1. Write a program to add two 8-bit numbers and store the sum without carry.

**MVI C, 00H**

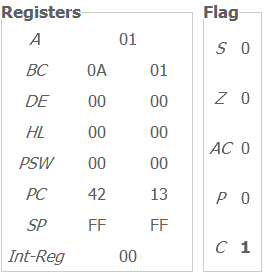
**MVI A, 250**

**MVI B, 10**

**ADD B**

**JNC NO\_CARRY**

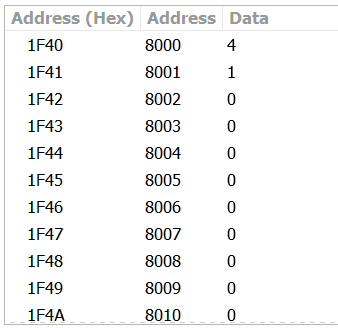
**INR C**

**NO\_CARRY: STA 8000**

**MOV A, C**

**STA 8001**

**HLT**

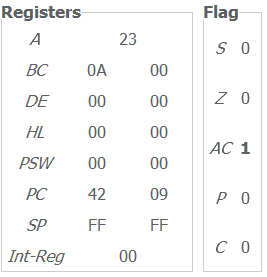
****

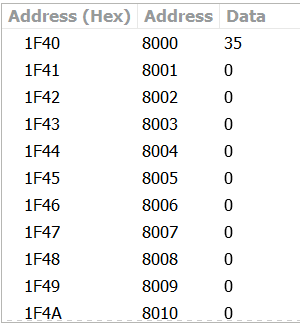
1. Write a program to add two 8-bit numbers and store the sum without carry.

**MVI A, 25**

**MVI B, 10**

**ADD B**

**STA 8000**

**HLT**

1. Write a program to subtract two 8-bit integers and store the difference with borrow.

**MVI C, 00H**

**MVI A, 33**

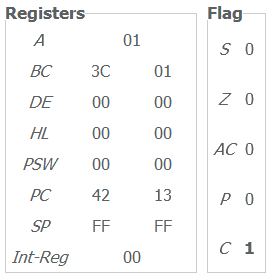
**MVI B,60**

**SUB B**

**JNC NO\_BORROW**

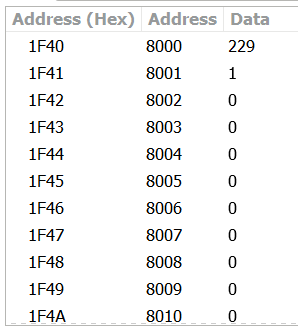
**INR C**

**NO\_BORROW: STA 8000**

**MOV A, C**

**STA 8001**

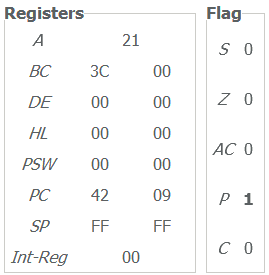
**HLT**

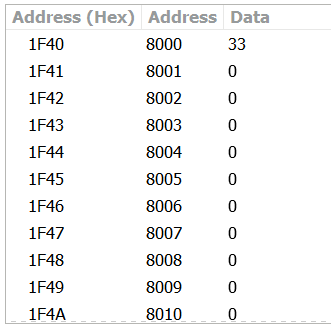
****4.Write a program to subtract two 8-bit integers and store the difference without borrow.

**MVI A,93**

**MVI B,60**

**SUB B**

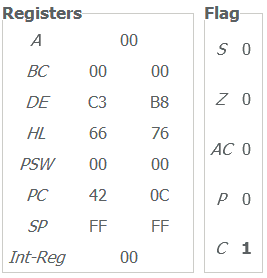
**STA 8000**

**HLT**

1. Write a program to add two 16-bit integers and store the sum without carry.

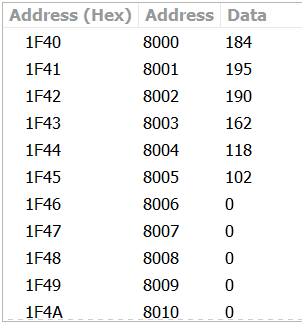
**LHLD 8000**

**XCHG**

**LHLD 8002**

**DAD D**

**SHLD 8004**

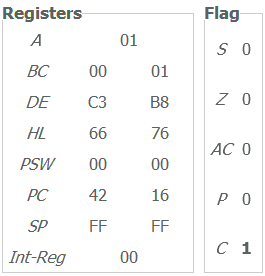
**HLT**

1. Write a program to add two 16-bit integers and store the sum with carry.

**LHLD 8000**

**XCHG**

**LHLD 8002**

**MVI C, 00H**

**DAD D**

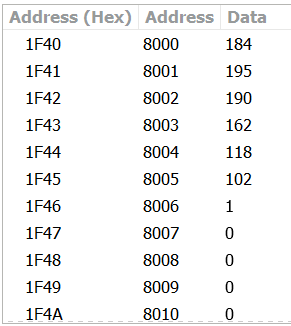
**JNC SKIP**

**INR C**

**SKIP: SHLD 8004**

**MOV A, C**

**STA 8006**

**HLT**

1. Write a program to subtract two 16-bit integers and store the difference without borrow.

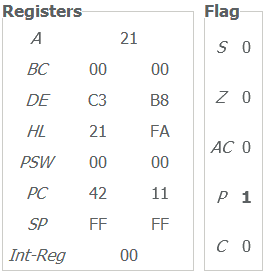
**LHLD** **8000**

**XCHG**

**LHLD 8002**

**MOV A, E**

**SUB L**

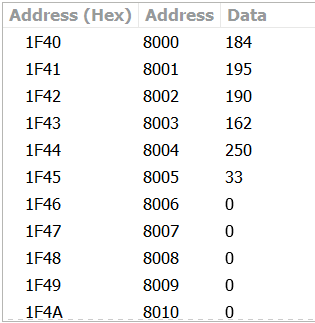
**MOV L, A**

**MOV A, D**

**SUB H**

**MOV H, A**

**SHLD 8004**

**HLT**

1. Write a program to subtract two 16-bit integers and store the difference with borrow.

**LHLD 8100H**

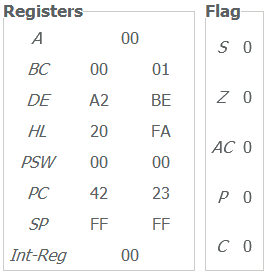
**XCHG**

**LHLD 8102H**

**MVI B, 00H**

**MVI C, 00H**

**MOV A, E**

**SUB L**

**JNC SKIP**

**INR C**

**SKIP: MOV E, A**

**MOV A, D**

**SUB C**

**SUB H**

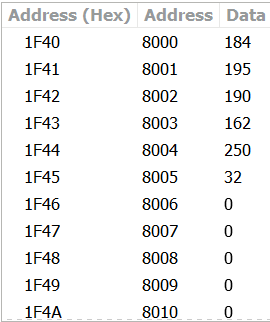
**JNC NO\_BORROW**

**INR B**

**NO\_BORROW: MOV D, A**

**XCHG**

**SHLD 8104H**

** MOV A, B**

**STA 8106H**

**HLT**

1. Write a program to find if the number is even or odd (If even store in 8000 and if odd store in 8001).

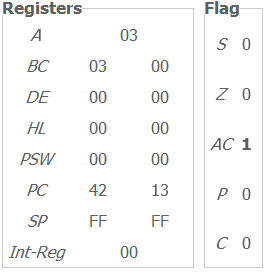
**MVI A, 3**

**MOV B, A**

**ANI 01H**

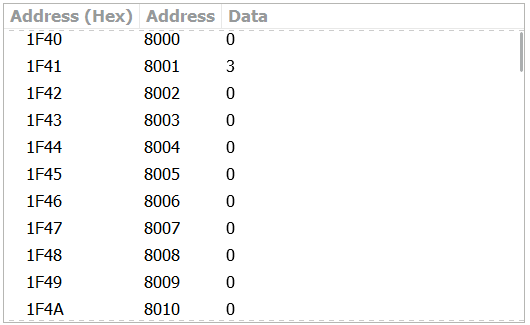
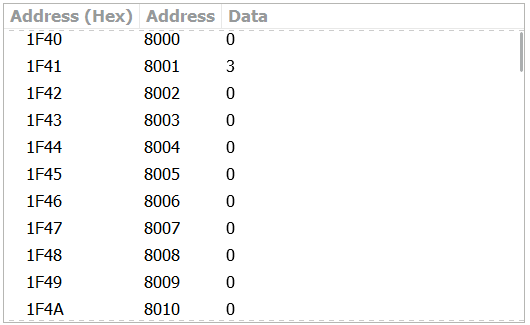
**MOV A, B**

**JZ EVEN**

**STA 8001**

**JMP END**

**EVEN: STA 8000**

**END: HLT**

1. Write a program to find if the number is positive or negative (If positive store in 8001 and if negative store in 8002).

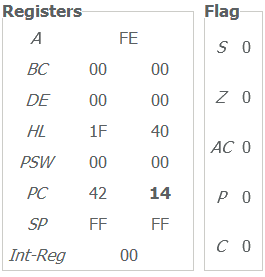
**LXI H, 8000**

**MOV A, M**

**RAL**

**JC NEG**

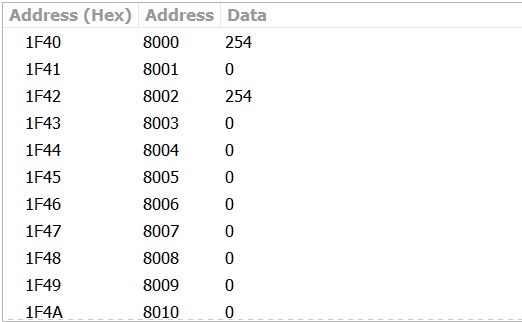
**RAR**

**STA 8001**

**JMP SKIP**

**NEG: RAR**

**STA 8002**

**SKIP: HLT**

1. Write a program to multiply two numbers and store the product in memory location 8000.

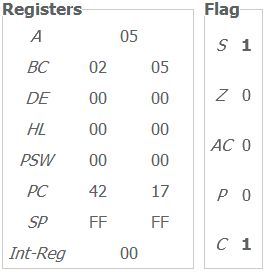
**LXI H,9H**

**MVI B,0AH**

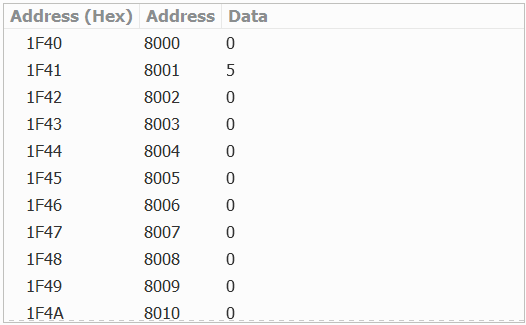
**XCHG**

**REPEAT: DAD D**

**DCR B**

**JNZ REPEAT**

**SHLD 8000**

**HLT**

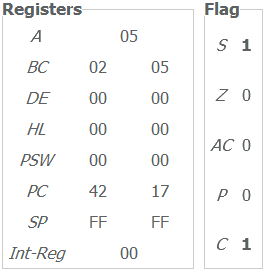
1. Write a program to divide two numbers and store remainder in 8000 and quotient in 8001.

**MVI A, 0AH**

**MVI B, 2**

**MVI C,00H**

**REPEAT: CMP B**

**JC STORE**

**SUB B**

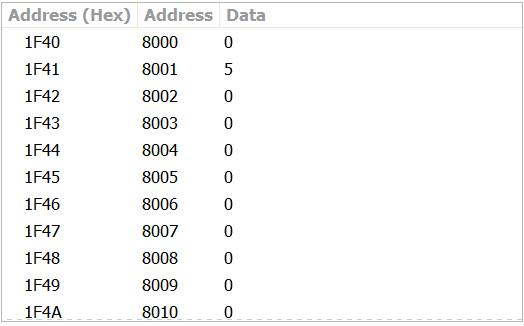
**INR C**

**JMP REPEAT**

**STORE: STA 8000**

**MOV A, C**

**STA 8001**

**HLT**

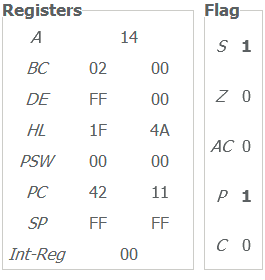
1. Write a program to store first 10 even natural numbers in memory location starting from 8100.

**MVI A,00H**

**MVI B,02H**

**LXI H,8000**

**MVI D,09**

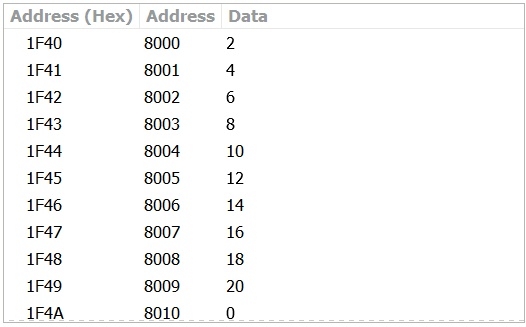
** LOOP: ADD B**

**MOV M, A**

**INX H**

**DCR D**

**JP LOOP**

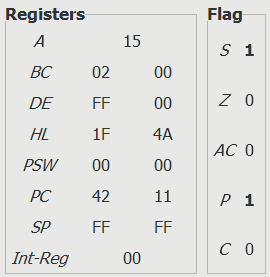
**HLT**

1. Write a program to store first 10 odd numbers in memory location starting from 8000.

**MVI A,01H**

**MVI B,02H**

**LXI H,8000**

**MVI D,09**

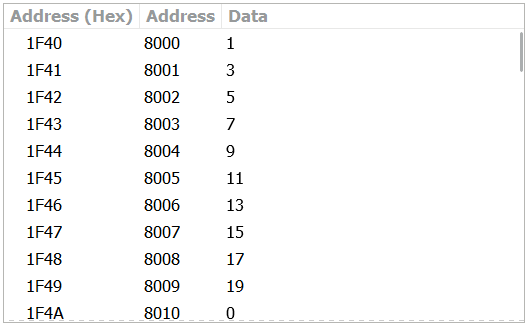
**LOOP: MOV M, A**

**ADD B**

**INX H**

**DCR D**

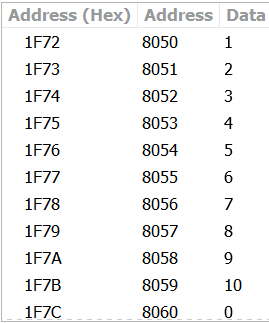
**JP LOOP**

**HLT**

1. There are 10 numbers stored from memory location 8050. Find those numbers are even or odd and store in memory location starting from 9000 if even and store in memory location starting from 9050.

**LXI D,9050**

**LXI H,8050**

**LXI B,9000**

**START: MOV A, M**

**ANI 01H**

**JZ EVEN**

**MOV A, M**

**INITIAL DATA**

**STAX D**

**INX D**

**JMP END**

**EVEN: MOV A, M**

**STAX B**

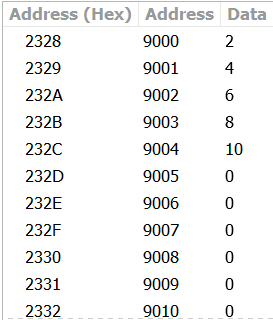
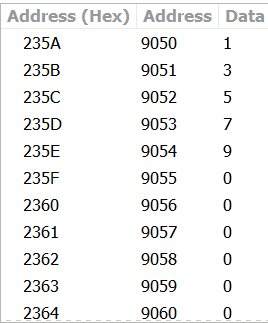
**INX B**

**END: INX H**

**MOV A, M**

**CPI 00H**

**JNZ START**

**HLT**

**ODD DATA**

**EVEN DATA**

1. Write a program to find greater and smaller number between two numbers and store the greater number in 8050 and the smaller number in 8051.

**MVI A, 4**

**MVI B, 5**

**CMP B**

**JC BGRT**

**STA 8050**

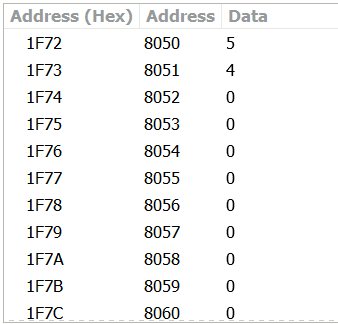
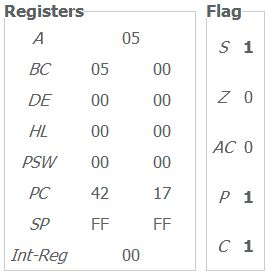
**MOV A, B**

**JMP STORE**

**BGRT: STA 8051**

**MOV A, B**

**STORE: STA 8050**

** HLT**

1. Write a program to find the largest number among three numbers stored in memory location 8000-8002 and store it in 8003.

**LDA 8000**

**MOV B, A**

**LDA 8001**

**CMP B**

**JNC AGRT**

**LOAD: LDA 8002**

**CMP B**

**JNC AGRT1**

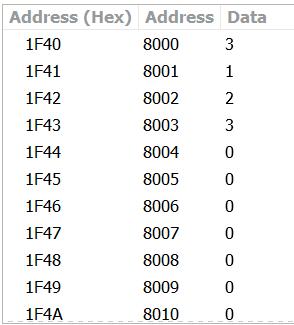
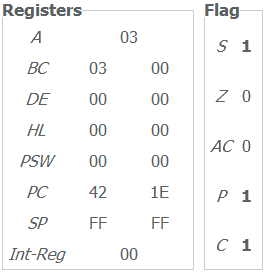
**MOV A, B**

**AGRT1: STA 8003**

**JMP END**

**AGRT: MOV B, A**

**JMP LOAD**

** END: HLT**

1. Write a program to find the smallest number among three numbers stored in memory location 8000-8002 and store it in 8003.

**LDA 8000**

**MOV B, A**

**LDA 8001**

**CMP B**

**JC AGRT**

**LOAD: LDA 8002**

**CMP B**

**JC AGRT1**

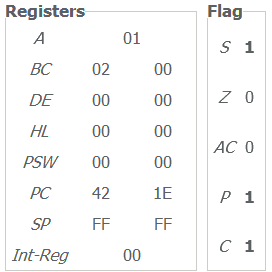
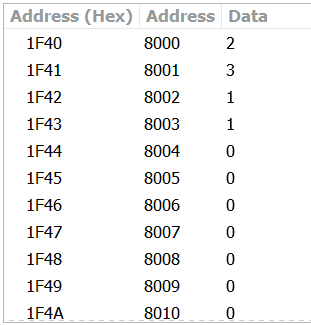
**MOV A, B**

**AGRT1: STA 8003**

**JMP END**

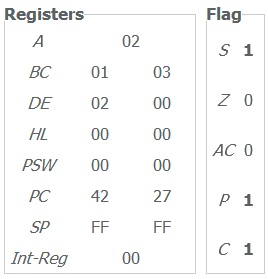
**AGRT: MOV B, A**

**JMP LOAD**

** END: HLT**

1. Write a program to find the middle number among three stored in memory location 8000-8002 and store it in 8003.

**MVI A, 2**

**MOV D, A**

**MVI B, 1**

**MVI C, 3**

**CMP B**

**JC BGRT**

**CMP C**

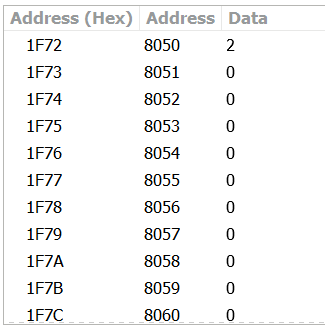
**JC STORE**

**MOV A, B**

**CMP C**

**JNC STORE**

**MOV A, C**

**JMP STORE**

**BGRT: MOV A, B**

**CMP C**

**JC STORE**

**MOV A, D**

**CMP C**

**JNC STORE**

**MOV A, C**

**STORE: STA 8050**

**HLT**

1. Write a program to sort the numbers in ascending order.

**MVI C,0AH**

**START: LXI H,8049**

**LOOP2: INX H**

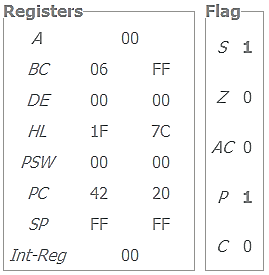
**LOOP1: INX H**

**MOV A, M**

**CPI 00H**

**JZ END**

**DCX H**

**MOV A, M**

**INX H**

**CMP M**

**JC LOOP1**

**MOV B, M**

**MOV M, A**

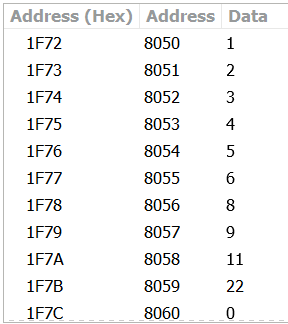
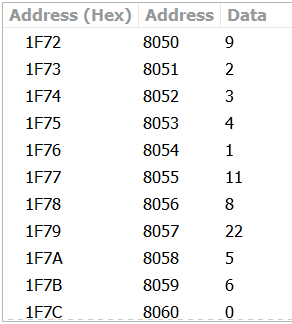
**DCX H**

**MOV M, B**

**JMP LOOP2**

**END: DCR C**

**JP START**

**HLT**

**INITIAL DATA**

**SORTED DATA**

1. Write a program to sort the numbers in descending order.

**MVI C,0AH** *//increase for sorting more numbers*

**START: LXI H,8049**

**LOOP2: INX H**

**LOOP1: INX H**

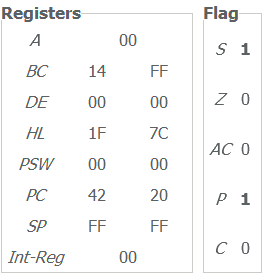
**MOV A, M**

**CPI 00H**

**JZ END**

**DCX H**

**MOV A, M**

**INX H**

**CMP M**

**JNC LOOP1**

**MOV B, M**

**MOV M, A**

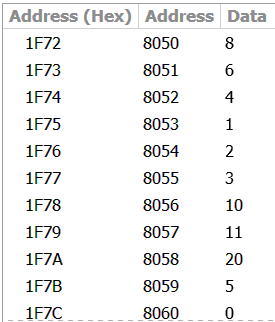
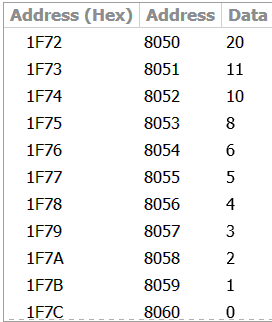
**DCX H**

**MOV M, B**

**JMP LOOP2**

**END: DCR C**

**JP START**

**HLT**

**SORTED DATA**

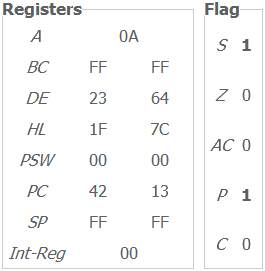
**INITIAL DATA**

1. There are 10 numbers in a memory block 8050-8059, transfer them in another memory block 9050-9059.

**MVI B,09H**

**LXI H,8050**

**LXI D,9050**

**LOOP: MOV A, M**

**XCHG**

**MOV M, A**

**INX H**

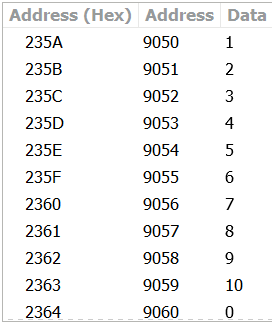
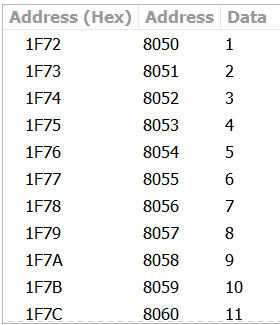
**XCHG**

**INX H**

**DCR B**

**JP LOOP**

**HLT**



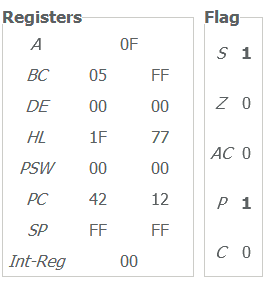
**TRANSFERRED DATA**

**INITIAL DATA**

1. Write a program to find the sum of five numbers stored in a memory block 8050-8054 and store the sum in 8060.

**MVI A,00H**

**MVI C,04H**

**LXI H,8050**

**LOOP: MOV B, M**

**ADD B**

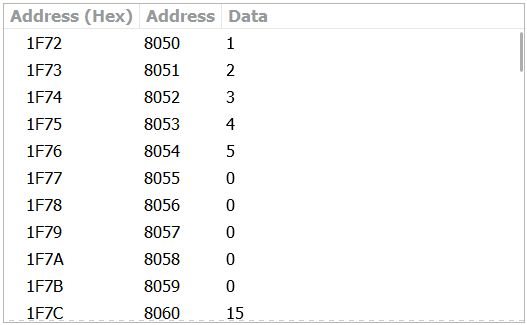
**INX H**

**DCR C**

**JP LOOP**

**STA 8060**

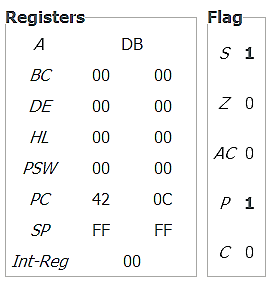
**HLT**



1. Write a program to find the 1’s complement and 2’s complement of any number and store the 1’s complement in 8050 and 2’s complement in 8051.

**MVI A,37**

**CMA**

**STA 8050­­**

**ADI 01H**

**STA 8051**

**HLT**

