Name:	Preeti Rambahal Prajapati
Roll No:	48
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:

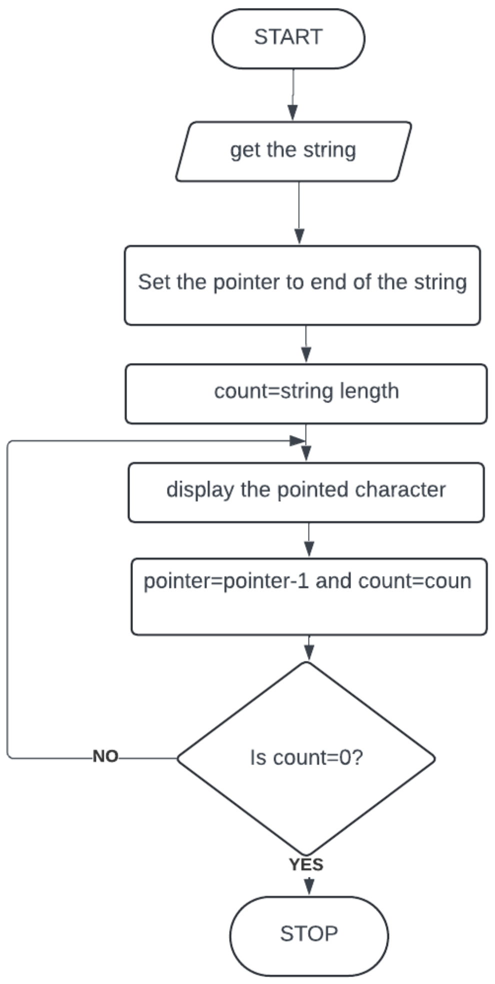
This program will read the string entered by the user and then reverse it. Reverse a string is the technique that reverses or changes the order of a given string so that the last character of the string becomes the first character of the string and so on.

Algorithm:

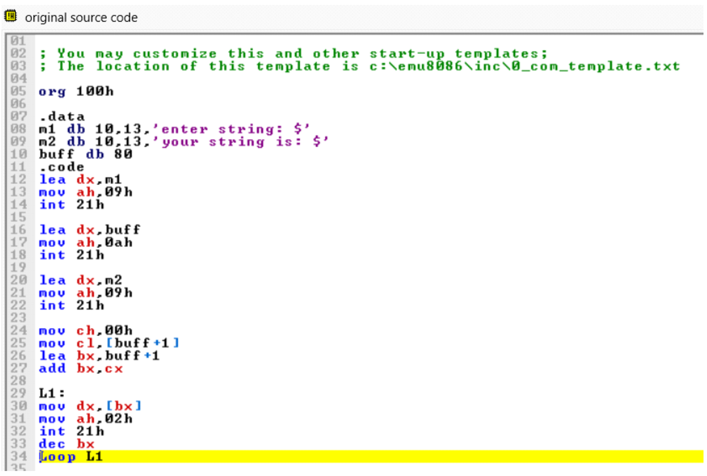
- Start
- Initialize the data segment
- Display the message -1
- Input the string
- Display the message 2
- Take characters count in DI
- Point to the end character and read it
- Display the character

- Decrement the count
- Repeat until the count is zero
- To terminate the program using DOS interrupt
    - Initialize AH with 4ch
    - Call interrupt INT 21h
- Stop

Flowchart:

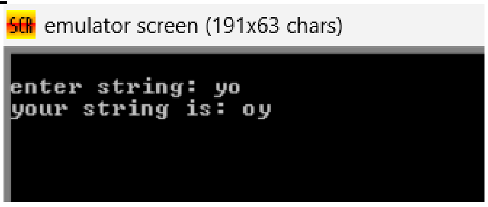

START
get the string
Set the pointer to end of the string
count=string length
display the pointed character
pointer=pointer-1 and count=coun
Is count=0?
NO
YES
STOP

Assembly Code:


original source code

```

; You may customize this and other start-up templates;
; The location of this template is c:\emu8086\inc\0_com_template.txt

org 100h

.data
m1 db 10,13,'enter string: $'
m2 db 10,13,'your 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
```

Output:


emulator screen (191x63 chars)

```
enter string: yo
your string is: oy
```

Conclusion:

- Explain the difference between XLAT and XLATB

Ans. XLAT and XLATB are x86 assembly language instructions used for table lookups, but they have some differences:

- XLAT:
  - XLAT is an instruction that performs a lookup in a translation table.
  - It uses the AL register as an index to access a byte-sized entry in a translation table, typically located in memory.
  - After the lookup, XLAT replaces the content of AL with the byte value found at the memory address calculated using the sum of the contents of AL and the base address of the translation table.
- XLATB:
  - XLATB is essentially the same as XLAT, but it’s a legacy alternative mnemonic for the same instruction. The “B” suffix in XLATB stands for “byte.”
  - In terms of functionality, XLATB operates exactly like XLAT and performs the same table lookup using the AL register as an index.

In summary, both XLAT and XLATB instructions perform byte-sized table lookups using the content of the AL register as an index. The difference lies in their naming convention, with XLAT being the more commonly used mnemonic, while XLATB is a legacy alternative. The choice between them is mostly a matter of programmer preference.

- Explain the instruction LAHF.

Ans. The LAHF instruction, standing for “Load AH from Flags,” is an x86 assembly language instruction used to load the contents of the flags register into the AH register. Here’s an explanation of the LAHF instruction:

- Purpose:
  - LAHF is used to transfer the status of the lower byte of the flags register (EFLAGS) into the AH register.
  - The flags register contains various status flags that reflect the outcome of arithmetic and logical operations, such as the zero flag (ZF), sign flag (SF), carry flag (CF), and others.
- Usage:
  - LAHF is typically used in conjunction with the SAHF instruction (Store AH into Flags), which performs the opposite operation.
  - Before using LAHF, the programmer often saves the contents of the AH register if it holds important data, as LAHF will overwrite its contents.
  - After executing LAHF, the AH register contains the lower byte of the flags register, where each bit corresponds to a specific flag’s status.
- Example:
  - For instance, if the carry flag (CF) and the zero flag (ZF) are set in the flags register, the value loaded into AH by LAHF would be 00000011 in binary, corresponding to a hexadecimal value of 0x03.
- Flags Represented:
  - The bits in the AH register represent the status of the following flags (from bit 0 to bit 7): CF, reserved, PF, reserved, AF, reserved, ZF, SF.

In summary, the LAHF instruction is used to load the lower byte of the flags register into the AH register, providing access to the status of various processor flags for further manipulation or decision-making in assembly language programs.