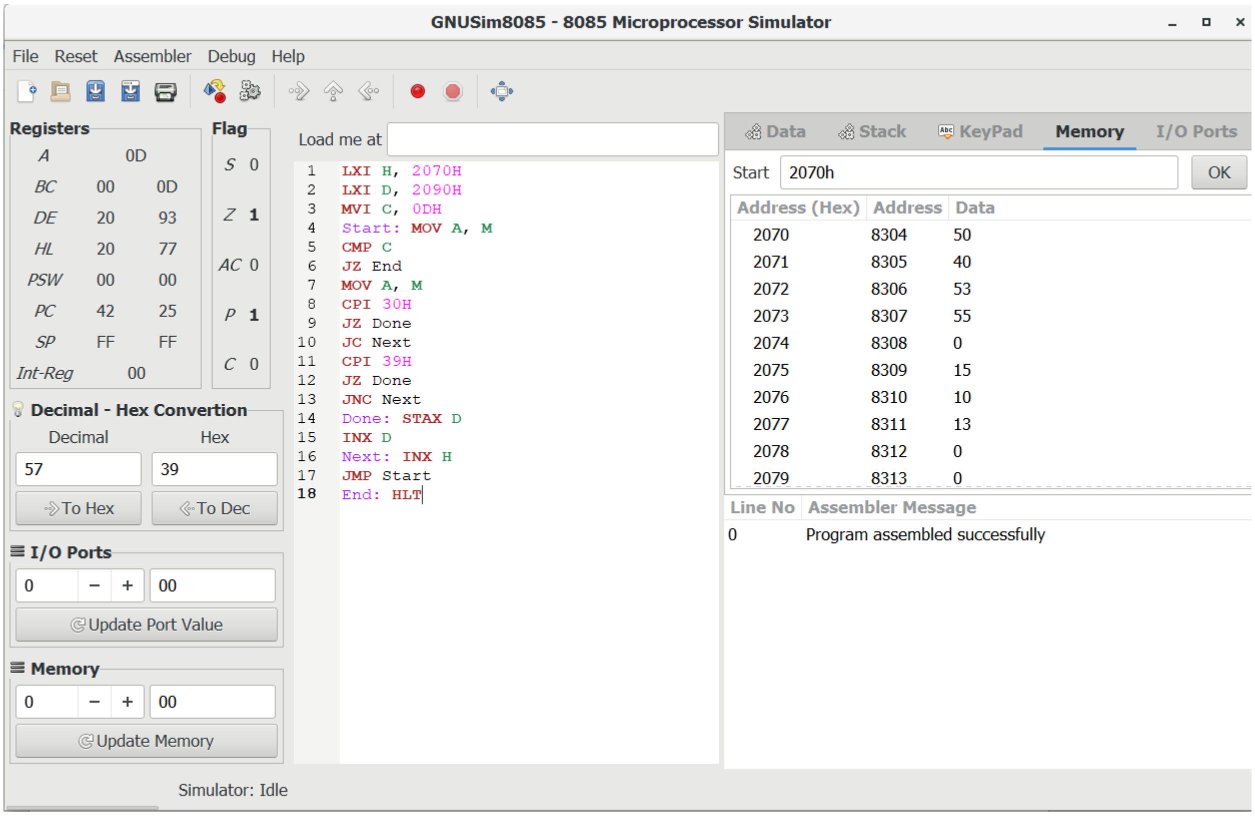
U18CO018
Shubham Shekhaliya
Assignment-5 (MIT)

1-> A string of readings is stored in memory, locations starting at 2070H, and the end of the string is indicated by the byte 0DH.Write a program to check each byte in the string, and the save the bytes in the range of 30H to 39H (both inclusive) in memory locations starting from 2090H.

Code: -

```
LXI H, 2070H
LXI D, 2090H
MVI C, 0DH
Start: MOV A, M
CMP C
JZ End
MOV A, M
CPI 30H
JZ Done
JC Next
CPI 39H
JZ Done
JNC Next
Done: STAX D
INX D
Next: INX H
JMP Start
End: HLT
```



Registers		
A	0D	
BC	00	0D
DE	20	93
HL	20	77
PSW	00	00
PC	42	25
SP	FF	FF
Int-Reg	00	

Flag	
S	0
Z	1
AC	0
P	1
C	0

Decimal - Hex Convertion

Decimal: 57 Hex: 39

I/O Ports: 0 00

Memory: 0 00

Simulator: Idle

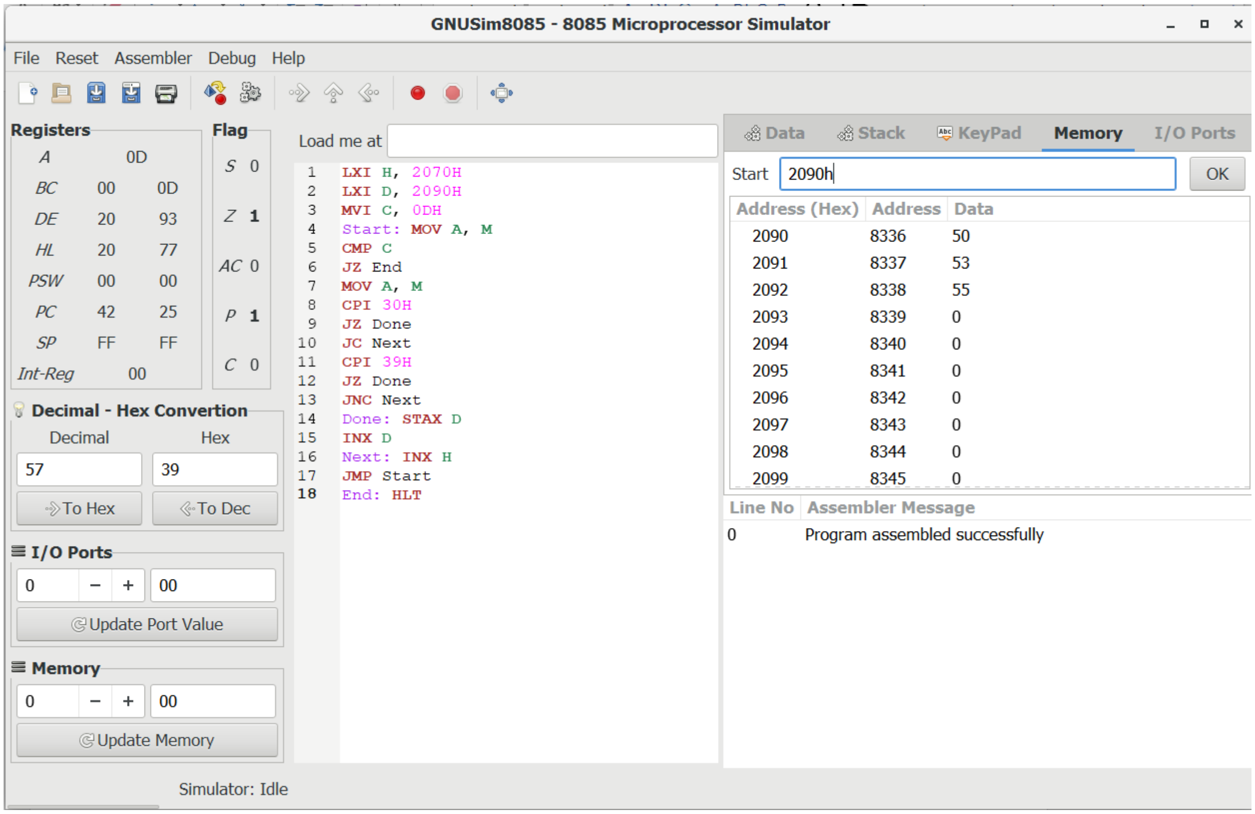
Load me at

```
LXI H, 2070H
LXI D, 2090H
MVI C, 0DH
Start: MOV A, M
CMP C
JZ End
MOV A, M
CPI 30H
JZ Done
JC Next
CPI 39H
JZ Done
JNC Next
Done: STAX D
INX D
Next: INX H
JMP Start
End: HLT
```

Start 2070h

Address (Hex)	Address	Data
2070	8304	50
2071	8305	40
2072	8306	53
2073	8307	55
2074	8308	0
2075	8309	15
2076	8310	10
2077	8311	13
2078	8312	0
2079	8313	0

Line No	Assembler Message
0	Program assembled successfully



Registers		
A	0D	
BC	00	0D
DE	20	93
HL	20	77
PSW	00	00
PC	42	25
SP	FF	FF
Int-Reg	00	

Flag	
S	0
Z	1
AC	0
P	1
C	0

Decimal - Hex Convertion

Decimal: 57 Hex: 39

I/O Ports: 0 00

Memory: 0 00

Simulator: Idle

Load me at

```
LXI H, 2070H
LXI D, 2090H
MVI C, 0DH
Start: MOV A, M
CMP C
JZ End
MOV A, M
CPI 30H
JZ Done
JC Next
CPI 39H
JZ Done
JNC Next
Done: STAX D
INX D
Next: INX H
JMP Start
End: HLT
```

Start 2090h

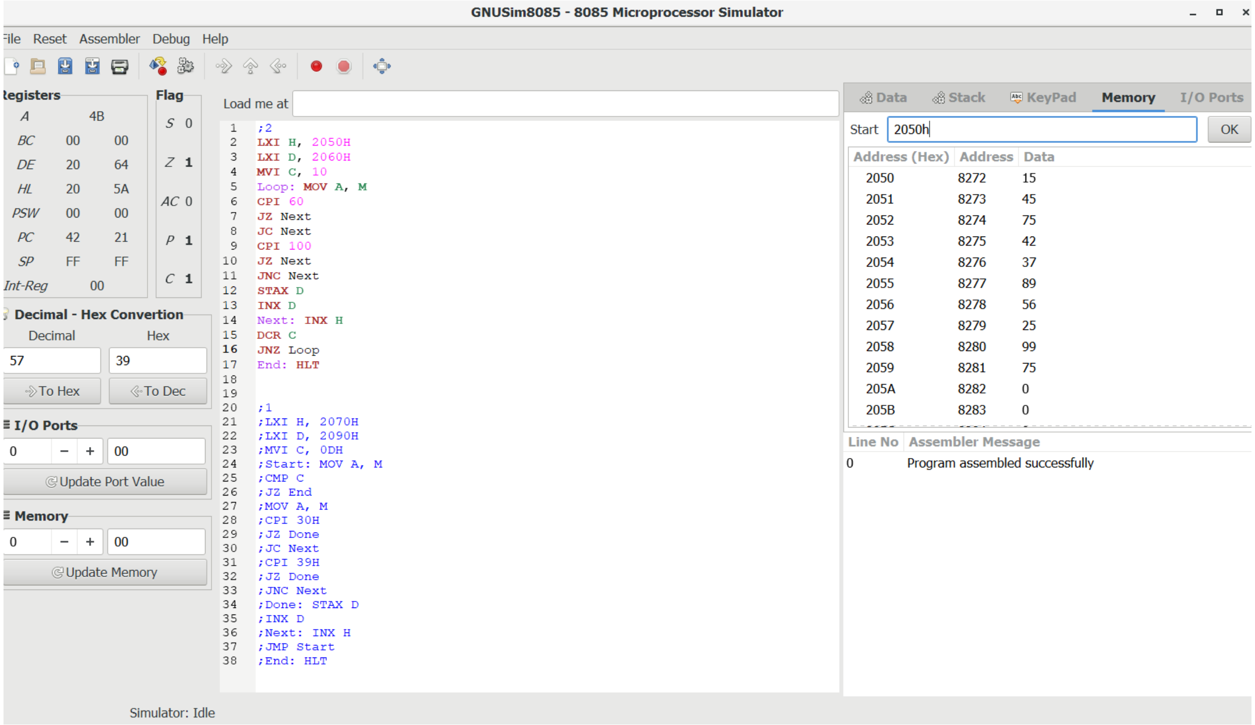
Address (Hex)	Address	Data
2090	8336	50
2091	8337	53
2092	8338	55
2093	8339	0
2094	8340	0
2095	8341	0
2096	8342	0
2097	8343	0
2098	8344	0
2099	8345	0

Line No	Assembler Message
0	Program assembled successfully

2-> A set of ten bytes is stored in memory starting with the address 2050H.Write a program to check each byte , and save the bytes that are higher than 6010 and lower than 10010 in memory locations starting from 2060H.

Code:-

```
LXI H, 2050H
LXI D, 2060H
MVI C, 10
Loop: MOV A, M
CPI 60
JZ Next
JC Next
CPI 100
JZ Next
JNC Next
STAX D
INX D
Next: INX H
DCR C
JNZ Loop
End: HLT
```



Registers		
A	4B	
BC	00	00
DE	20	64
HL	20	5A
PSW	00	00
PC	42	21
SP	FF	FF
Int-Reg	00	

Flag	
S	0
Z	1
AC	0
P	1
C	1

Decimal - Hex Convertion

Decimal: 57 Hex: 39

I/O Ports: 0 − + 00

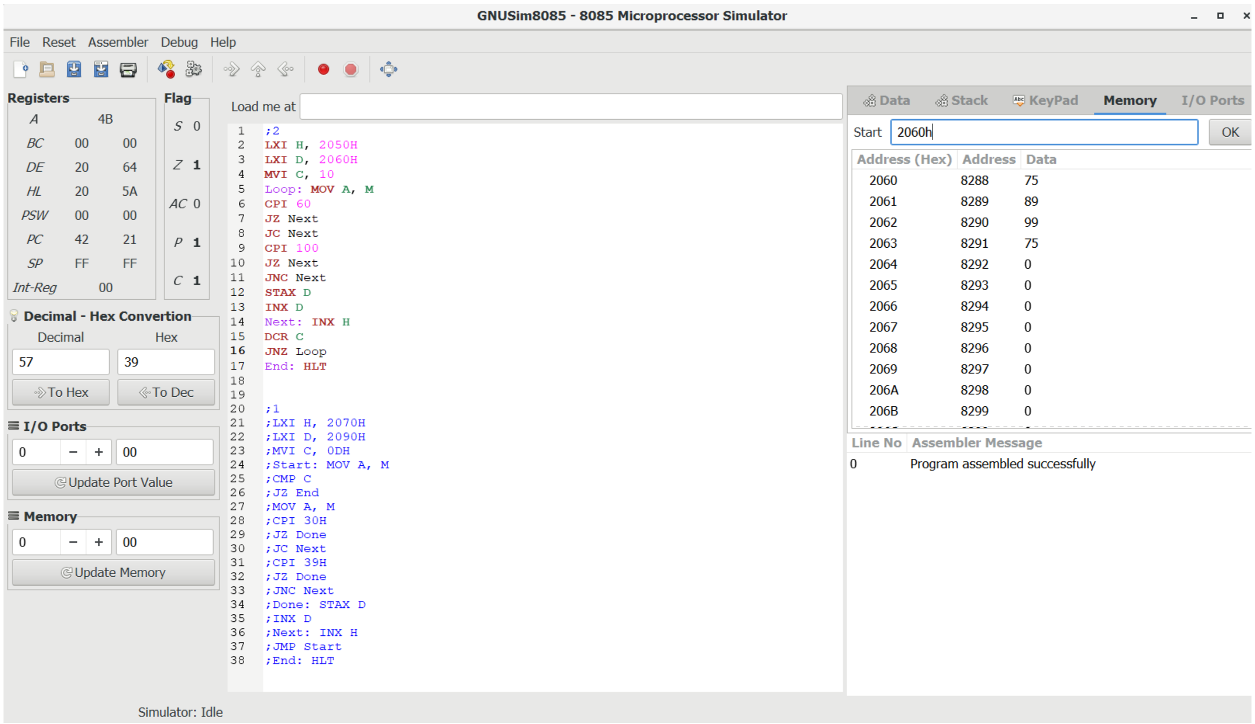
Memory: 0 − + 00

```
;2
LXI H, 2050H
LXI D, 2060H
MVI C, 10
Loop: MOV A, M
CPI 60
JZ Next
JC Next
CPI 100
JZ Next
JNC Next
STAX D
INX D
Next: INX H
DCR C
JNZ Loop
End: HLT


;1
;LXI H, 2070H
;LXI D, 2090H
;MVI C, 0DH
;Start: MOV A, M
;CMP C
;JZ End
;MOV A, M
;CPI 30H
;JZ Done
;JC Next
;CPI 39H
;JZ Done
;JNC Next
;Done: STAX D
;INX D
;Next: INX H
;JMP Start
;End: HLT
```

Start: 2050h

Address (Hex)	Address	Data
2050	8272	15
2051	8273	45
2052	8274	75
2053	8275	42
2054	8276	37
2055	8277	89
2056	8278	56
2057	8279	25
2058	8280	99
2059	8281	75
205A	8282	0
205B	8283	0

Line No	Assembler Message
0	Program assembled successfully



Registers		
A	4B	
BC	00	00
DE	20	64
HL	20	5A
PSW	00	00
PC	42	21
SP	FF	FF
Int-Reg	00	

Flag	
S	0
Z	1
AC	0
P	1
C	1

Decimal - Hex Convertion

Decimal	Hex
57	39

I/O Ports: 0 − + 00

Memory: 0 − + 00

```
;2
LXI H, 2050H
LXI D, 2060H
MVI C, 10
Loop: MOV A, M
CPI 60
JZ Next
JC Next
CPI 100
JZ Next
JNC Next
STAX D
INX D
Next: INX H
DCR C
JNZ Loop
End: HLT


;1
;LXI H, 2070H
;LXI D, 2090H
;MVI C, 0DH
;Start: MOV A, M
;CMP C
;JZ End
;MOV A, M
;CPI 30H
;JZ Done
;JC Next
;CPI 39H
;JZ Done
;JNC Next
;Done: STAX D
;INX D
;Next: INX H
;JMP Start
;End: HLT
```

Start 2060h

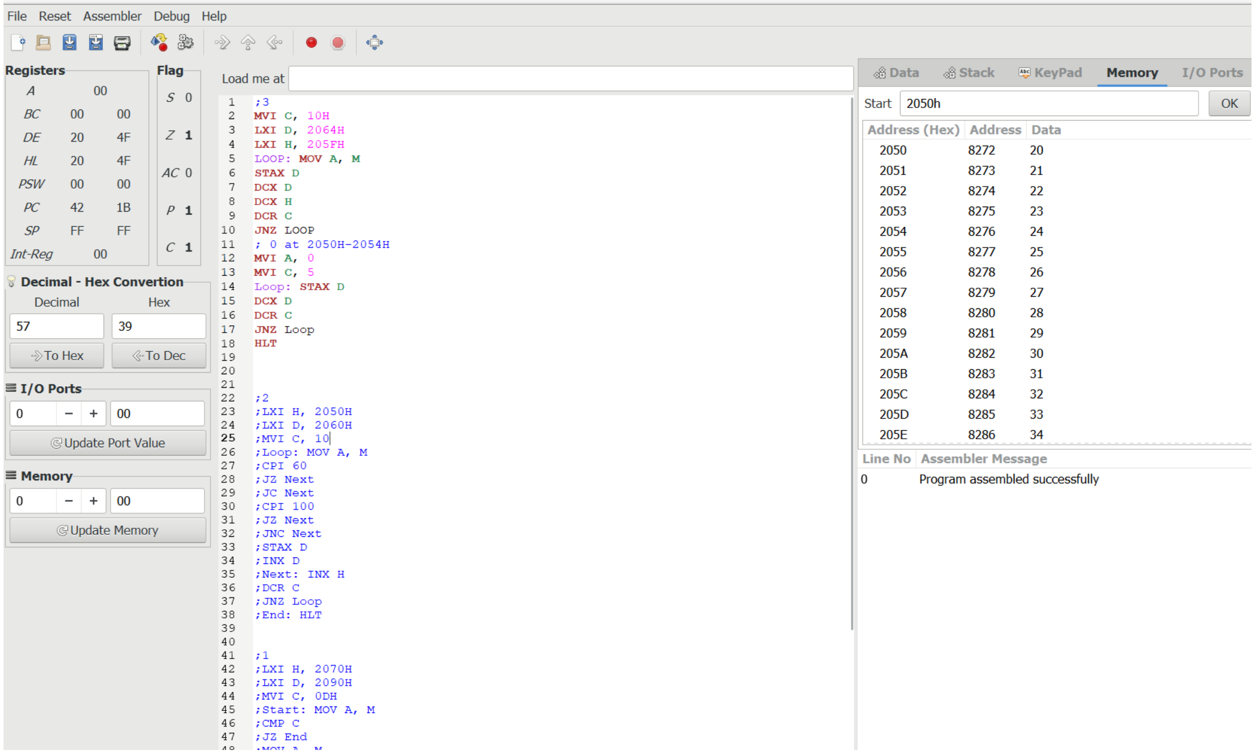
Address (Hex)	Address	Data
2060	8288	75
2061	8289	89
2062	8290	99
2063	8291	75
2064	8292	0
2065	8293	0
2066	8294	0
2067	8295	0
2068	8296	0
2069	8297	0
206A	8298	0
206B	8299	0

Line No	Assembler Message
0	Program assembled successfully

Simulator: Idle

3-> Data bytes are stored in memory locations from 2050H to 205FH.To insert an additional five bytes of data, It is necessary to shift the data string by five memory locations. Write a program to store the data string from 2055H to 2064H.Use any sixteen bytes of data to verify your program.

```
MVI C, 10H
LXI D, 2064H
LXI H, 205FH
LOOP: MOV A, M
STAX D
DCX D
DCX H
DCR C
JNZ LOOP
; 0 at 2050H-2054H
MVI A, 0
MVI C, 5
Loop: STAX D
DCX D
DCR C
JNZ Loop
HLT
```



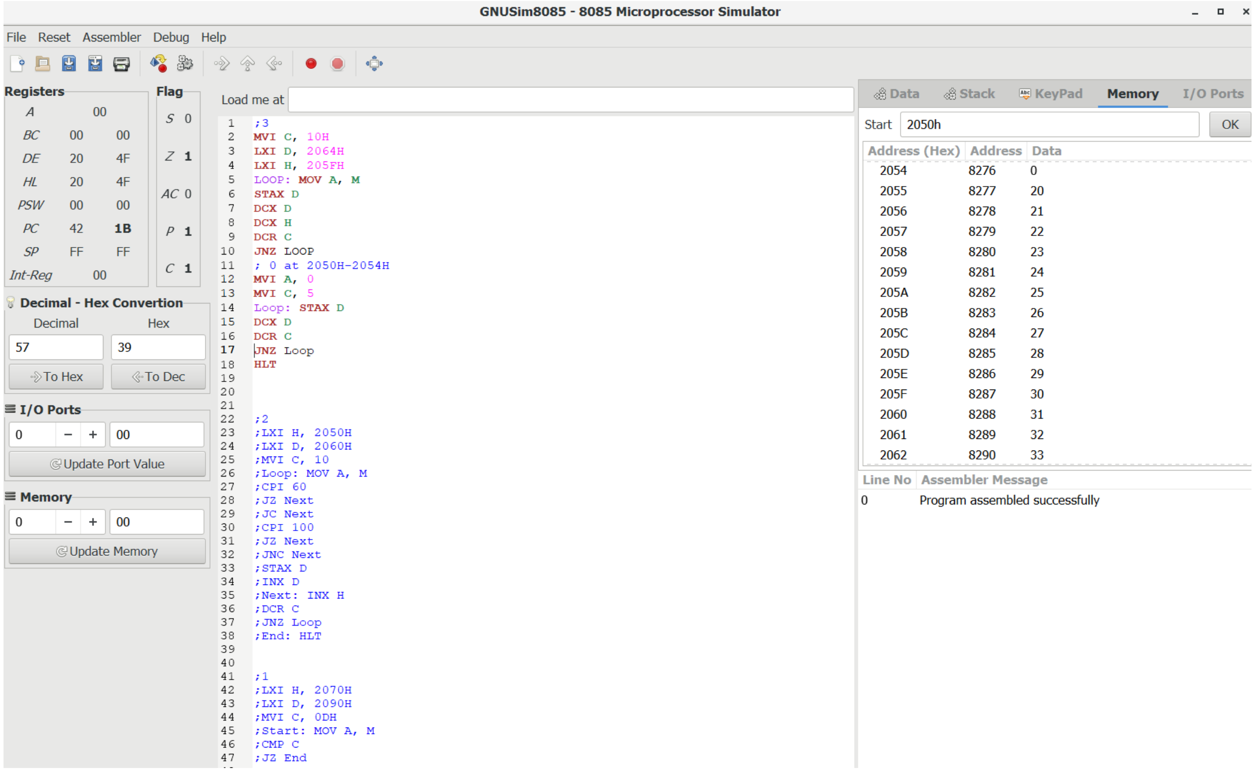
Registers		
A	00	
BC	00	00
DE	20	4F
HL	20	4F
PSW	00	00
PC	42	1B
SP	FF	FF
Int-Reg	00	

Flag	
S	0
Z	1
AC	0
P	1
C	1

```
;3
MVI C, 10H
LXI D, 2064H
LXI H, 205FH
LOOP: MOV A, M
STAX D
DCX D
DCX H
DCR C
JNZ LOOP
; 0 at 2050H-2054H
MVI A, 0
MVI C, 5
Loop: STAX D
DCX D
DCR C
JNZ Loop
HLT



;2
;LXI H, 2050H
;LXI D, 2060H
;MVI C, 10
;Loop: MOV A, M
;CPI 60
;JZ Next
;JC Next
;CPI 100
;JZ Next
;JNC Next
;STAX D
;INX D
;Next: INX H
;DCR C
;JNZ Loop
;End: HLT


;1
;LXI H, 2070H
;LXI D, 2090H
;MVI C, 0DH
;Start: MOV A, M
;CMP C
;JZ End
;MOV A, M
```

Address (Hex)	Address	Data
2050	8272	20
2051	8273	21
2052	8274	22
2053	8275	23
2054	8276	24
2055	8277	25
2056	8278	26
2057	8279	27
2058	8280	28
2059	8281	29
205A	8282	30
205B	8283	31
205C	8284	32
205D	8285	33
205E	8286	34

Line No	Assembler Message
0	Program assembled successfully



Registers		
A	00	
BC	00	00
DE	20	4F
HL	20	4F
PSW	00	00
PC	42	1B
SP	FF	FF
Int-Reg	00	

Flag	
S	0
Z	1
AC	0
P	1
C	1

```
;3
MVI C, 10H
LXI D, 2064H
LXI H, 205FH
LOOP: MOV A, M
STAX D
DCX D
DCX H
DCR C
JNZ LOOP
; 0 at 2050H-2054H
MVI A, 0
MVI C, 5
Loop: STAX D
DCX D
DCR C
JNZ Loop
HLT



;2
;LXI H, 2050H
;LXI D, 2060H
;MVI C, 10
;Loop: MOV A, M
;CPI 60
;JZ Next
;JC Next
;CPI 100
;JZ Next
;JNC Next
;STAX D
;INX D
;Next: INX H
;DCR C
;JNZ Loop
;End: HLT


;1
;LXI H, 2070H
;LXI D, 2090H
;MVI C, 0DH
;Start: MOV A, M
;CMP C
;JZ End
```

Address (Hex)	Address	Data
2054	8276	0
2055	8277	20
2056	8278	21
2057	8279	22
2058	8280	23
2059	8281	24
205A	8282	25
205B	8283	26
205C	8284	27
205D	8285	28
205E	8286	29
205F	8287	30
2060	8288	31
2061	8289	32
2062	8290	33

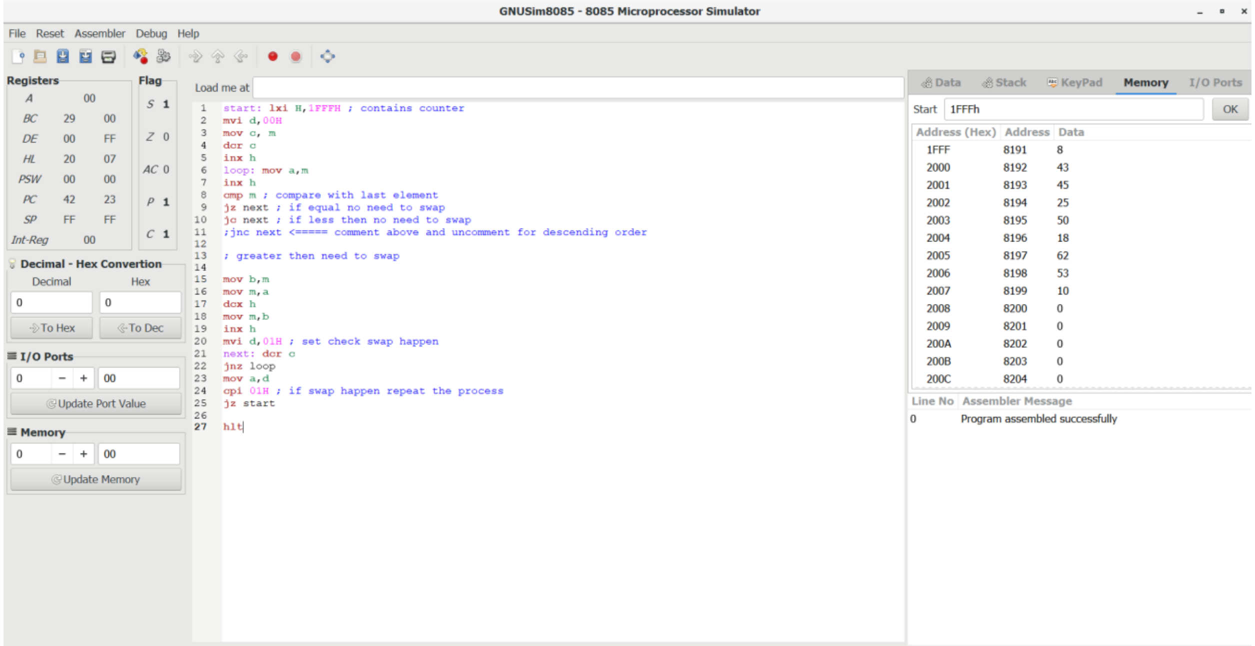
Line No	Assembler Message
0	Program assembled successfully

4-> Write a Program to Sort the array in ascending order/descending order.

Code:-

```
start: lxi H,1FFFH ; contains counter
mvi d,00H
mov c, m
dcr c
inx h
loop: mov a,m
inx h
cmp m ; compare with last element
jz next ; if equal no need to swap
jc next ; if less then no need to swap
;jnc next <===== comment above and uncomment for descending order
; greater then need to swap
mov b,m
mov m,a
dcx h
mov m,b
inx h
mvi d,01H ; set check swap happen
next: dcr c
jnz loop
mov a,d
cpi 01H ; if swap happen repeat the process
jz start
hlt
```

Before Sort



Registers		
A	00	
BC	29	00
DE	00	FF
HL	20	07
PSW	00	00
PC	42	23
SP	FF	FF
Int-Reg	00	

Flag	
S	1
Z	0
AC	0
P	1
C	1

Decimal: 0 Hex: 0

I/O Ports: 0 − + 00

Memory: 0 − + 00

Load me at

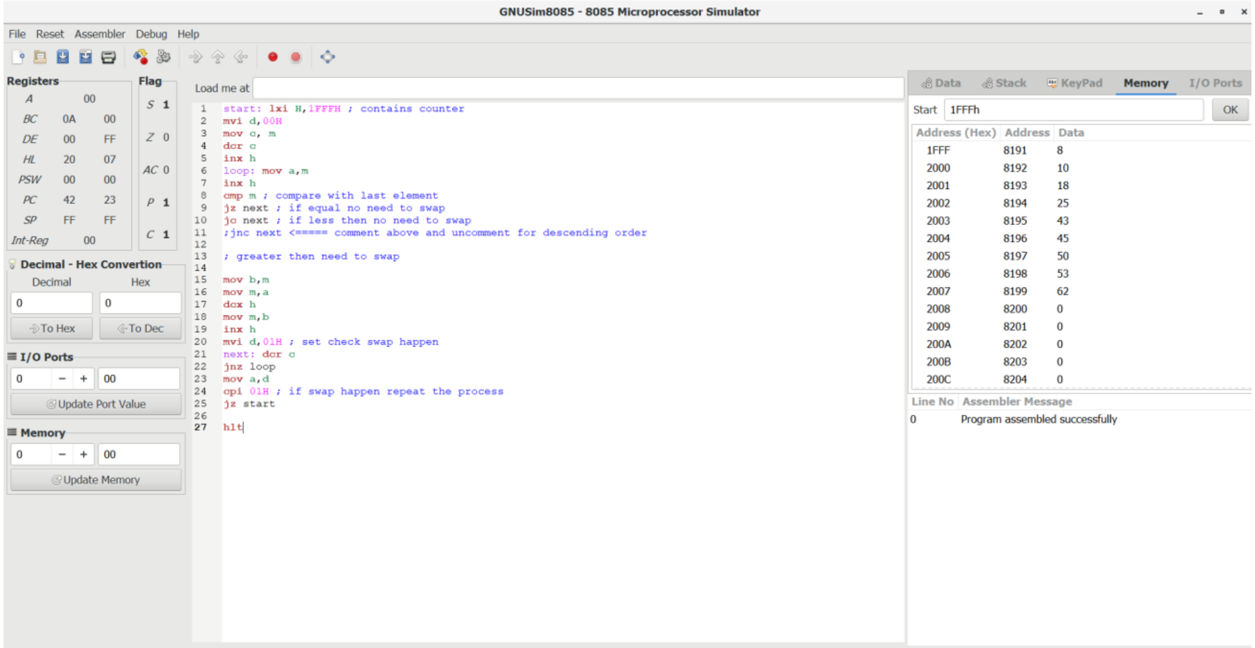
```
start: lxi H,1FFFH ; contains counter
mvi d,00H
mov c, m
dcr c
inx h
loop: mov a,m
inx h
cmp m ; compare with last element
jz next ; if equal no need to swap
jc next ; if less then no need to swap
;jnc next <===== comment above and uncomment for descending order

; greater then need to swap

mov b,m
mov m,a
dcx h
mov m,b
inx h
mvi d,01H ; set check swap happen
next: dcr c
jnz loop
mov a,d
cpi 01H ; if swap happen repeat the process
jz start

hlt
```

Start 1FFFh

Address (Hex)	Address	Data
1FFF	8191	8
2000	8192	43
2001	8193	45
2002	8194	25
2003	8195	50
2004	8196	18
2005	8197	62
2006	8198	53
2007	8199	10
2008	8200	0
2009	8201	0
200A	8202	0
200B	8203	0
200C	8204	0

Line No	Assembler Message
0	Program assembled successfully

After Sort



Registers		
A	00	
BC	0A	00
DE	00	FF
HL	20	07
PSW	00	00
PC	42	23
SP	FF	FF
Int-Reg	00	

Flag	
S	1
Z	0
AC	0
P	1
C	1

Decimal: 0 Hex: 0

I/O Ports: 0 − + 00

Memory: 0 − + 00

Load me at

```
start: lxi H,1FFFH ; contains counter
mvi d,00H
mov c, m
dcr c
inx h
loop: mov a,m
inx h
cmp m ; compare with last element
jz next ; if equal no need to swap
jc next ; if less then no need to swap
;jnc next <===== comment above and uncomment for descending order

; greater then need to swap

mov b,m
mov m,a
dcx h
mov m,b
inx h
mvi d,01H ; set check swap happen
next: dcr c
jnz loop
mov a,d
cpi 01H ; if swap happen repeat the process
jz start

hlt
```

Start 1FFFh

Address (Hex)	Address	Data
1FFF	8191	8
2000	8192	10
2001	8193	18
2002	8194	25
2003	8195	43
2004	8196	45
2005	8197	50
2006	8198	53
2007	8199	62
2008	8200	0
2009	8201	0
200A	8202	0
200B	8203	0
200C	8204	0

Line No	Assembler Message
0	Program assembled successfully

5-> Write a Program to find Largest number in a given data array.

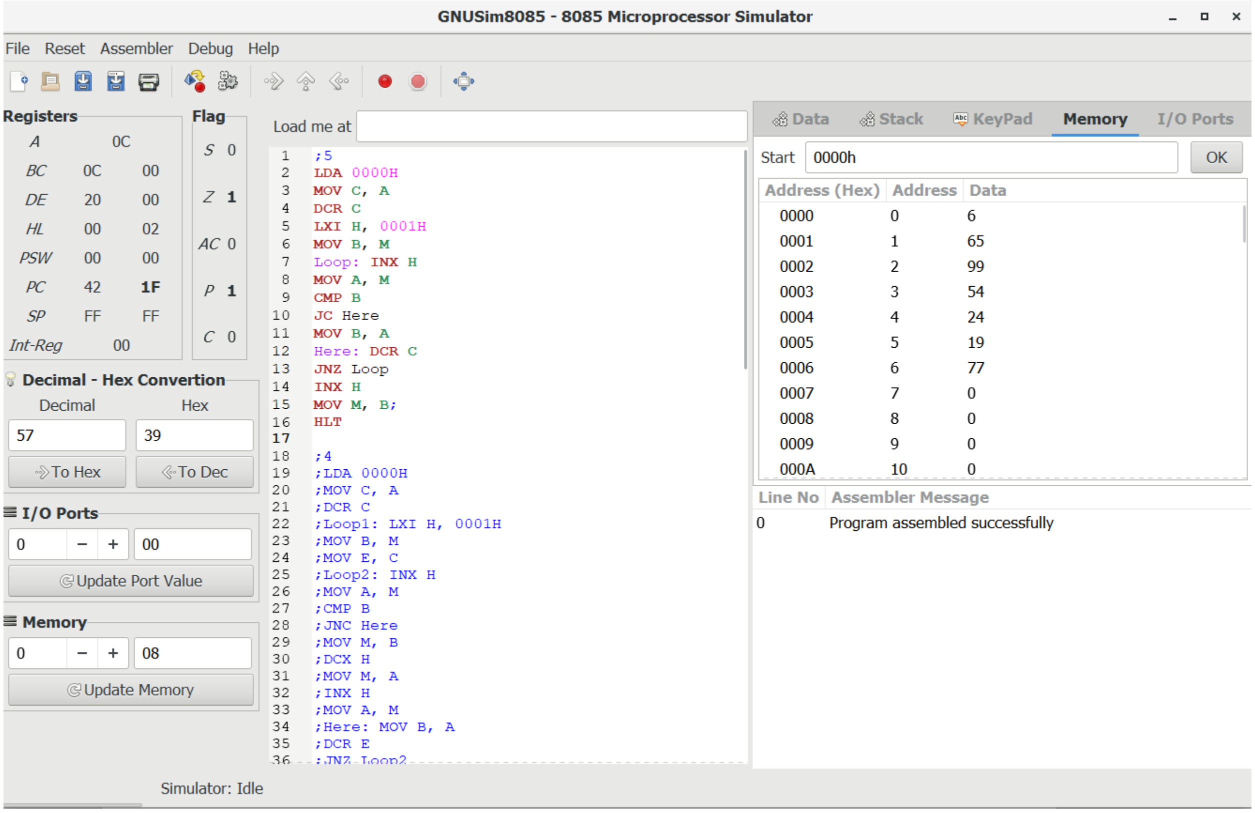
Code:-

LDA 0000H

MOV C, A

DCR C

```
LXI H, 0001H
MOV B, M
Loop: INX H
MOV A, M
CMP B
JC Here
MOV B, A
Here: DCR C
JNZ Loop
INX H
MOV M, B;
HLT
```



GNUSim8085 - 8085 Microprocessor Simulator

Registers

A	0C	
BC	0C	00
DE	20	00
HL	00	02
PSW	00	00
PC	42	1F
SP	FF	FF
Int-Reg	00	

Flag

S	0
Z	1
AC	0
P	1
C	0

Decimal - Hex Convertion

Decimal: 57 Hex: 39

I/O Ports

0 − + 00

Memory

0 − + 08

Load me at

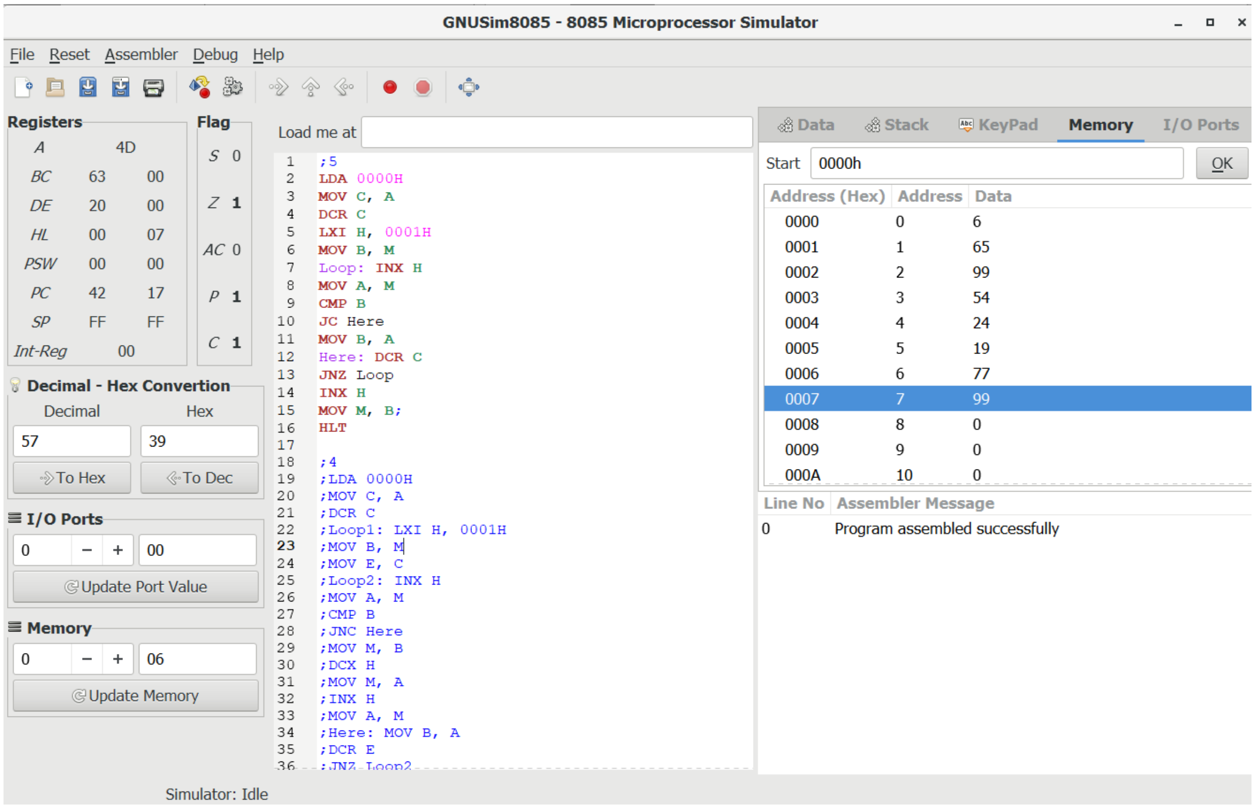
```
;5
LDA 0000H
MOV C, A
DCR C
LXI H, 0001H
MOV B, M
Loop: INX H
MOV A, M
CMP B
JC Here
MOV B, A
Here: DCR C
JNZ Loop
INX H
MOV M, B;
HLT

;4
;LDA 0000H
;MOV C, A
;DCR C
;Loop1: LXI H, 0001H
;MOV B, M
;MOV E, C
;Loop2: INX H
;MOV A, M
;CMP B
;JNC Here
;MOV M, B
;DCX H
;MOV M, A
;INX H
;MOV A, M
;Here: MOV B, A
;DCR E
;JNZ Loop2
```

Start 0000h

Address (Hex)	Address	Data
0000	0	6
0001	1	65
0002	2	99
0003	3	54
0004	4	24
0005	5	19
0006	6	77
0007	7	0
0008	8	0
0009	9	0
000A	10	0

Line No	Assembler Message
0	Program assembled successfully

Simulator: Idle



Registers		
A	4D	
BC	63	00
DE	20	00
HL	00	07
PSW	00	00
PC	42	17
SP	FF	FF
Int-Reg	00	

Flag	
S	0
Z	1
AC	0
P	1
C	1

Decimal	Hex
57	39

I/O Ports: 0 – + 00

Memory: 0 – + 06

```
;5
LDA 0000H
MOV C, A
DCR C
LXI H, 0001H
MOV B, M
Loop: INX H
MOV A, M
CMP B
JC Here
MOV B, A
Here: DCR C
JNZ Loop
INX H
MOV M, B;
HLT

;4
;LDA 0000H
;MOV C, A
;DCR C
;Loop1: LXI H, 0001H
;MOV B, M
;MOV E, C
;Loop2: INX H
;MOV A, M
;CMP B
;JNC Here
;MOV M, B
;DCX H
;MOV M, A
;INX H
;MOV A, M
;Here: MOV B, A
;DCR E
;JNZ Loop2
```

Start 0000h

Address (Hex)	Address	Data
0000	0	6
0001	1	65
0002	2	99
0003	3	54
0004	4	24
0005	5	19
0006	6	77
0007	7	99
0008	8	0
0009	9	0
000A	10	0

Line No	Assembler Message
0	Program assembled successfully

6-> Write a Program to move a block starting at location 2000H To location 3000H with overlap/without overlap.

Code:-

lxi h,1FFFH

mov c,m ;store block size

inx h ; first data byte of the source

lxi d,3000H ; destination address

mvi b,00H

dad b ; HL <- HL + BC(counter)

dcx h ; Point to the last of source

xchg

dad b

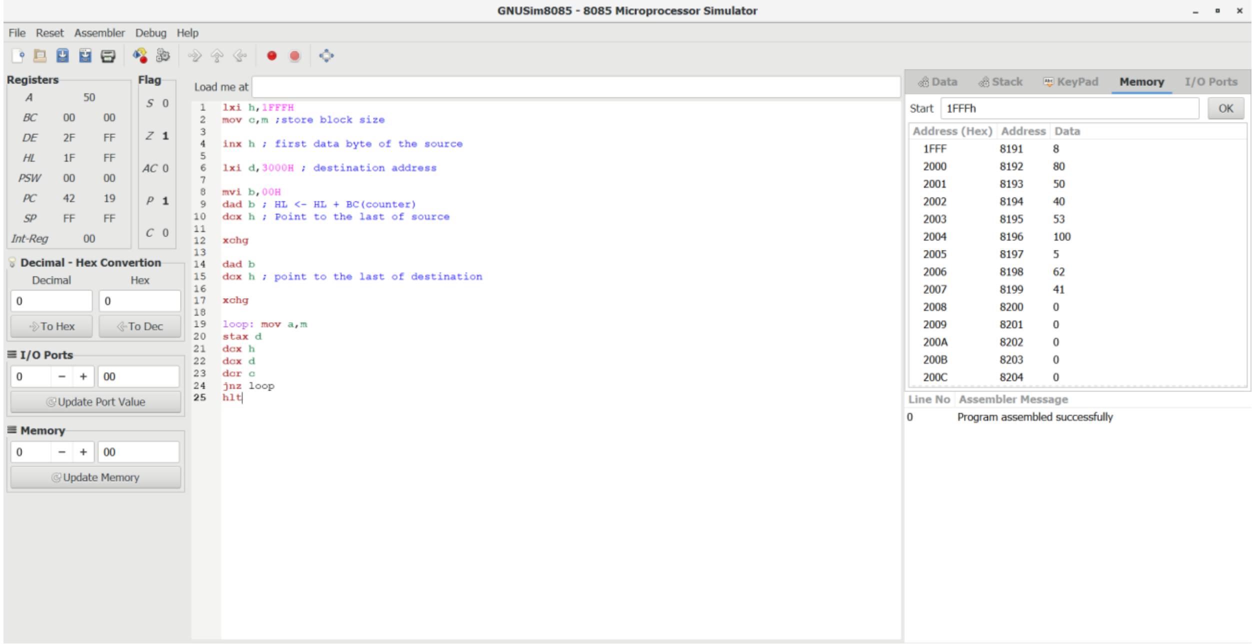
dcx h ; point to the last of destination

xchg

loop: mov a,m

stax d

```
dcx h
dcx d
dcr c
jnz loop
hlt
```


GNUSim8085 - 8085 Microprocessor Simulator
File Reset Assembler Debug Help
Data Stack KeyPad Memory I/O Ports
Decimal - Hex Convertion
To Hex
To Dec
I/O Ports
Update Port Value
Memory
Update Memory

Registers

A	50	
BC	00	00
DE	2F	FF
HL	1F	FF
PSW	00	00
PC	42	19
SP	FF	FF
Int-Reg	00	

Flag

S	0
Z	1
AC	0
P	1
C	0

Decimal 0 Hex 0

I/O Ports 0 − + 00

Memory 0 − + 00

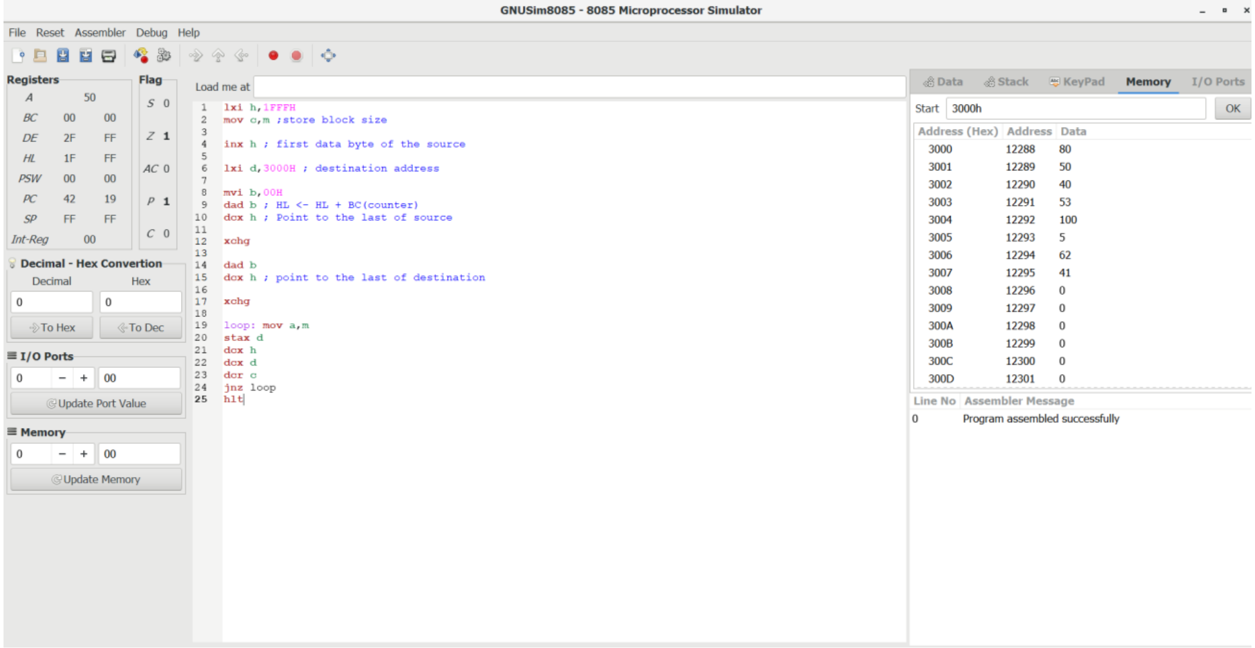
Load me at

```
lxi h,1FFFH
mov c,m ;store block size

inx h ; first data byte of the source

lxi d,3000H ; destination address

mvi b,00H
dad b ; HL <- HL + BC(counter)
dcx h ; Point to the last of source

xchg

dad b
dcx h ; point to the last of destination

xchg

loop: mov a,m
stax d
dcx h
dcx d
dcr c
jnz loop
hlt
```

Start 1FFFh OK

Address (Hex)	Address	Data
1FFF	8191	8
2000	8192	80
2001	8193	50
2002	8194	40
2003	8195	53
2004	8196	100
2005	8197	5
2006	8198	62
2007	8199	41
2008	8200	0
2009	8201	0
200A	8202	0
200B	8203	0
200C	8204	0

Line No	Assembler Message
0	Program assembled successfully


GNUSim8085 - 8085 Microprocessor Simulator
File Reset Assembler Debug Help
Data Stack KeyPad Memory I/O Ports
Decimal - Hex Convertion
To Hex
To Dec
I/O Ports
Update Port Value
Memory
Update Memory

Registers

A	50	
BC	00	00
DE	2F	FF
HL	1F	FF
PSW	00	00
PC	42	19
SP	FF	FF
Int-Reg	00	

Flag

S	0
Z	1
AC	0
P	1
C	0

Decimal 0 Hex 0

I/O Ports 0 − + 00

Memory 0 − + 00

Load me at

```
lxi h,1FFFH
mov c,m ;store block size

inx h ; first data byte of the source

lxi d,3000H ; destination address

mvi b,00H
dad b ; HL <- HL + BC(counter)
dcx h ; Point to the last of source

xchg

dad b
dcx h ; point to the last of destination

xchg

loop: mov a,m
stax d
dcx h
dcx d
dcr c
jnz loop
hlt
```

Start 3000h OK

Address (Hex)	Address	Data
3000	12288	80
3001	12289	50
3002	12290	40
3003	12291	53
3004	12292	100
3005	12293	5
3006	12294	62
3007	12295	41
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