

# Experiment 3

Name:Pratik Chavan

Div/Batch:A/A1      Roll No.07

.MODEL SMALL

.STACK 100H

.DATA

NUM DB 5      ; Number for factorial (change this value as needed)

FACT DW 1      ; Variable to store factorial result

MSG DB 'Factorial: \$' ; Message to display before result

.CODE

MAIN PROC

MOV AX, @DATA

MOV DS, AX

MOV AL, NUM      ; Load number in AL

CBW              ; Convert AL to AX (sign-extend)

CALL FACTORIAL   ; Call factorial procedure

MOV DX, OFFSET MSG

MOV AH, 09H

INT 21H          ; Print message

CALL PRINT\_NUM   ; Print the factorial result

MOV AH, 4CH

```

    INT 21H      ; Exit program
MAIN ENDP

; Factorial Procedure
FACTORIAL PROC

    MOV CX, AX   ; Move number to CX for loop counter
    MOV AX, 1     ; Initialize AX = 1 (Factorial starts at 1)

FACTORIAL_LOOP:
    MUL CX       ; AX = AX * CX
    LOOP FACTORIAL_LOOP

    MOV FACT, AX ; Store the result in FACT
    RET

FACTORIAL ENDP

; Print Number Procedure
PRINT_NUM PROC

    MOV AX, FACT ; Load factorial result
    MOV CX, 0     ; Clear CX (digit counter)

NEXT_DIGIT:
    MOV DX, 0
    MOV BX, 10
    DIV BX        ; AX / 10 → Quotient in AX, Remainder in DX
    PUSH DX       ; Push remainder (digit) onto stack
    INC CX        ; Increment digit counter
    TEST AX, AX   ; Check if AX is zero

```

JNZ NEXT\_DIGIT ; If not, continue extracting digits

PRINT\_LOOP:

POP DX ; Get digit from stack

ADD DL, '0' ; Convert to ASCII

MOV AH, 02H

INT 21H ; Print digit

LOOP PRINT\_LOOP ; Repeat for remaining digits

RET

PRINT\_NUM ENDP

END MAIN

OUTPUT:

The screenshot shows a DOS debugger window with the following content:

CPU 80486		3=1711	
#fact#main		ax 0078	c=0
cs:0000 B88108	♦ MOV AX, DGROUP	bx 0000	z=0
cs:0003 8ED8	♦ MOV DS, AX	cx 0000	s=0
cs:0005 A00000	♦ MOV AL, NUM ; Load num	dx 0000	o=0
cs:0008 98	♦ CBW ; Convert AL to AX	si 0000	p=1
cs:0009 E80E00	♦ CALL FACTORIAL ; Call	di 0000	a=0
cs:000C BA0300	♦ MOV DX, OFFSET MSG	bp 0000	i=1
cs:000F B409	♦ MOV AH, 09H	sp 0100	d=0
cs:0011 CD21	♦ INT 21H ; Print message	ds 0881	
cs:0013 E81100	♦ CALL PRINT_NUM ; Print	es 086C	
cs:0016 B44C	♦ MOV AH, 4CH	ss 0882	
cs:0018 CD21	♦ INT 21H ; Exit program	cs 087C	
#fact#factorial		ip 000C	
es:0000 CD 20 7D 9D 00 EA FF FF = }¥ R			
es:0008 AD DE 32 0B C5 05 6B 07 i  2d *k.			
es:0010 15 03 28 08 15 03 93 01 \$*(S*6@			
es:0018 01 01 01 00 02 04 FF FF @@@ @.			
		ss:0102 0403	
		ss:0100 52FB	

# Experiment 3

Name:Pratik Chavan

Div/Batch:A/A1      Roll No.07

.MODEL SMALL

.STACK 100H

.DATA

    ARRAY DB 10, 25, 15, 40, 5, 30, 50, 20

    LEN EQU \$ - ARRAY

    MIN DB 0

    MAX DB 0

.CODE

MAIN PROC

    MOV AX, @DATA

    MOV DS, AX

    MOV SI, 0

    MOV AL, ARRAY[SI]

    MOV MIN, AL

    MOV MAX, AL

FIND\_MIN\_MAX:

    MOV AL, ARRAY[SI]

    CMP AL, MAX

    JG UPDATE\_MAX

    CMP AL, MIN

JL UPDATE\_MIN

JMP NEXT\_ELEMENT

UPDATE\_MAX:

MOV MAX, AL

JMP NEXT\_ELEMENT

UPDATE\_MIN:

MOV MIN, AL

NEXT\_ELEMENT:

INC SI

CMP SI, LEN

JL FIND\_MIN\_MAX

MOV AX, 4C00H

INT 21H

MAIN ENDP

END MAIN

OUTPUT:

The screenshot shows a debugger window with the following content:

Address	Disassembly	Comment	Register/Value
cs:000D	8A840000	+ mov al, [array + si]	ax 0C07
cs:0011	3AC4	+ cmp al, ah ; Compare c	bx 0000
cs:0013	7F15	+ jg update_max ; If AL	cx 0000
cs:0015	3AC0	+ cmp al, al ; Compare c	dx 0000
cs:0017	7C0D	+ jl update_min ; If AL	si 0006
cs:0019	46	+ inc si	di 0000
cs:001A	A00700	+ mov al, [n] ; Load arr	bp 0000
cs:001D	3BF0	+ cmp si, ax ; Compare i	sp 0100
cs:001F	7CEC	+ jl find_min_max ; If s	ds 0B7F
cs:0021	B8004C	+ mov ax, 4C00h	es 0B6C
cs:0024	CD21	+ int 21h	ss 0B80
#max#update_min			cs 0B7C
			ip 001F

Registers (right side):

c=1
z=0
s=1
o=0
p=1
a=1
i=1
d=0

Stack (bottom):

ss:0102 0403
ss:0100 52FB