CHANDNI JHA

REGISTRATION 690

ROLL CSE21030

Assignment 4

Github Link : https://github.com/ChandniJha630/CSC512-MIcroprocessor-And-Microcontroller-Lab/tree/main/Assignment%204

INDEX

PROB.1 FIVE NUMBERS ARE STORED IN CONSECUTIVE MEMORY LOCATIONS STARTING FROM 9000H. SORT THE NUMBER IN ASCENDING/DESCENDING ORDER.

CODE:

```
;&lt;Program title&gt;
jmp start
;data
;code
start: nop
MVI C,05H
DCR C
OutLoop: MOV D,C
LXI H,9000H
InLoop: MOV A,M
INX H
CMP M
JC Skip
MOV B,M
MOV M,A
DCX H
MOV M,B
INX H
Skip: DCR D
JNZ InLoop
DCR C
JNZ OutLoop
hlt
```

ALGORITHM

The value 5 is loaded into register C.

The "OutLoop" label indicates an outer loop that repeats as long as the value in C is not zero.

Inside the outer loop, the value in register C is copied to register D.

The address 9000H is loaded into register pair H-L.

The "InLoop" label indicates an inner loop that iterates through the memory addresses.

Inside the inner loop, the value at the memory location pointed to by HL is loaded into register A.

The value in A is compared with the value at the current memory location.

If A is less than M (value at memory location), a swap operation occurs using registers A and B.

After swapping, the inner loop moves back one memory location, and the swap is completed.

The "Skip" label is used to bypass the swap if the values are not swapped.

The value in register D is decremented.

If D is not zero, the inner loop continues.

After the inner loop completes, the value in register C is decremented.

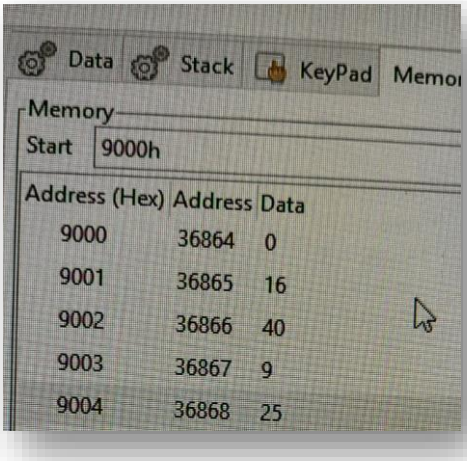
If C is not zero, the outer loop continues.

Once the outer loop finishes, the sorting operation is complete.

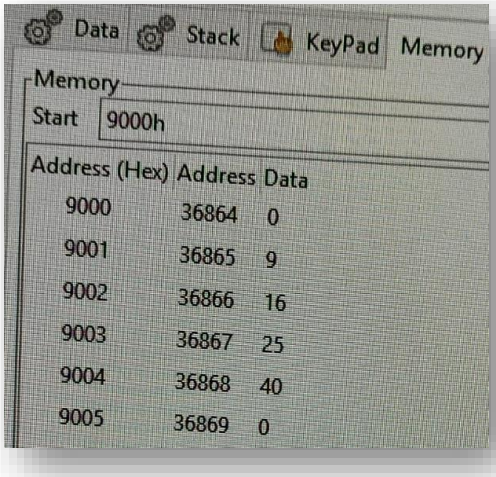
CODE

```
;&lt;Program title&gt;

jmp start

;data

;code
start: nop
MVI C,05H
DCR C

OutLoop: MOV D,C
LXI H,9000H

InLoop: MOV A,M
INX H
CMP M

JC Skip
MOV B,M
MOV M,A
DCX H
MOV M,B
INX H
Skip: DCR D
JNZ InLoop

DCR C
JNZ OutLoop

hlt
```

INPUT



OUTPUT



PROB.2 GENERATE THE FIRST 10 ELEMENTS OF THE FIBONACCI SERIES.

CODE

```
;&lt;Program title&gt;
jmp start
;data
;code
start: nop
LXI H,8C00H
XRA A
STA 8C00H
MOV C,M
INX H
MVI A,01H
STA 8C01H
MOV D,M
MVI B,08H
LOOP: XRA A
ADD C
ADD D
INX H
MOV M,A
MOV C,D
MOV D,M
DCR B
JNZ LOOP
hlt
```

ALGORITHM

LXI H, 8C00H: This loads the immediate value 8C00H into the register pair H-L. This sets up a memory address for further operations.

XRA A: This performs an exclusive OR operation between the accumulator A and itself, effectively setting the accumulator A to zero.

STA 8C00H: This stores the value of the accumulator A (which is now 0) into memory location 8C00H.

MOV C, M: This moves the value from the memory location pointed to by HL into register C.

INX H: This increments the value in register pair H-L, pointing to the next memory location.

MVI A, 01H: This moves the immediate value 01H into the accumulator A.

STA 8C01H: This stores the value of the accumulator A (which is 01H) into memory location 8C01H.

MOV D, M: This moves the value from the memory location pointed to by HL into register D.

MVI B, 08H: This moves the immediate value 08H into register B.

LOOP: This label marks the beginning of a loop.

XRA A: This XORs the accumulator A with itself, effectively setting it to zero.

ADD C: This adds the value in register C to the accumulator A.

ADD D: This adds the value in register D to the accumulator A.

INX H: This increments the value in register pair H-L, pointing to the next memory location.

MOV M, A: This moves the value in the accumulator A to the memory location pointed to by HL.

MOV C, D: This moves the value in register D to register C.

MOV D, M: This moves the value from the memory location pointed to by HL into register D.

DCR B: This decrements the value in register B.

CODE IMAGE

Load me at

```
;&lt;Program title&gt;

jmp start

;data

;code
start: nop
LXI H,8C00H
XRA A
STA 8C00H
MOV C,M
INX H
MVI A,01H
STA 8C01H
MOV D,M
MVI B,08H
LOOP: XRA A
ADD C
ADD D
INX H
MOV M,A
MOV C,D
MOV D,M
DCR B
JNZ LOOP

hlt
```

OUTPUT

Data | Stack | KeyPad | Memory | I/O

Memory

Start 8C00h

Address (Hex)	Address	Data
8C00	35840	0
8C01	35841	1
8C02	35842	1
8C03	35843	2
8C04	35844	3
8C05	35845	5
8C06	35846	8
8C07	35847	13
8C08	35848	21
8C09	35849	34
8C0A	35850	0

PROB.3 ADDITION/SUBTRACTION OF TWO NUMBERS GIVEN THROUGH PORTS. RESULT WILL ALSO BE STORED IN SOME PORT ADDRESS.

CODE

ADDITION

```
;&lt;Program title&gt;
jmp start
;data
;code
start: nop
IN 02H
MOV B,A
IN 01H
ADD B
OUT 03H
Hlt
```

SUBTRACTION

```
;&lt;Program title&gt;
jmp start
;data
;code
start: nop
IN 02H
MOV B,A
IN 01H
SUB B
OUT 03H
Hlt
```

ALGORITHM

Read an input value from port 02H and store it in the accumulator A.

Copy the value from the accumulator A into register B.

Read another input value from port 01H and overwrite the value in the accumulator A.

Add/Subtract the value in register B to the new value in the accumulator A and store the result in the accumulator A.

CODE IMAGES

ADDITION

```
;&lt;Program title&gt;

jmp start

;data

;code
start: nop

IN 02H
MOV B,A
IN 01H
ADD B
OUT 03H
Hlt
```

SUBTRACTION

```
Load me at

;&lt;Program title&gt;

jmp start

;data

;code
start: nop

IN 02H
MOV B,A
IN 01H
SUB B
OUT 03H
Hlt
```

INPUTS – OUTPUTS

ADDITION

Data | Stack | KeyPad | Memory | I/O Ports

I/O Ports

Start

Address (Hex)	Address	Data
00	0	0
01	1	5
02	2	4
03	3	9
04	4	0

SUBTRACTION

Data | Stack | KeyPad | Memory | I/O Ports

I/O Ports

Start

Address (Hex)	Address	Data
00	0	0
01	1	5
02	2	4
03	3	1