Title: Write 64-bit ALP to perform following string operations

1. Length of String

2. Reverse of String

Code:

%macro scall 4

mov rax,%1

mov rdi,%2

mov rsi,%3

mov rdx,%4

syscall

%endmacro

section .data

menumsg db 10,10,"#### Menu for String Operation####",10

db 10,"1. Enter the String"

db 10,"2. Caculate the Length"

db 10,"3. Reverse the String"

db 10,"4. Exit"

db 10,"Enter your Choice",10

menumsg\_len equ $-menumsg

m1 db 10,"Enter the String::"

l1 equ $-m1

m2 db 10,"Length of the String is::"

l2 equ $-m2

m3 db 10,"Reversed String is ::"

l3 equ $-m3

m4 db 10,"Do you want to continue::"

l4 equ $-m4

m5 db 10,"Wrong Choice Entered Please Try Again!!!",10,10

l5 equ $-m5

section .bss

accbuff resb 50

accbuff\_len equ $-accbuff

resbuff resb 50

resbuff\_len equ $-resbuff

dnumbuff resb 16

choice resb 02

acctlen resq 1

section .text

global \_start:

\_start:

menu:

scall 1,1,menumsg,menumsg\_len

scall 0,0,choice,02

cmp byte[choice],'1'

jne case2

call entstr\_proc

jmp exit1

case2:

cmp byte[choice],'2'

jne case3

call length\_proc

jmp exit1

case3:

cmp byte[choice],'3'

jne case4

call reverse\_proc

jmp exit1

case4:

cmp byte[choice],'4'

je case4

scall 1,1,m5,l5

jmp menu

exit1:

scall 1,1,m4,l4

scall 0,0,choice,02

cmp byte[choice],'y'

jne exit

jmp menu

exit:

mov rax,60

mov rbx,00

syscall

;----------------- ENTER STRING ---------------------

entstr\_proc:

scall 1,1,m1,l1

scall 0,0,accbuff,accbuff\_len

dec rax

mov [acctlen],rax

ret

;-------------- LENGTH OF STRING--------------------

length\_proc:

scall 1,1,m2,l2

mov rbx,[acctlen]

call disp64\_proc

ret

;----------------DISPLAY PROCEDURE----------------------

disp64\_proc:

mov rdi,dnumbuff

mov rcx,16

dispUp1:

rol rbx,4

mov dl,bl

and dl,0fh

cmp dl,09h

jbe next

add dl,07h

next:

add dl,30h

mov [rdi],dl

inc rdi

loop dispUp1

scall 01,01,dnumbuff,16

ret

;--------------------REVERSE STRING---------------------

reverse\_proc:

mov rsi,accbuff

mov rdi,resbuff

mov rcx,[acctlen]

add rsi,rcx

dec rsi

again:

mov al,[rsi]

mov [rdi],al

dec rsi

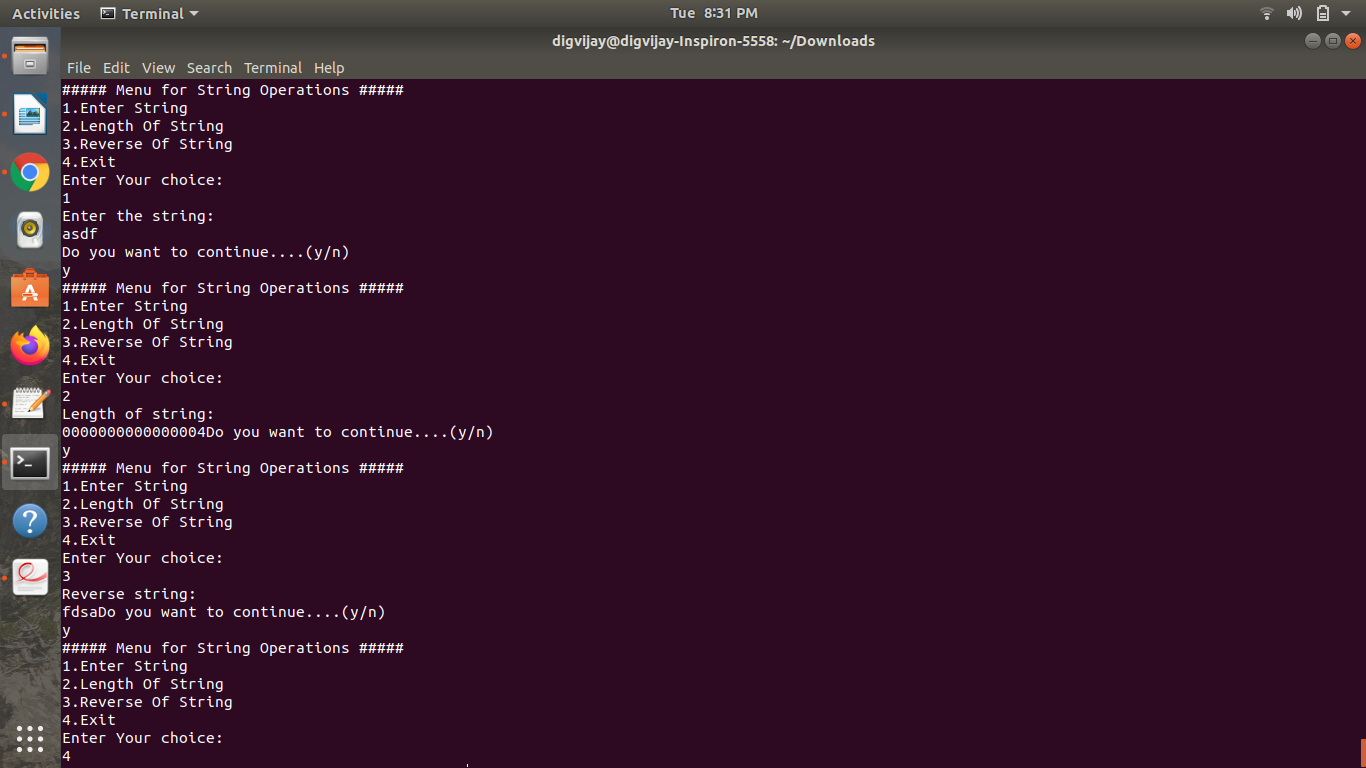
inc rdi

loop again

scall 1,1,m3,l3

scall 1,1,resbuff,[acctlen]

ret

Output: