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
```

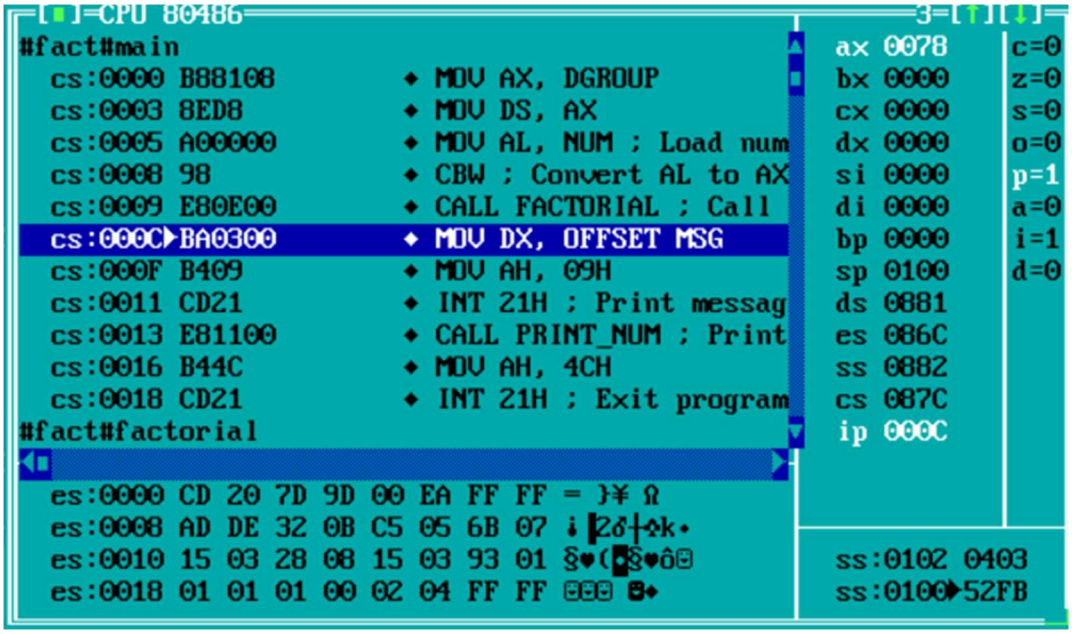
```
    JNZ NEXT_DIGIT   ; If not, continue extracting digits

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:



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: