Assignment 7  
Write 64-bit ALP to perform following string operations i) Length of String ii) Reverse of String  
  
section .data

prompt db "Enter a string: ", 0 ; Prompt for user input

result\_len db "Length: ", 0

result\_rev db "Reversed: ", 0

newline db 0xA, 0 ; Newline character

section .bss

input resb 128 ; Reserve 128 bytes for user input

length resb 1 ; Reserve space for length (1 byte)

reverse resb 128 ; Reserve space for reversed string

section .text

global \_start

\_start:

; Print the prompt

mov rax, 1 ; syscall: write

mov rdi, 1 ; file descriptor: stdout

mov rsi, prompt ; Message: "Enter a string: "

mov rdx, 15 ; Length of the message

syscall

; Read the user input

mov rax, 0 ; syscall: read

mov rdi, 0 ; file descriptor: stdin

lea rsi, [input] ; Buffer to store the input

mov rdx, 128 ; Max number of bytes to read

syscall

; Calculate the length of the string

xor rcx, rcx ; RCX will hold the length (initialize to 0)

lea rsi, [input] ; Load the address of the input string

find\_length:

cmp byte [rsi + rcx], 0xA ; Check for newline character (end of input)

je done\_length ; If newline found, end the loop

cmp byte [rsi + rcx], 0 ; Check for null terminator

je done\_length ; If null byte found, end the loop

inc rcx ; Increment length counter

jmp find\_length ; Continue to next byte

done\_length:

mov byte [length], cl ; Store the length in the 'length' buffer

; Reverse the string

lea rsi, [input] ; Load the address of the input string

lea rdi, [reverse] ; Load the address of the reverse buffer

mov rdx, rcx ; RDX contains the length of the string

dec rcx ; RCX is the index of the last character

reverse\_loop:

mov al, [rsi + rcx] ; Load character from the end of the string

mov [rdi], al ; Store character in the reverse buffer

inc rdi ; Move to the next position in reverse buffer

dec rcx ; Move to the previous character in the input string

jns reverse\_loop ; Repeat until RCX becomes -1

mov byte [rdi], 0 ; Null-terminate the reversed string

; Print the result length

mov rax, 1 ; syscall: write

mov rdi, 1 ; file descriptor: stdout

mov rsi, result\_len ; Message: "Length: "

mov rdx, 8 ; Length of the message

syscall

movzx rax, byte [length] ; Load the length into RAX

add al, '0' ; Convert to ASCII character

mov rsi, rsp ; Store the ASCII character on stack

mov byte [rsi], al ; Store it

mov rax, 1 ; syscall: write

mov rdi, 1 ; file descriptor: stdout

mov rdx, 1 ; Print 1 byte

syscall

; Print the newline

mov rax, 1 ; syscall: write

mov rdi, 1 ; file descriptor: stdout

mov rsi, newline ; Newline character

mov rdx, 1 ; Length of newline

syscall

; Print the reversed string

mov rax, 1 ; syscall: write

mov rdi, 1 ; file descriptor: stdout

mov rsi, result\_rev ; Message: "Reversed: "

mov rdx, 10 ; Length of the message

syscall

; Print the reversed string

mov rax, 1 ; syscall: write

mov rdi, 1 ; file descriptor: stdout

lea rsi, [reverse] ; Load reversed string

movzx rdx, byte [length] ; Use the length of the reversed string

syscall

; Exit program

mov rax, 60 ; syscall: exit

xor rdi, rdi ; exit code 0

syscall