UITS

UNIVERSITY OF INFORMATION TECHNOLOGY AND SCIENCES

Assignment on

Lab Report- 03

Course Title

Microprocessor and MicroControllers

Course Code

CSE 360

Submitted by

KM Jakaria

Section: 6A

Batch: 53

ID: 043231000510103 7

Submitted to

Md. Ismail

Lecturer

Department of CSE

UITS

Date of Submission

03 Sept, 2025

Problem No: 01

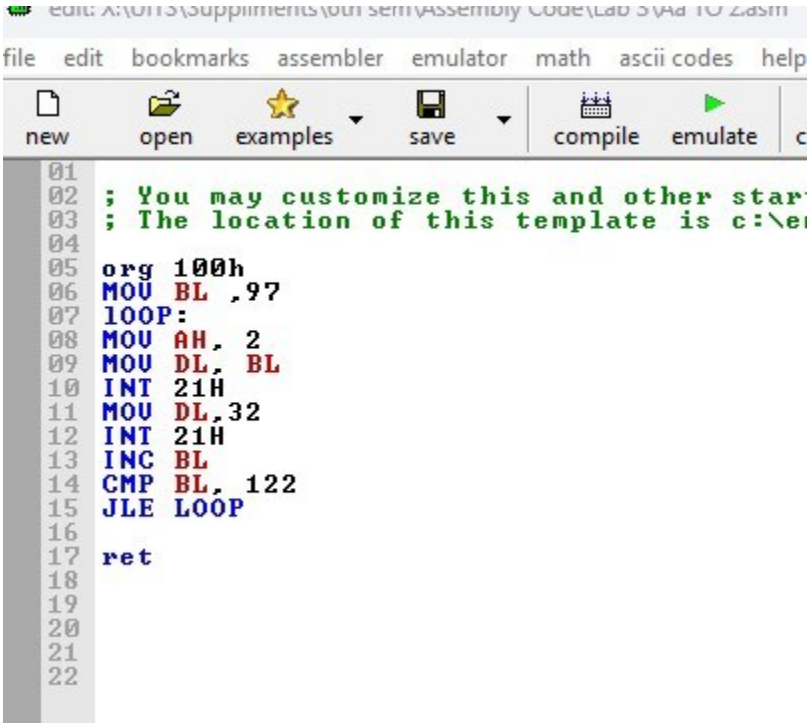
Experiment No: 01

Experiment Name: Write assembly language programs to print letters from a to z and numbers from 9 to 1 as output using a loop

Process:

First, the program is executed by initializing the BL register with the lowercase letter 'a' (97), which is the ASCII value. After that, it moves into a loop structure, with each iteration carrying out the subsequent actions: The DOS interrupt function AH=02H is first configured for character output. INT 21H is used to transfer the current character from BL to DL and print it. Immediately after, another call to INT 21H with AH=02H outputs a space character (ASCII 20H) to separate the letters. The next character in the alphabetical sequence is indicated by incrementing the BL register. Next, BL is compared to the ASCII value 122 ('z') by the software. The loop jumps back to the label LOOP if BL is less than or equal to 122. All letters from 'a' to 'z' are handled thanks to this iterative procedure. The loop ends when BL surpasses 122, and the RET instruction is used to transfer control back to the operating system.

Implementation:

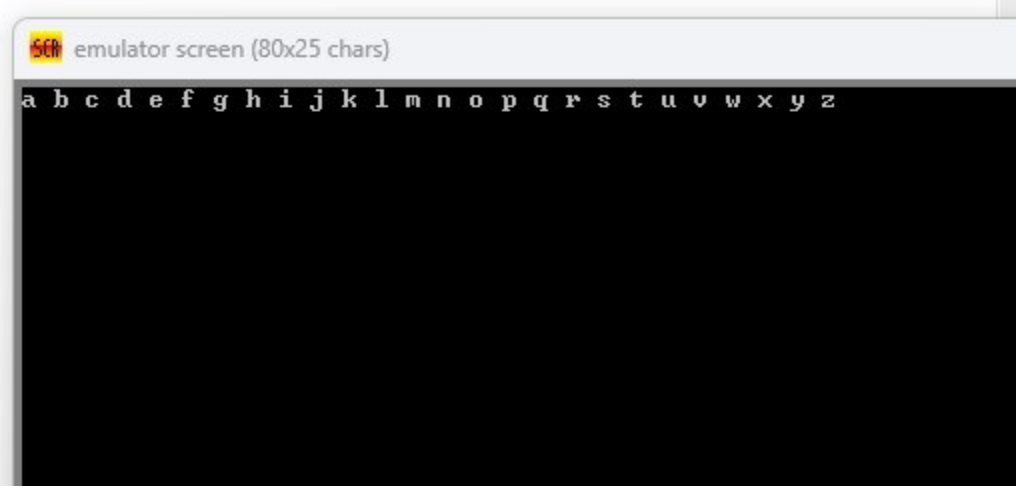


Result:

When the program is executed, it produces a continuous sequence of lowercase letters from 'a' to 'z', with each character followed by a space. The output appears as a single line of text:

a b c d e f g h i j k l m n o p q r s t u v w x y z

The result demonstrates successful iteration through the entire alphabet, with proper spacing between each letter, achieving the intended output without any user input required.



Conclusion:

This program successfully illustrates several important 8086 assembly programming ideas, such as register manipulation, loop control, and using DOS interrupts for output actions. Using a loop technique that checks termination conditions and increases ASCII values, it successfully prints the entire lowercase alphabet. The solution demonstrates the effective use of few instructions to produce the intended result, demonstrating how iterative processes can be used to generate and display sequential data. The exercise strengthens knowledge of conditional jumps, system interrupts, and character encoding (ASCII), laying the groundwork for more complex programming problems including string processing and assembly language iterative algorithms.

Problem No: 02

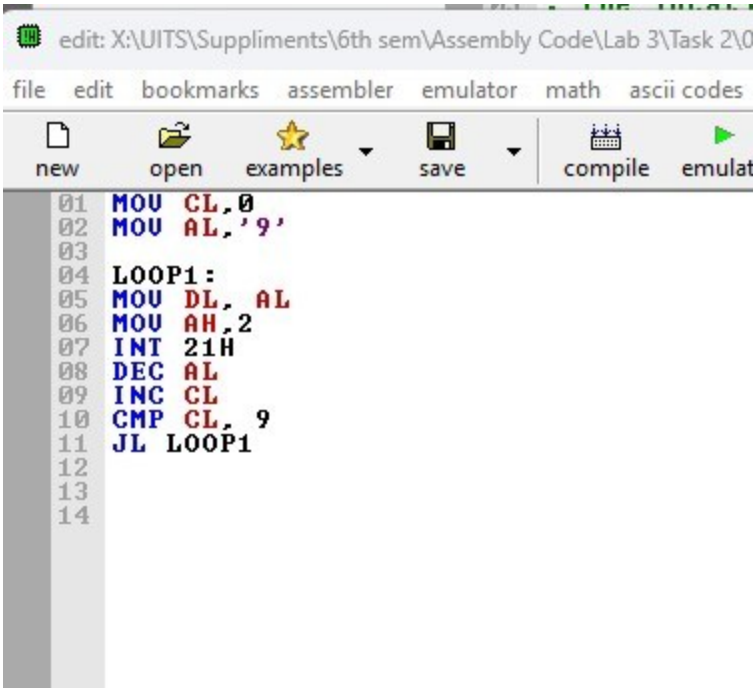
Experiment No: 02

Experiment Name: Write an assembly program to Print 9 to 1 as an output using loop

Process:

The digit '9', or ASCII value 57, is used by the program to initialize the BL register. It then goes into a loop, with each iteration carrying out the subsequent actions: The DOS interrupt function AH=02H is first configured for character output. INT 21H is used to transfer the current digit from BL to DL and print it. A space character (ASCII 20H) is printed to separate the digits immediately after another call to INT 21H with AH=02H. The next lower digit is indicated by decrementing the BL register. After then, the application contrasts BL with ASCII value 48 ('0'). The loop jumps back to the label LOOP if BL is greater than or equal to 48. This iterative procedure guarantees that every digit between '9' and '0' is treated. The loop ends when BL drops below 48, and the program uses the RET command to hand control back to the operating system.

Implementation:

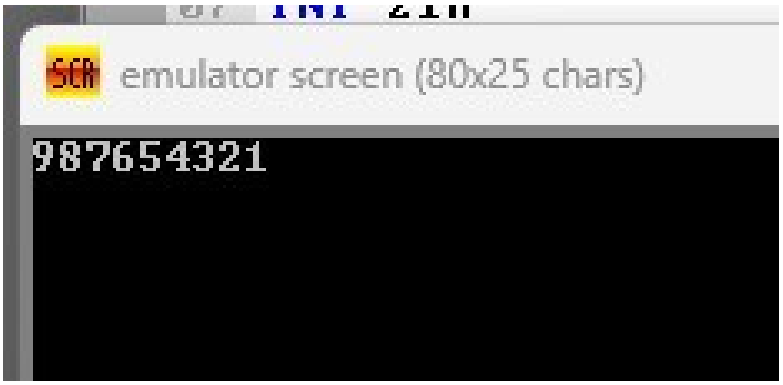


Result:

The program generates a descending sequence of numbers from '9' to '0', with a space between each digit, when it is run. One line of text represents the output:

9 8 7 6 5 4 3 2 1

The outcome shows that the digits were successfully iterated through in reverse order, with appropriate spacing between each digit, producing the desired output without requiring any input from the user.



Conclusion:

This application successfully illustrates output formatting and loop control in 8086 assembly programming. Using a decrementing loop method that modifies ASCII data and verifies termination conditions, it correctly displays numbers from 9 to 0. The implementation emphasizes the effective use of register manipulation for sequential output creation and DOS interrupts for display operations. The practice strengthens knowledge of conditional jumps, ASCII digit representation, and assembly language iterative procedures. The application serves as a basis for more intricate applications including numerical processing and display techniques by clearly demonstrating countdown loops and formatted output.