 7E | 1 | 2 | 7 |
| √ | 511B | | ANI 80 | E6 | 2 | 2 | 7 |
| | 511C | | | 80 | | | |
| √ | 511D | | JZ POSITIVE | CA | 3 | 3 | 10 |
| | 511E | | | 24 | | | |
| | 511F | | | 51 | | | |
| √ | 5120 | NEG | INR E | 1C | 1 | 1 | 4 |
| √ | 5121 | | JMP CONTI... | C3 | 3 | 3 | 10 |
| | 5122 | | | 25 | | | |
| | 5123 | | | 51 | | | |
| √ | 5124 | POSIT... | INR D | 14 | 1 | 1 | 4 |
| √ | 5125 | CONT... | INX H | 23 | 1 | 1 | 6 |
| √ | 5126 | | DCR C | 0D | 1 | 1 | 4 |
| √ | 5127 | | JNZ NEXT | C2 | 3 | 3 | 10 |

IF SIGN BIT IS 0 THEN JUMP ON

Simulate

Start From → 5111

Run all At a Time | Step By Step

Registers | Memory | Devices

Registers :

| Register | Value | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| Accumulator | 06 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| Register B | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Register C | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Register D | 0E | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 |
| Register E | 06 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| Register H | 50 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| Register L | 14 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 |
| Memory(M) | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Resister | Value | S | Z | * | AC | * | P | * | CY |
|---|---|---|---|---|---|---|---|---|---|
| Flag Resister | 54 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 |

| Type | Value |
|---|---|
| Stack Pointer(SP) | 0000 |
| Memory Pointer (HL) | 5014 |
| Program Status Word(PSW) | 0654 |
| Program Counter(PC) | 5130 |
| Clock Cycle Counter | 1568 |
| Instruction Counter | 228 |

| SOD | SID | INTR | TRAP | R7.5 | R6.5 | R5.5 |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |

For SIM instruction

| SOD | SDE | * | R7.5 | MSE | M7.5 | M6.5 | M5.5 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

For RIM instruction

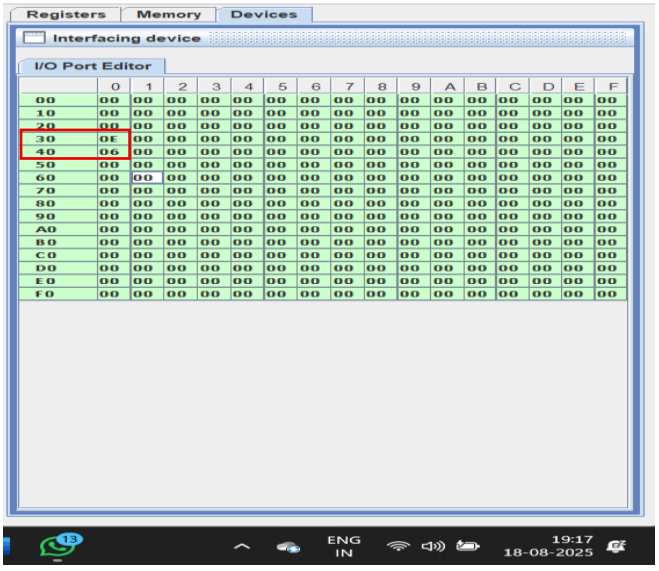
| SID | I7.5 | I6.5 | I5.5 | IE | M7.5 | M6.5 | M5.5 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

No. Converter Tool :

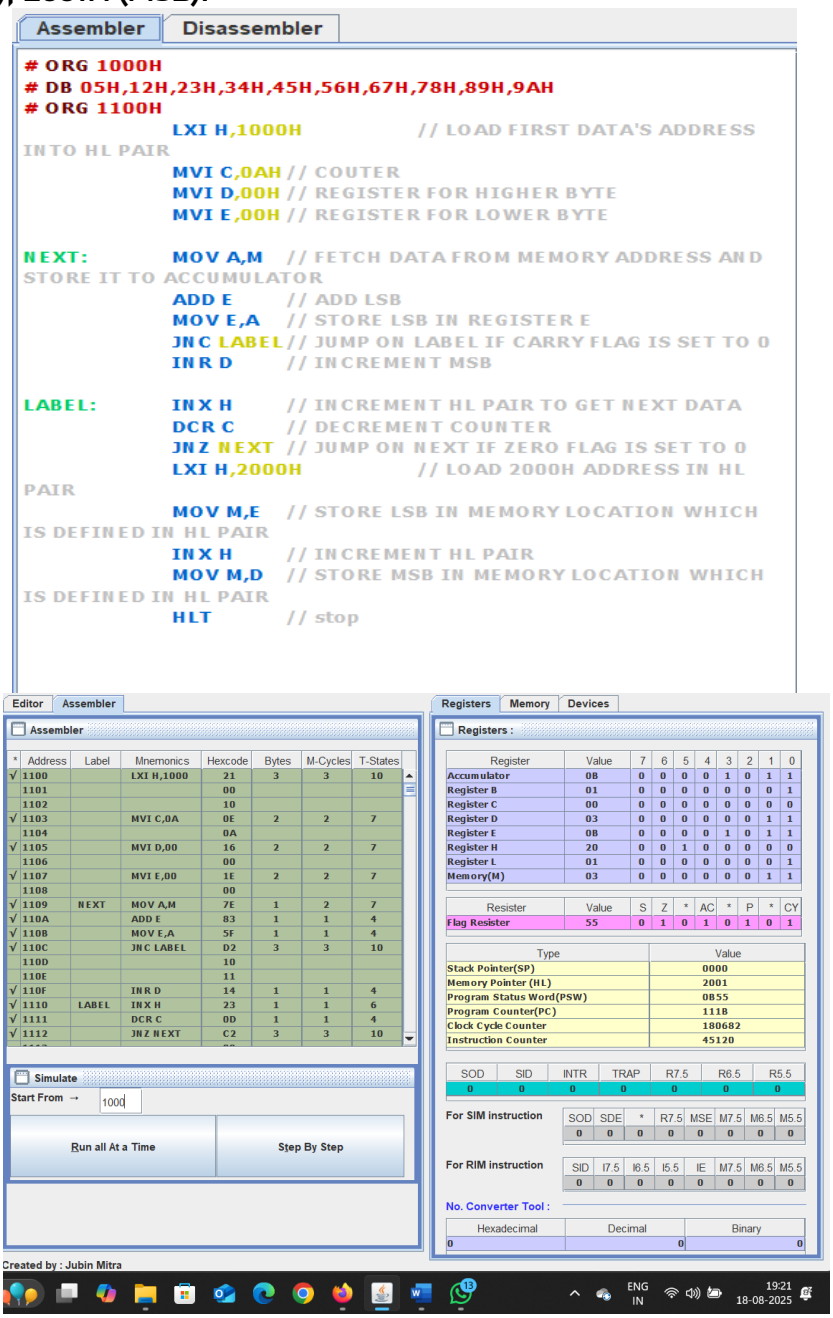
| Hexadecimal | Decimal | Binary |
|---|---|---|
| 0 | 0 | 0 |

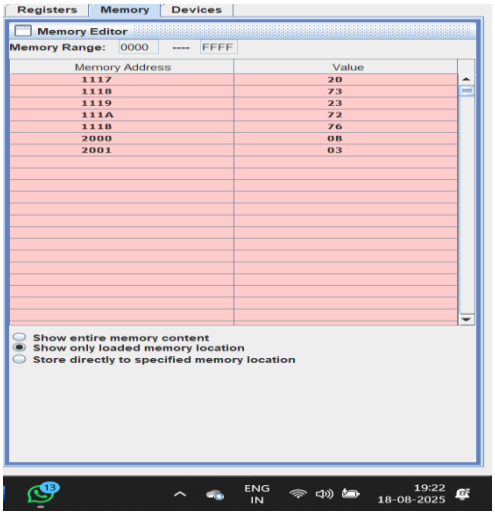
Created by : Jubin Mitra

ENG IN 19:16 18-08-2025

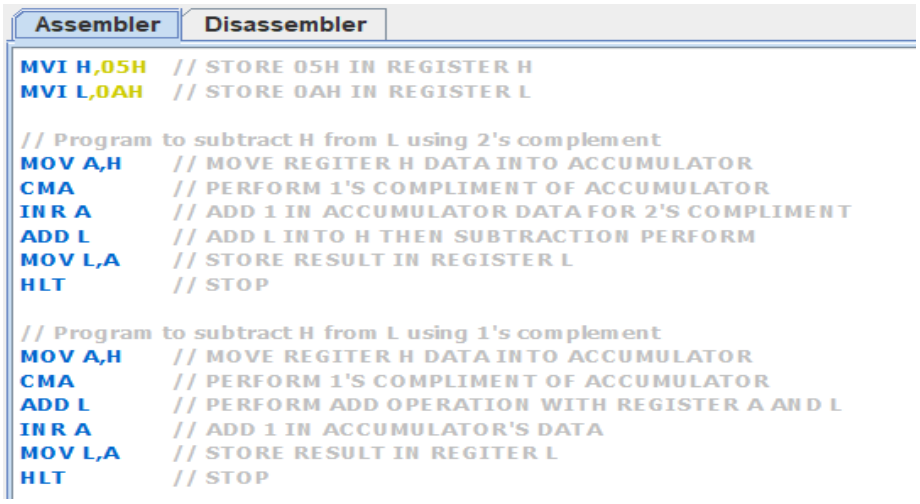



3. Write a program to add 10 bytes stored in a string starting from 1000H .Store result at 2000H (LSB), 2001H (MSB).

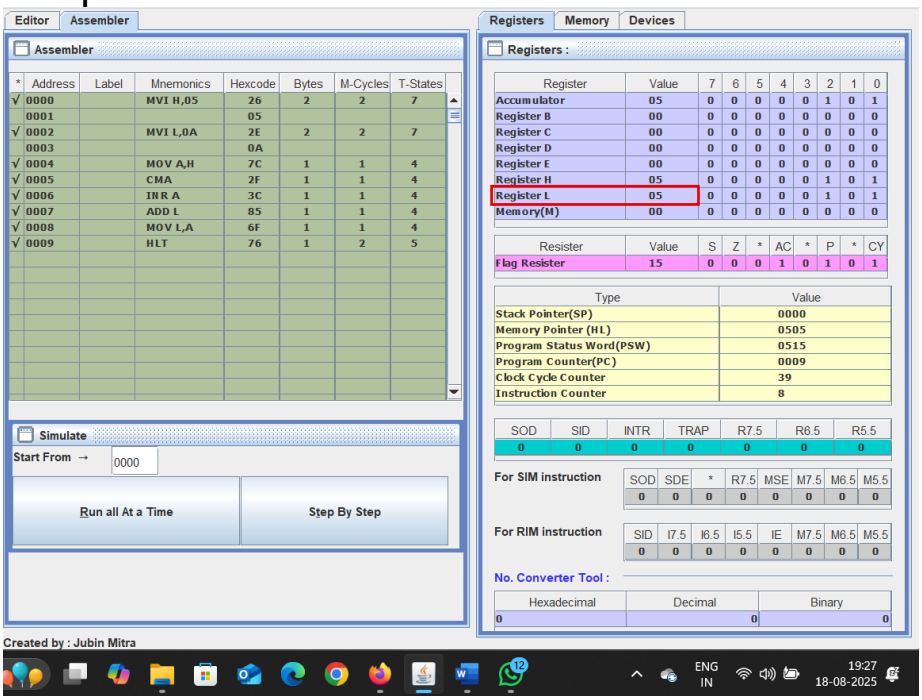





4. Write a program to subtract the contents of register H from register L without using any of subtract instruction.




(a) Apply 2's Complement method.



## (b) Apply 1's Complement method.