Green University of Bangladesh
Department of Computer Science and Engineering (CSE)

Faculty of Sciences and Engineering
Semester: (Fall, Year:2025), B.Sc. in CSE (Day)

Lab Report NO: 03
Course Title: Microprocessors, Microcontrollers, and Embedded System
Course Code: CSE 304 **Section:** 232-D1

Lab Experiment Name: Implementation of loop using assembly language.

Student Details

	Name	ID
1.	Rukonuzzaman Topu	232002280

Submission Date : 05/11/2025

Course Teacher's Name : Jarin Tasnim Tonvi

[For Teachers use only: Don't Write Anything inside this box]

Lab Report Status

Marks: …… **Signature:.....**
Comments:..... **Date:.....**

1. INTRODUCTION

This experiment demonstrates the use of looping, arithmetic operations, and user interaction in assembly language programming using the 8086 microprocessor. Two different problems are solved to understand the basic logic-building and control flow in low-level programming.

In the first program, the factorial of a given number is calculated. The user enters a single-digit number, and the program multiplies it by all smaller positive integers using a loop structure. The result is then displayed on the screen. This task helps to understand register operations, MUL instruction, and loop control using CX register.

In the second program, the summation of squares ($1^2 + 2^2 + 3^2 + \ldots + n^2$) is performed. The program takes an input from the user, squares each number within the range, and accumulates the result. It demonstrates the use of arithmetic instructions (MUL, ADD), conditional jumps (JBE), and data handling in memory and registers.

Both programs utilize DOS interrupt 21H for input and output operations, showing how system calls are used in assembly to interact with the user. These exercises build a foundation for understanding loops, counters, and arithmetic computation in assembly language.

2. OBJECTIVES

1. To learn how to use loops (LOOP, CMP, and conditional jumps) for repeated arithmetic operations.
2. To implement mathematical operations such as factorial and summation of squares using the MUL and ADD instructions.
3. To practice using registers (AX, BX, CX, DX) and memory variables for storing and processing data.
4. To develop logical thinking and problem-solving skills through low-level algorithm design.

3. PROCEDURE

Problem-1:Take a number n from user. After that find out the factorial of that number n.

1. Initialize the data segment using MOV AX, @DATA and MOV DS, AX.
2. Display the message “Enter the number:” using interrupt INT 21H.
3. Take a single-digit input from the user and convert it from ASCII to numeric form.
4. Set RESULT = 1 and load the input value into CX as the loop counter.
5. Use a loop to multiply RESULT by CX in each iteration until CX becomes zero.
6. Store the final factorial value in memory and display it using interrupt INT 21H.
7. Terminate the program using MOV AH,4CH and INT 21H.

Problem-2: Implement a loop to find out the summation of 1^2 + 2^2 + 3^2 + ... + n^2.

1. Initialize the data segment using MOV AX, @DATA and MOV DS, AX.
2. Display the message “Enter number:” and take a single-digit input using INT 21H.
3. Convert the ASCII input to numeric by subtracting 30H and store it in variable n.
4. Set SUM = 0 and start a loop with counter i = 1.

5. In each loop, multiply i * i using the MUL instruction and add it to SUM.
6. Continue looping until i becomes equal to n.
7. Display the message “Sum of squares is:” followed by the result using INT 21H.
8. Exit the program using MOV AH,4CH and INT 21H.

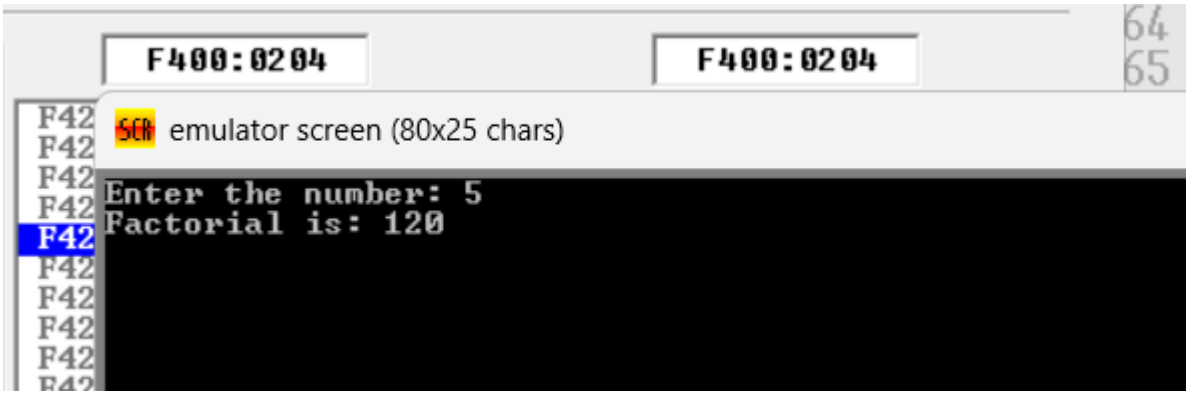
4. IMPLEMENTATION

Source Code:Take a number n from user. After that find out the factorial of that number n.

```
.MODEL SMALL
.STACK 100H
.DATA
N DB ?
MSG1 DB "Enter the number: $"
MSG2 DB 0DH,0AH,"Factorial is: $"
RESULT DW 1

.CODE
MAIN PROC
   MOV AX,@DATA
   MOV DS,AX

   ; Prompt user
   MOV AH,9
   LEA DX,MSG1
   INT 21H

   ; Take input
   MOV AH,1
   INT 21H
   SUB AL,30H
   MOV N,AL

   ; RESULT = 1
   MOV AX,1
   MOV RESULT,AX

   ; CX = N
   MOV CL,N
   MOV CH,0

FACTORIAL_LOOP:
   CMP CX,0
   JE PRINT_RESULT
   MOV AX,RESULT
   MUL CX
   MOV RESULT,AX
   LOOP FACTORIAL_LOOP
```

```
PRINT_RESULT:
   MOV AH,9
   LEA DX,MSG2
   INT 21H

   MOV AX,RESULT
   MOV BX,10
   MOV CX,0
STORE_DIGIT:
   MOV DX,0
   DIV BX
   PUSH DX
   INC CX
   CMP AX,0
   JNE STORE_DIGIT

PRINT_DIGITS:
   POP DX
   ADD DL,30H
   MOV AH,2
   INT 21H
   LOOP PRINT_DIGITS

   MOV AH,4CH
   INT 21H
MAIN ENDP
END MAIN
```

Output :



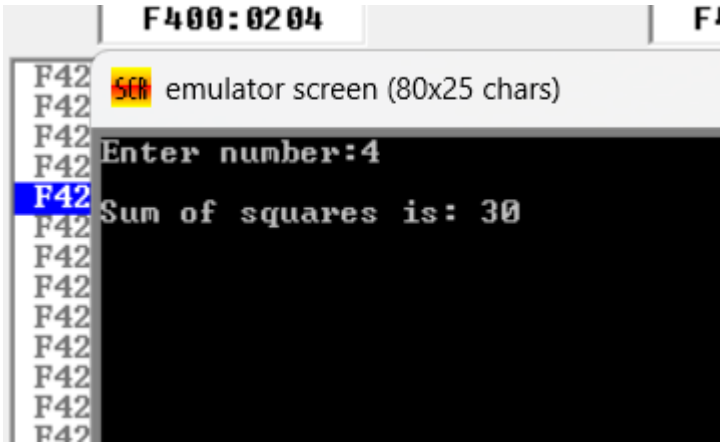
Source Code: Implement a loop to find out the summation of 1^2 + 2^2 + 3^2 + ... + n^2. You can take n from user as an input.

```
.MODEL SMALL
```

```
.STACK 100H
.DATA

msg1 DB 'Enter number:$'
msg2 DB 0DH,0AH,'Sum of squares is: $'
n DB ?
sum DW 0

.CODE
MAIN PROC
    MOV AX, @DATA
    MOV DS, AX

    ; Show message
    MOV AH, 9
    LEA DX, msg1
    INT 21H

    ; Take user input (single digit)
    MOV AH, 1
    INT 21H
    SUB AL, 30H       ; convert ASCII to number
    MOV n, AL

    ; Initialize
    MOV CX, 1         ; counter = 1
    MOV SUM, 0

loop_start:
    MOV AX, CX
    MUL CX            ; AX = CX * CX  (square)
    ADD SUM, AX       ; SUM = SUM + i^2
    INC CX
    MOV AL, n
    CMP CL, AL
    JBE loop_start    ; if i <= n, continue loop

    ; New line
    MOV AH,2
    MOV DL,10
    INT 21H
    MOV DL,13
    INT 21H

    ; Show message
    MOV AH,9
    LEA DX,msg2
    INT 21H
```

```
    ; Print result (two digits)
    MOV AX, SUM
    MOV BL,10
    DIV BL
    MOV BH,AL
    MOV BL,AH

    ADD BH,30H
    ADD BL,30H

    MOV DL,BH
    MOV AH,2
    INT 21H

    MOV DL,BL
    MOV AH,2
    INT 21H

    ; Exit
    MOV AH,4CH
    INT 21H

MAIN ENDP
END MAIN
```

Output:



DISCUSSION

In this experiment, two assembly language programs were written to perform mathematical operations using looping and arithmetic instructions in the 8086 microprocessor. The first program calculated the factorial of a given number, while the second program found the summation of squares from 1^2 up to n^2. Both programs helped to understand the practical use of loops, arithmetic operations, and input-output handling through system interrupts.

In the factorial program, the number entered by the user was multiplied repeatedly by all smaller positive integers using a loop until it reached one. The MUL instruction was used for multiplication, and the result was stored in the RESULT variable. This process demonstrated how the CX register could serve as a loop counter and how data is processed step by step in low-level operations.

In the summation of squares program, the logic involved squaring each number in sequence using the MUL instruction and adding the result to an accumulator variable named SUM. The loop continued until the counter became equal to the input value. This program illustrated the concept of iteration, conditional checking, and accumulation in assembly language.

Overall, both programs successfully applied the concepts of looping, arithmetic operations, and interrupt-driven input/output. They provided a strong foundation for understanding how mathematical logic is implemented at the machine level, enhancing the ability to write more complex assembly programs in the future.