Name:	Viraj Oza
Roll No:	69
Class/Sem:	SE/IV
Experiment No.:	6
Title:	To perform program to reverse the word in string
Date of Performance:	
Date of Submission:	
Marks:	
Sign of Faculty:	

Aim: Assembly Language Program to reverse the word in string.

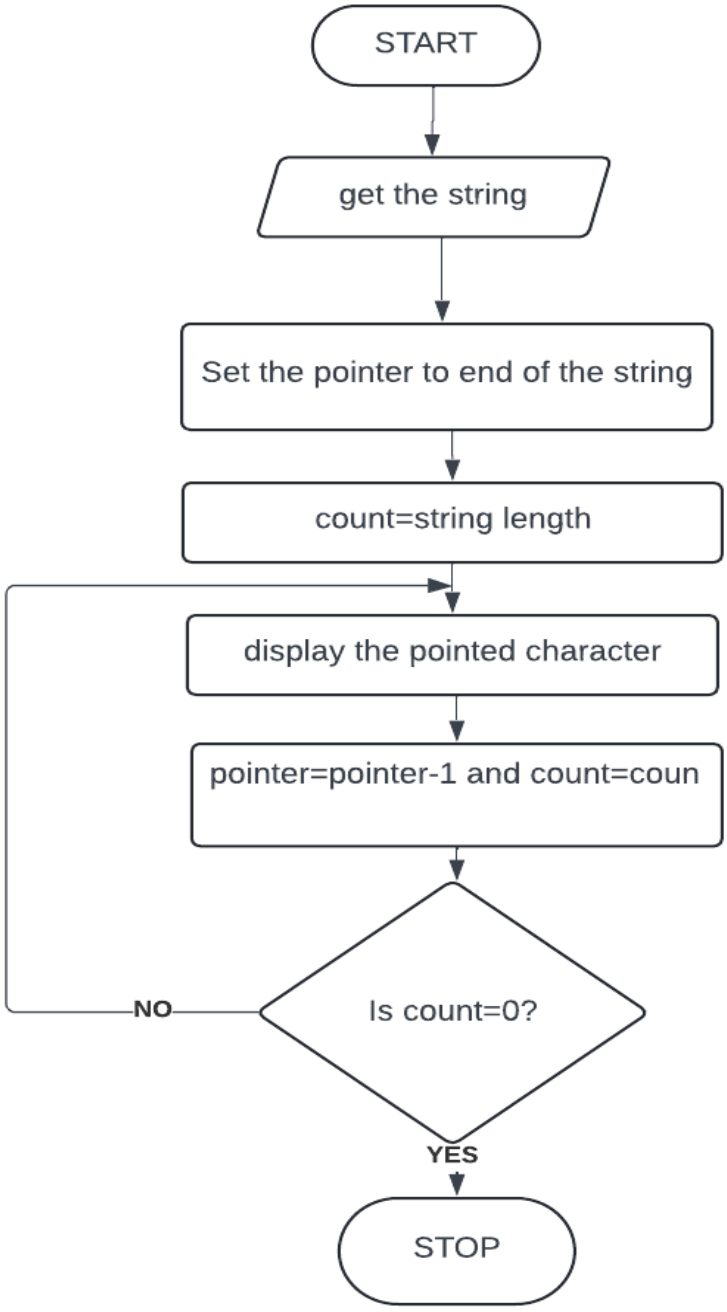
Theory:

This program will read the string entered by the user and then reverse it. Reverse a string is the technique that reverses or changes the order of a given string so that the last character of the string becomes the first character of the string and so on.

Algorithm:

1. Start
2. Initialize the data segment
3. Display the message -1
4. Input the string
5. Display the message 2
6. Take characters count in DI
7. Point to the end character and read it
8. Display the character
9. Decrement the count
10. Repeat until the count is zero
11. To terminate the program using DOS interrupt
    a. Initialize AH with 4ch
    b. Call interrupt INT 21h
12. Stop

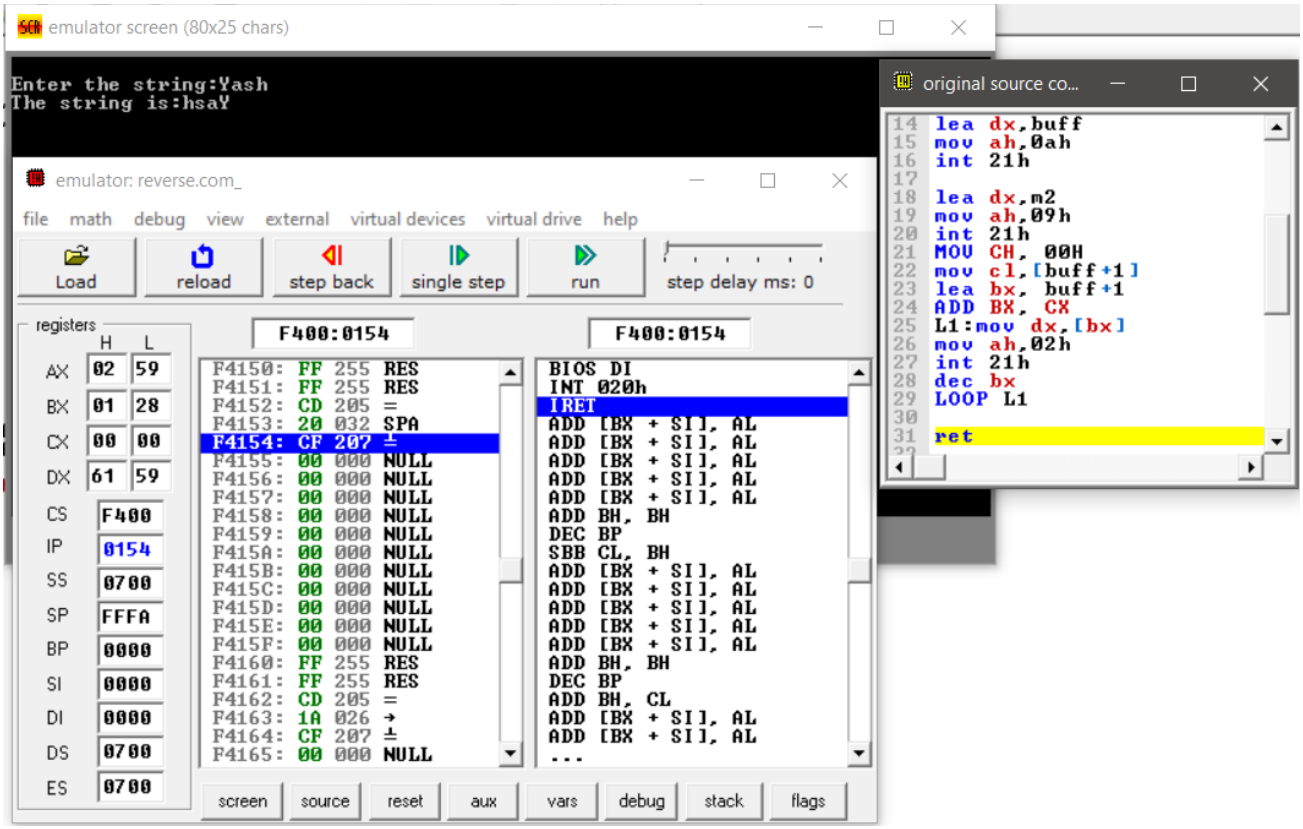
Flowchart


START
get the string
Set the pointer to end of the string
count=string length
display the pointed character
pointer=pointer-1 and count=coun
Is count=0?
NO
YES
STOP

Program:

```
org 100h

.data
m1 db 10,13,'Enter the string:$'
m2 db 10,13,'The string is:$'
buff db 80


.code
lea dx,m1
mov ah,09h
int 21h

lea dx,buff
mov ah,0ah
int 21h

lea dx,m2
mov ah,09h
int 21h
MOV CH, 00H
mov cl,[buff+1]
lea bx, buff+1
ADD BX, CX
L1:mov dx,[bx]
mov ah,02h
int 21h
dec bx
LOOP L1

ret
```

Output:



Conclusion: In this practical exercise, we implemented a program to reverse the order of words in a given string. The program starts by accepting a string input from the user. It then iterates through the string, identifying individual words delimited by spaces. For each word encountered, it reverses the characters within that word. Finally, the program outputs the modified string with the words reversed.

1. Explain the difference between XLAT and XLATB

Ans. XLAT and XLATB are both x86 assembly language instructions used for data manipulation, but they have different functionalities:

XLAT (Translate):

a. The XLAT instruction is used for byte translation in assembly language.
b. It translates a byte at the address formed by combining the contents of the BX register and the AL register with a byte from a lookup table located in memory.
c. After translation, the result replaces the value in the AL register.

XLATB (Byte)

d. The XLATB instruction is an alternative mnemonic for XLAT.
e. It performs the same function as XLAT but is used for byte operations, specifically emphasizing that only a byte is being translated.

2. Explain the instruction LAHF.

Ans. The `LAHF` instruction is a mnemonic in x86 assembly language, standing for "Load AH from Flags". Here's a breakdown of its functionality:

Purpose:

- `LAHF` is used to load the status flags from the processor's flags register (EFLAGS) into the AH register.

Registers:

- AH: The upper 8 bits of the AX register, which is part of the general-purpose registers in x86 architecture.
- EFLAGS: The flags register containing various status flags such as zero, carry, parity, etc.

Operation:

- When `LAHF` is executed, the status flags from the EFLAGS register are copied into the AH register in the following arrangement:
  - Bit 0 (CF) of EFLAGS is copied to bit 0 (CF) of AH.
  - Bit 2 (PF) of EFLAGS is copied to bit 2 (PF) of AH.
  - Bit 4 (AF) of EFLAGS is copied to bit 4 (AF) of AH.
  - Bit 6 (ZF) of EFLAGS is copied to bit 6 (ZF) of AH.
  - Bit 7 (SF) of EFLAGS is copied to bit 7 (SF) of AH.
  - Bit 8 (TF) of EFLAGS is copied to bit 8 (TF) of AH.
  - Bit 9 (IF) of EFLAGS is copied to bit 9 (IF) of AH.
  - Bit 10 (DF) of EFLAGS is copied to bit 10 (DF) of AH.
  - Bit 11 (OF) of EFLAGS is copied to bit 11 (OF) of AH.

Usage:

- `LAHF` is typically used in conjunction with the `SAHF` (Store AH into Flags) instruction to save and restore status flags during interrupt service routines or other context switches.
- It can also be used in conditional branching or decision-making based on the status of specific flags.