



# UITs

UNIVERSITY OF INFORMATION  
TECHNOLOGY AND SCIENCES

Assignment on

## **Lab Report- 04**

Course Title

**Microprocessor and MicroControllers**

Course Code

**CSE 360**

Submitted by

**KM Jakaria**

**Section: 6A**

**Batch: 53**

**ID: 0432310005101037**

Submitted to

**Md. Ismail**

**Lecturer**

**Department of CSE**

**UITs**

Date of Submission

**03 Sept, 2025**

## Problem No: 01

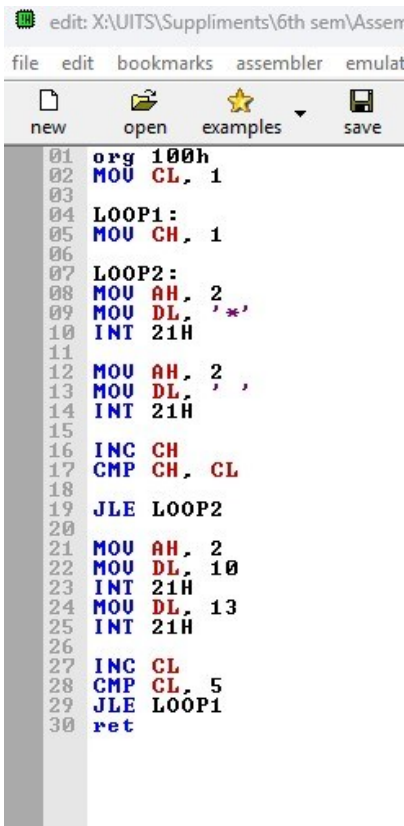
## Experiment No: 01

**Experiment Name:** Write an assembly program to print right angle triangle using star (\*)

### Process:

The CL register, which indicates the number of rows and asterisks per row, is initialized by the program to 1. The rows are controlled by an outside loop (LOOP1). The program sets the CH register, which acts as the inner loop's counter (LOOP2), to 1 for every row. The inner loop uses DOS interrupt INT 21H (function AH=02H) to print an asterisk and a space. CH is increased when each asterisk-space pair is printed, and the loop keeps going until CH is greater than CL, which indicates that CL asterisks have been printed in the current row. The program outputs a newline sequence (carriage return and line feed) to advance to the following line after the inner loop is finished. After incrementing the outer loop counter (CL), the procedure is repeated until CL reaches 5 (printing 4 rows). After then, the software uses RET to give the operating system back control.

### Implementation:



```
edit: X:\UITS\Suppliments\6th sem\Assen
file  edit  bookmarks  assembler  emulat
new  open  examples  save
01  org 100h
02  mov CL, 1
03
04  LOOP1:
05  mov CH, 1
06
07  LOOP2:
08  mov AH, 2
09  mov DL, '*'
10  int 21h
11
12  mov AH, 2
13  mov DL, ' ',
14  int 21h
15
16  inc CH
17  cmp CH, CL
18
19  jle LOOP2
20
21  mov AH, 2
22  mov DL, 10
23  int 21h
24  mov DL, 13
25  int 21h
26
27  inc CL
28  cmp CL, 5
29  jle LOOP1
30  ret
```

## Result:

The software generates the following output when it is run:

```
*  
* *  
* * *  
* * * *
```

The result is a four-row, right-angled triangle with an increasing number of asterisks (1, 2, 3, 4) in each row, each followed by a space.



## Conclusion:

This software shows how to format output in 8086 assembly code and control nested loops. It effectively creates a geometric pattern using registers and DOS interrupts, demonstrating the usage of repeated procedures to produce visual patterns. The implementation strengthens knowledge of conditional jumps, loop methods, and system interrupts for managing output. The program's ability to handle row and column-based output is demonstrated by its effectiveness in creating a correctly formatted right-angled triangle, which lays the groundwork for more intricate patterns and graphics in assembly programming.

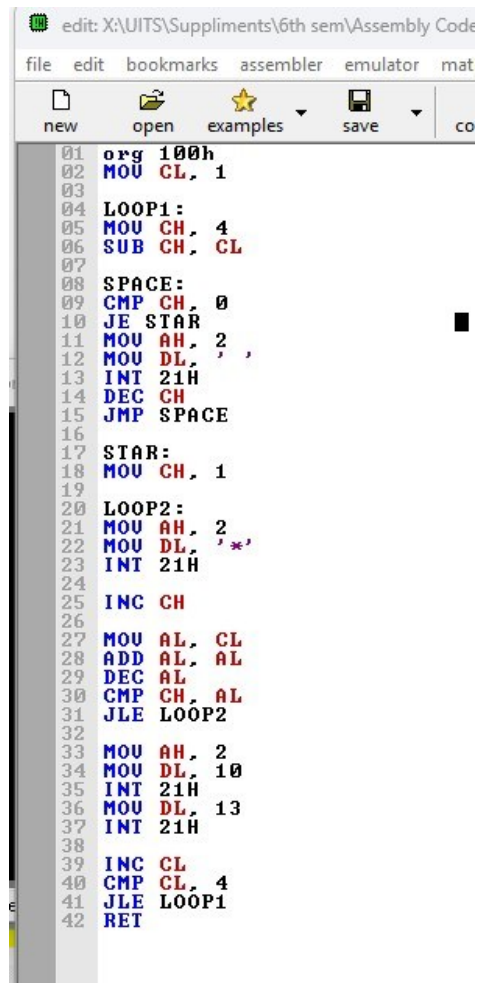
**Problem No: 02****Experiment No: 02**

**Experiment Name:** Write an assembly program to print an equilateral triangle using star (\*)

**Process:**

An equilateral triangle with a foundation of seven asterisks (four rows) is intended to be printed by the program. The current row number, 1, is initialized in the CL register. The program initially subtracts the row number (CL) from 4 for each row to determine how many leading spaces are needed for centering. After that, it uses DOS interrupt INT 21H (function AH=02H) to enter a loop (SPACE) and display these spaces. The software prints the spaces first, then the asterisks. The formula  $2 \times CL - 1$ , which produces odd numbers (1, 3, 5, 7) for rows 1 through 4, determines how many asterisks are in each row. Solid lines are produced by the inner loop (LOOP2), which prints each asterisk without extra spaces in between. The program prints a newline sequence (carriage return and line feed) to advance to the next line after printing all of the asterisks for a row. Until all four rows are displayed, the procedure is repeated after incrementing the row counter (CL). After then, the software gives the operating system back control.

## Implementation:



The screenshot shows an assembly code editor with the following code:

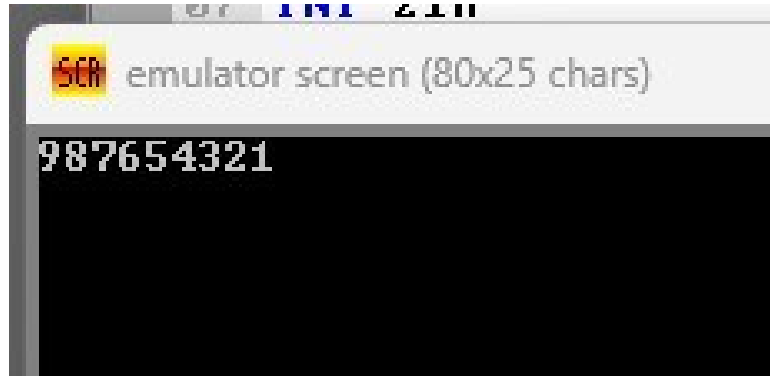
```
01 org 100h
02 MOV CL, 1
03
04 LOOP1:
05 MOV CH, 4
06 SUB CH, CL
07
08 SPACE:
09 CMP CH, 0
10 JE STAR
11 MOV AH, 2
12 MOV DL, ' ',
13 INT 21H
14 DEC CH
15 JMP SPACE
16
17 STAR:
18 MOV CH, 1
19
20 LOOP2:
21 MOV AH, 2
22 MOV DL, '*',
23 INT 21H
24
25 INC CH
26
27 MOV AL, CL
28 ADD AL, AL
29 DEC AL
30 CMP CH, AL
31 JLE LOOP2
32
33 MOV AH, 2
34 MOV DL, 10
35 INT 21H
36 MOV DL, 13
37 INT 21H
38
39 INC CL
40 CMP CL, 4
41 JLE LOOP1
42 RET
```

## Result:

The software generates the following output when it is run:

```
*
***
*****
```

The result is a four-row equilateral triangle with an odd number of asterisks (1, 3, 5, 7) centered on each row with respect to the base. Without any user input, the triangle is formatted and aligned correctly.



### Conclusion:

Advanced loop control, arithmetic computations, and structured output in 8086 assembly language are all demonstrated by this application. It effectively handles asterisk printing and leading spaces using stacked loops, demonstrating how to create geometric patterns programmatically. For display activities, the implementation strengthens knowledge of conditional jumps, register manipulation, and DOS interrupt services. The program's ability to handle symmetric patterns is demonstrated by its successful creation of an equilateral triangle that is visually realistic. This success lays the groundwork for more intricate graphics and pattern-generation jobs in assembly programming.