U18CO018
Shubham Shekhaliya
Assignment – 6
MIT

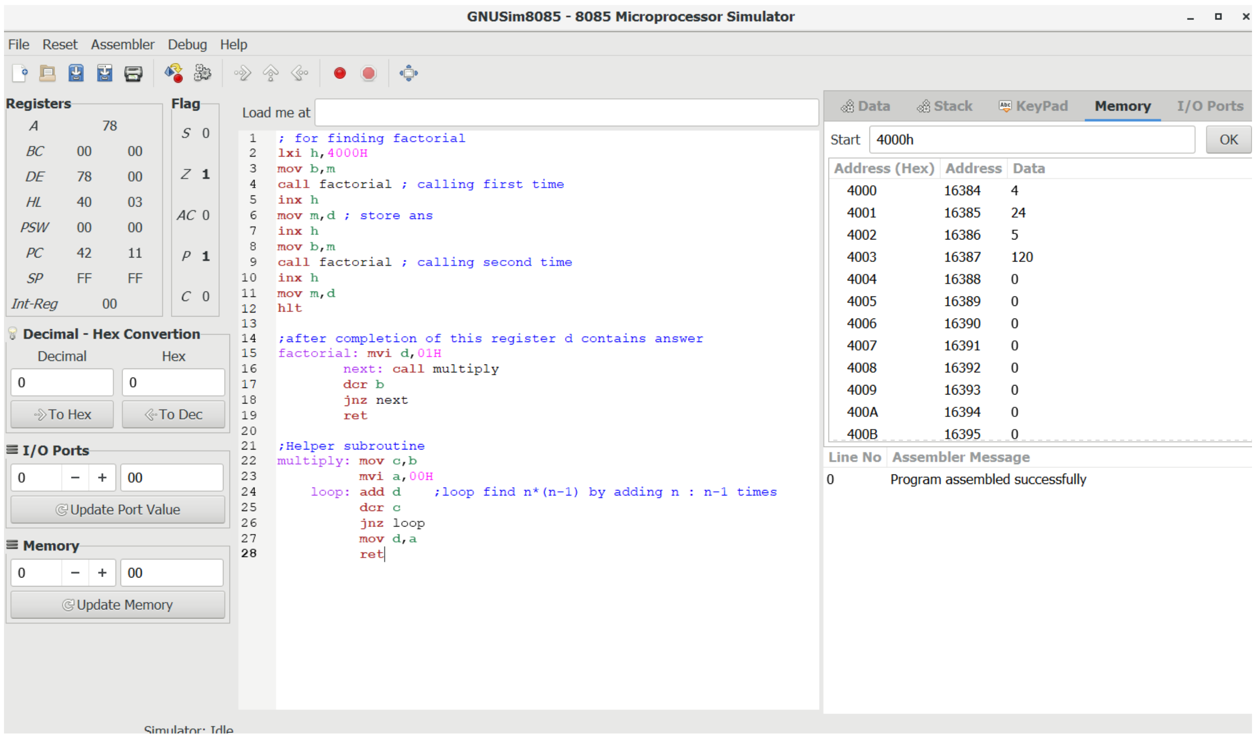
1-> WAP to find Factorial of a given number using Call and Subroutine.

Code :-

```
; for finding factorial
lxi h,4000H
mov b,m
call factorial ; calling first time
inx h
mov m,d ; store ans
inx h
mov b,m
call factorial ; calling second time
inx h
mov m,d
hlt

;after completion of this register d contains answer
factorial: mvi d,01H
     next: call multiply
     dcr b
     jnz next
     ret

;Helper subroutine
multiply: mov c,b
      mvi a,00H
  loop: add d    ;loop find n*(n-1) by adding n : n-1 times
      dcr c
      jnz loop
      mov d,a
      ret
```



GNUSim8085 - 8085 Microprocessor Simulator

Registers		
A	78	
BC	00	00
DE	78	00
HL	40	03
PSW	00	00
PC	42	11
SP	FF	FF
Int-Reg	00	

Flag	
S	0
Z	1
AC	0
P	1
C	0

```
; for finding factorial
lxi h,4000H
mov b,m
call factorial ; calling first time
inx h
mov m,d ; store ans
inx h
mov b,m
call factorial ; calling second time
inx h
mov m,d
hlt

;after completion of this register d contains answer
factorial: mvi d,01H
        next: call multiply
        dcr b
        jnz next
        ret

;Helper subroutine
multiply: mov c,b
          mvi a,00H
    loop: add d    ;loop find n*(n-1) by adding n : n-1 times
          dcr c
          jnz loop
          mov d,a
          ret
```

Start 4000h

Address (Hex)	Address	Data
4000	16384	4
4001	16385	24
4002	16386	5
4003	16387	120
4004	16388	0
4005	16389	0
4006	16390	0
4007	16391	0
4008	16392	0
4009	16393	0
400A	16394	0
400B	16395	0

Line No	Assembler Message
0	Program assembled successfully

Simulator: Idle

2-> WAP for Fibonacci Series using Call and Subroutine.

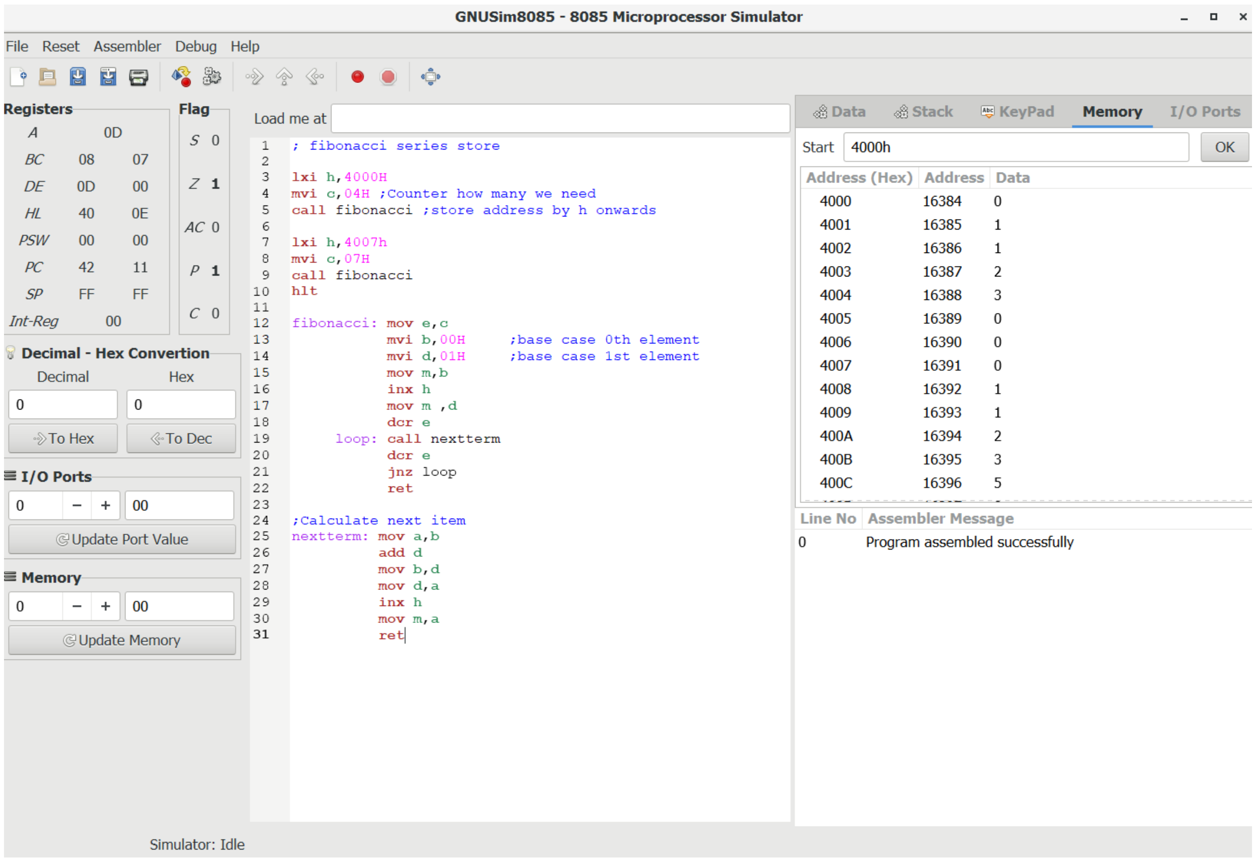
Code : -

```
; fibonacci series store
lxi h,4000H
mvi c,04H           ;Counter how many we need
call fibonacci      ;store address by h onwards

lxi h,4007h
mvi c,07H
call fibonacci
hlt
fibonacci: mov e,c
           mvi b,00H    ;base case 0th element
           mvi d,01H    ;base case 1st element
           mov m,b
           inx h
           mov m ,d
           dcr e
    loop: call nextterm
           dcr e
```

```
                jnz loop
                ret

;Calculate next item
nextterm: mov a,b
               add d
               mov b,d
               mov d,a
               inx h
               mov m,a
               ret
```



Registers

A	0D	
BC	08	07
DE	0D	00
HL	40	0E
PSW	00	00
PC	42	11
SP	FF	FF
Int-Reg	00	

Flag

S	0
Z	1
AC	0
P	1
C	0

Decimal - Hex Convertion

Decimal: 0 Hex: 0

I/O Ports: 0 00

Memory: 0 00

Load me at

```
; fibonacci series store

lxi h,4000H
mvi c,04H ;Counter how many we need
call fibonacci ;store address by h onwards

lxi h,4007h
mvi c,07H
call fibonacci
hlt

fibonacci: mov e,c
           mvi b,00H     ;base case 0th element
           mvi d,01H     ;base case 1st element
           mov m,b
           inx h
           mov m ,d
           dcr e
     loop: call nextterm
           dcr e
           jnz loop
           ret

;Calculate next item
nextterm: mov a,b
          add d
          mov b,d
          mov d,a
          inx h
          mov m,a
          ret
```

Start 4000h

Address (Hex)	Address	Data
4000	16384	0
4001	16385	1
4002	16386	1
4003	16387	2
4004	16388	3
4005	16389	0
4006	16390	0
4007	16391	0
4008	16392	1
4009	16393	1
400A	16394	2
400B	16395	3
400C	16396	5

Line No	Assembler Message
0	Program assembled successfully

Simulator: Idle

3-> WAP to find Multiplication of Two 8-Bit Numbers using Call and Subroutine.

Code : -

```
lxi h,2000h
mov b,m       ;get data
inx h
mov c,m      ; get data
```

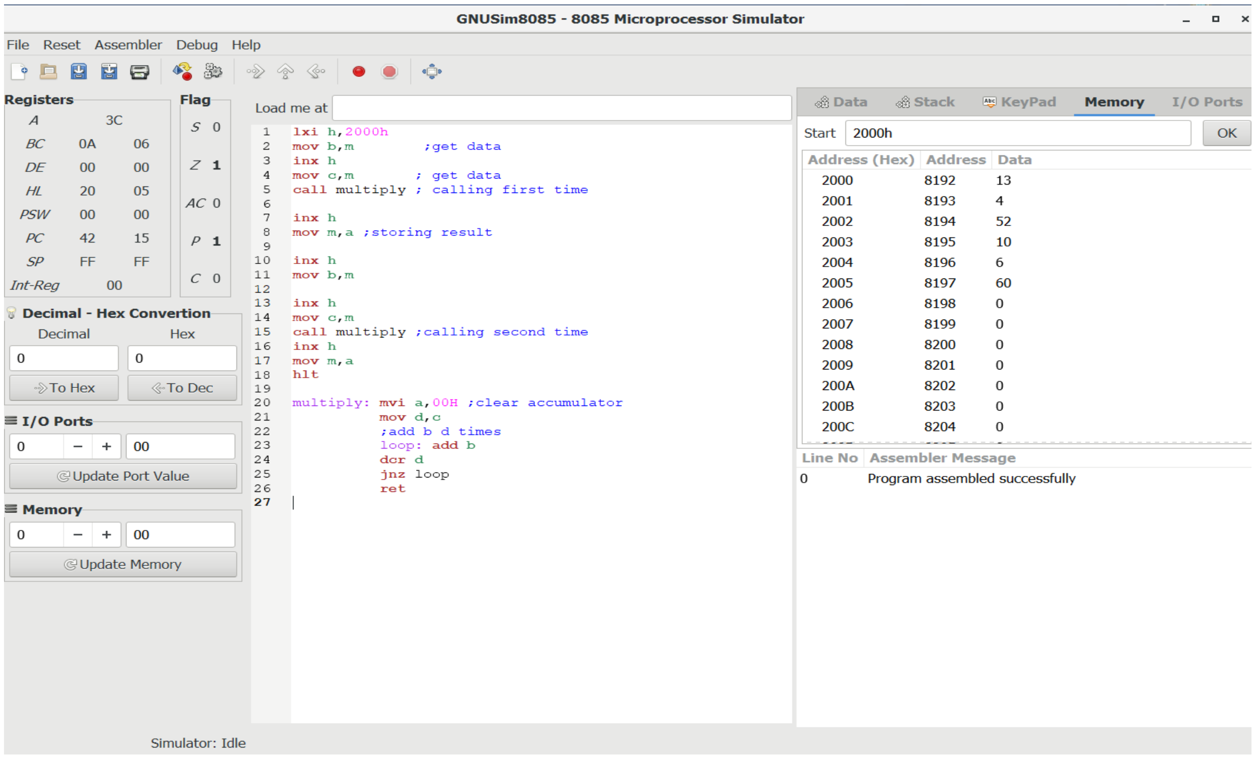
```
call multiply ; calling first time

inx h
mov m,a ;storing result

inx h
mov b,m

inx h
mov c,m
call multiply ;calling second time
inx h
mov m,a
hlt

multiply: mvi a,00H      ;clear accumulator
          mov d,c        ;add b d times
    loop: add b
          dcr d
          jnz loop
          ret
```



Registers

A	3C	
BC	0A	06
DE	00	00
HL	20	05
PSW	00	00
PC	42	15
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,2000h
mov b,m         ;get data
inx h
mov c,m        ; get data
call multiply ; calling first time

inx h
mov m,a ;storing result

inx h
mov b,m

inx h
mov c,m
call multiply ;calling second time
inx h
mov m,a
hlt

multiply: mvi a,00H ;clear accumulator
          mov d,c
          ;add b d times
          loop: add b
          dcr d
          jnz loop
          ret
```

Start 2000h

Address (Hex)	Address	Data
2000	8192	13
2001	8193	4
2002	8194	52
2003	8195	10
2004	8196	6
2005	8197	60
2006	8198	0
2007	8199	0
2008	8200	0
2009	8201	0
200A	8202	0
200B	8203	0
200C	8204	0

Line No	Assembler Message
0	Program assembled successfully

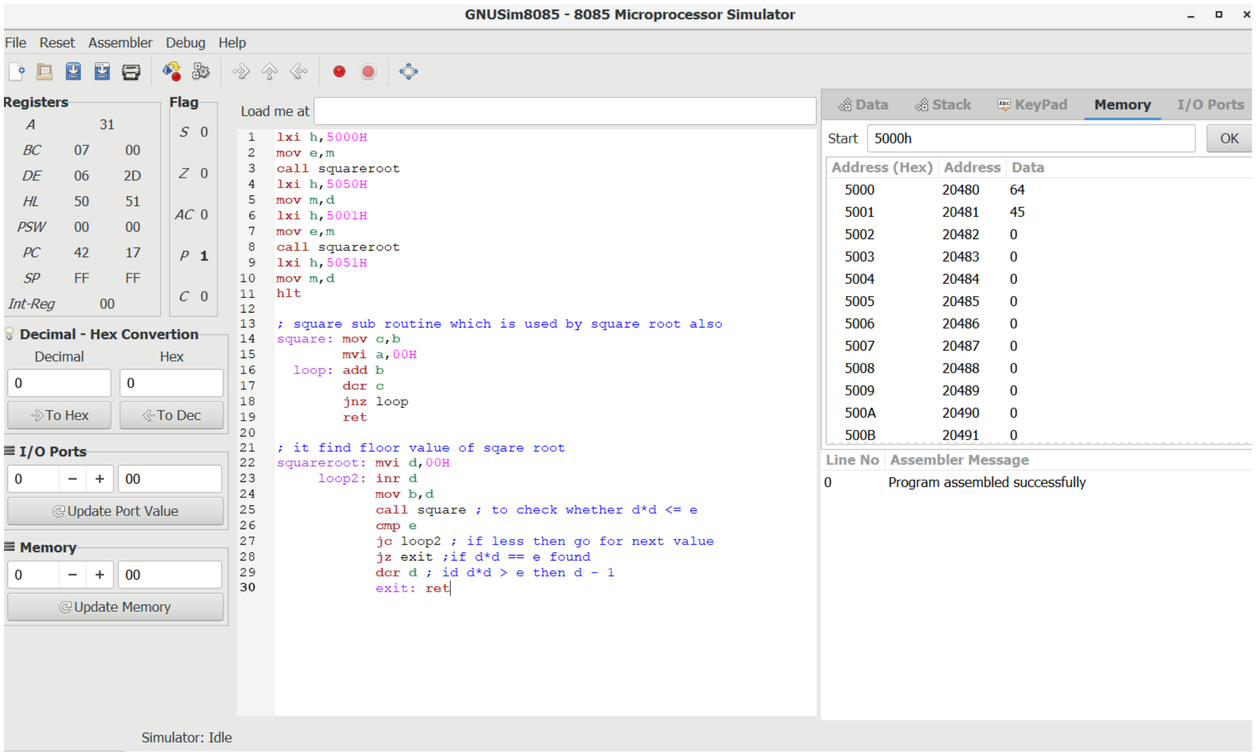
4-> Write Assembly language program to find the square/square root of a number .The number is stored at location 5000H, store the result at 5050H.
Code :-

```
lxi h,5000H
mov e,m
call squareroot
lxi h,5050H
mov m,d
lxi h,5001H
mov e,m
call squareroot
lxi h,5051H
mov m,d
hlt

; square sub routine which is used by square root also
square: mov c,b
        mvi a,00H
  loop: add b
        dcr c
        jnz loop
        ret

; it find floor value of sqare root
squareroot: mvi d,00H
   loop2: inr d
          mov b,d
          call square        ; to check whether d*d <= e
          cmp e
          jc loop2           ; if less then go for next value
          jz exit            ;if d*d == e found
          dcr d              ; id d*d > e then d - 1
          exit: ret
```

Before : -


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
OK

Registers

A	31	
BC	07	00
DE	06	2D
HL	50	51
PSW	00	00
PC	42	17
SP	FF	FF
Int-Reg	00	

Flag

S	0
Z	0
AC	0
P	1
C	0

Load me at

```
lxi h,5000H
mov e,m
call squareroot
lxi h,5050H
mov m,d
lxi h,5001H
mov e,m
call squareroot
lxi h,5051H
mov m,d
hlt

; square sub routine which is used by square root also
square: mov c,b
        mvi a,00H
  loop: add b
        dcr c
        jnz loop
        ret

; it find floor value of sqare root
squareroot: mvi d,00H
     loop2: inr d
            mov b,d
            call square ; to check whether d*d <= e
            cmp e
            jc loop2 ; if less then go for next value
            jz exit ;if d*d == e found
            dcr d ; id d*d > e then d - 1
            exit: ret
```

Start 5000h

Address (Hex)	Address	Data
5000	20480	64
5001	20481	45
5002	20482	0
5003	20483	0
5004	20484	0
5005	20485	0
5006	20486	0
5007	20487	0
5008	20488	0
5009	20489	0
500A	20490	0
500B	20491	0

Line No	Assembler Message
0	Program assembled successfully

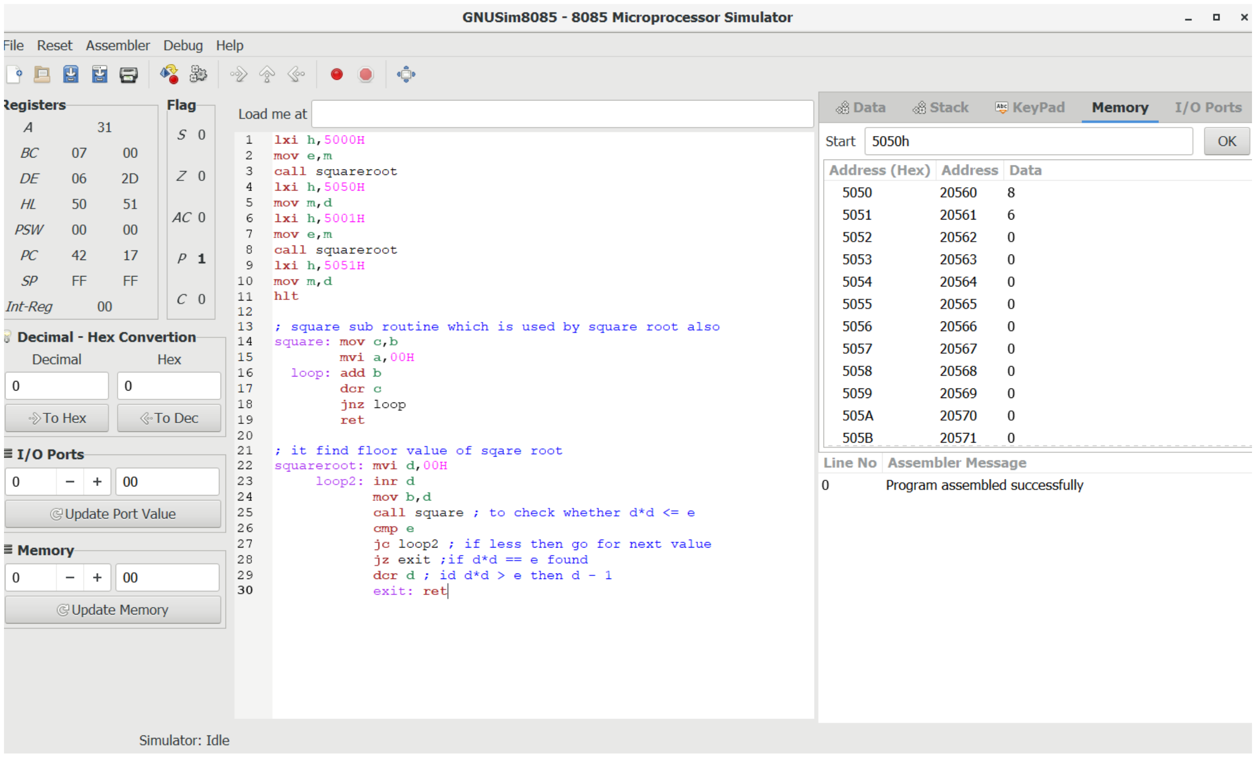
Decimal 0 Hex 0

I/O Ports 0 00

Memory 0 00

Simulator: Idle

After : -


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
OK

Registers

A	31	
BC	07	00
DE	06	2D
HL	50	51
PSW	00	00
PC	42	17
SP	FF	FF
Int-Reg	00	

Flag

S	0
Z	0
AC	0
P	1
C	0

Load me at

```
lxi h,5000H
mov e,m
call squareroot
lxi h,5050H
mov m,d
lxi h,5001H
mov e,m
call squareroot
lxi h,5051H
mov m,d
hlt

; square sub routine which is used by square root also
square: mov c,b
        mvi a,00H
  loop: add b
        dcr c
        jnz loop
        ret

; it find floor value of sqare root
squareroot: mvi d,00H
     loop2: inr d
            mov b,d
            call square ; to check whether d*d <= e
            cmp e
            jc loop2 ; if less then go for next value
            jz exit ;if d*d == e found
            dcr d ; id d*d > e then d - 1
            exit: ret
```

Start 5050h

Address (Hex)	Address	Data
5050	20560	8
5051	20561	6
5052	20562	0
5053	20563	0
5054	20564	0
5055	20565	0
5056	20566	0
5057	20567	0
5058	20568	0
5059	20569	0
505A	20570	0
505B	20571	0

Line No	Assembler Message
0	Program assembled successfully

Decimal 0 Hex 0

I/O Ports 0 00

Memory 0 00

Simulator: Idle