Computer Organisation & Architecture Assignment No. 4

Name: Akash Jivendra Bachhav

Batch: C3

Roll No: 199

Problem Statement:

Write an ALP program to implement various string operations.

Source Code:

```
%macro read 2
mov rax, 0
mov rdi, 0
mov rsi, %1
mov rdx, %2
syscall
%endmacro
%macro write 2
mov rax, 1
mov rdi, 1
mov rsi, %1
mov rdx, %2
syscall
%endmacro
section .data
menumsg db 10, "1. String length", 10
db "2. String copy", 10
db "3. String reverse", 10
db "4. String compare", 10
db "5. String concat", 10
db "6. Check palindrome", 10
db "7. String substring", 10
db "8. Exit", 10
db "Enter your choice: 1-8", 10
menulen equ $ - menumsg
msg1 db "Enter String 1", 10
len1 equ $ - msg1
msg2 db "Enter String 2", 10
```

```
len2 equ $ - msg2
msg3 db "The length of string: ", 10
len3 equ $ - msg3
msg4 db "The copied string: ", 10
len4 equ $ - msg4
msg5 db "The reverse String: ", 10
len5 equ $ - msg5
msg6 db "Strings are equal", 10
len6 equ $ - msg6
msg7 db "Strings are not equal", 10
len7 equ $ - msg7
msg8 db "The concatenated string: ", 10
len8 equ $ - msg8
msg9 db "String is a palindrome", 10
len9 equ $ - msg9
msg10 db "String is not a palindrome", 10
len10 equ $-msg10
msg11 db "Substring", 10
len11 equ $-msg11
msg12 db "Not a substring", 10
len12 equ $-msg12
msg13 db "Wrong choice", 10
len13 equ $-msg13
section .bss
string1 resb 20
string2 resb 20
string3 resb 40
l1 resq 1
l2 resq 1
l3 resq 1
choice resb 2
buff resb 16
char_buff resb 16
section .text
global _start
_start:
write msg1, len1
read string1, 20
dec rax
mov [l1], rax
write msg2, len2
read string2, 20
dec rax
mov [l2], rax
printmenu:
```

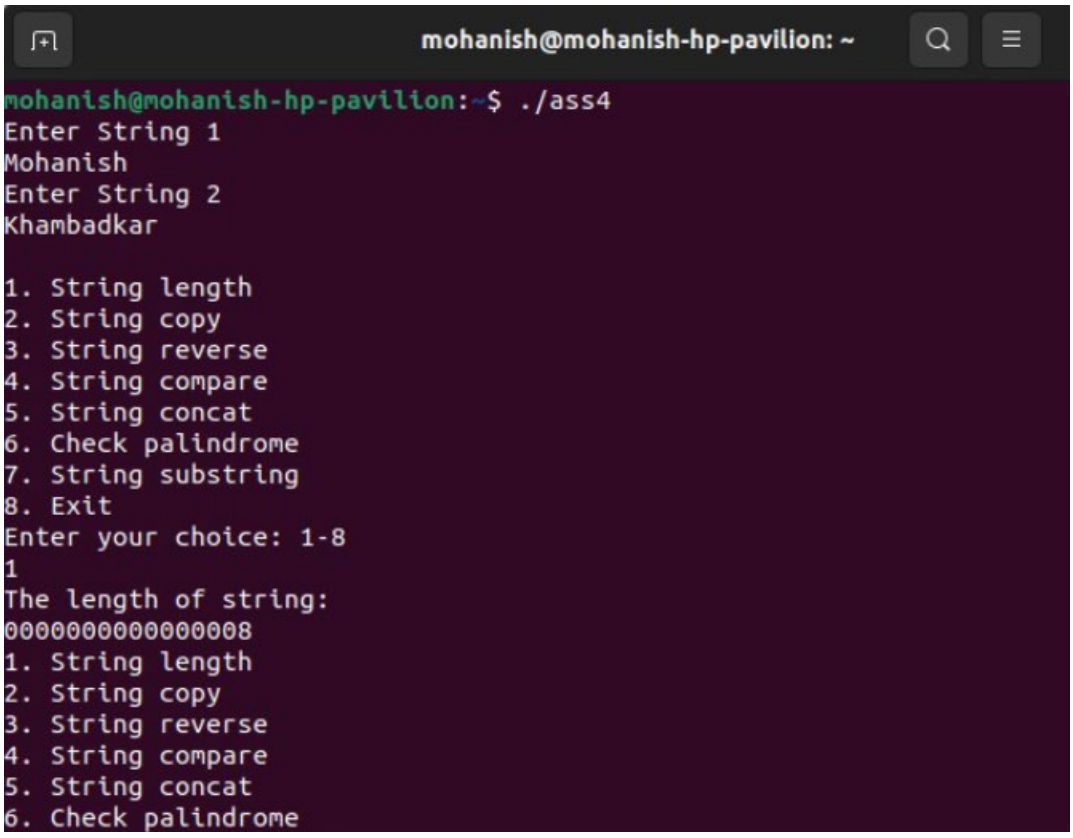
```
write menumsg, menulen
read choice, 2
cmp byte [choice], '1'
je strlen
cmp byte [choice], '2'
je strcpy
cmp byte [choice], '3'
je strrev
cmp byte [choice], '4'
je strcmp
cmp byte [choice], '5'
je strcat
cmp byte [choice], '6'
je strpalindrome
cmp byte [choice], '7'
je strsub
cmp byte [choice], '8'
je exit
write msg13, len13
jmp printmenu
strlen:
write msg3, len3
mov rbx, [l1]
call display
jmp printmenu
strcpy:
mov rsi, string1
mov rdi, string3
mov rcx, [l1]
cld ; clears the direction for incrementing the pointer
rep movsb ; to copy a string item byte by byte.
write msg4, len4
write string3, [l1]
jmp printmenu
strrev:
mov rsi, string1
add rsi, [l1]
dec rsi
mov rdi, string3
mov rcx, [l1]
up:
mov bl, byte [rsi]
mov byte [rdi], bl
dec rsi
inc rdi
```

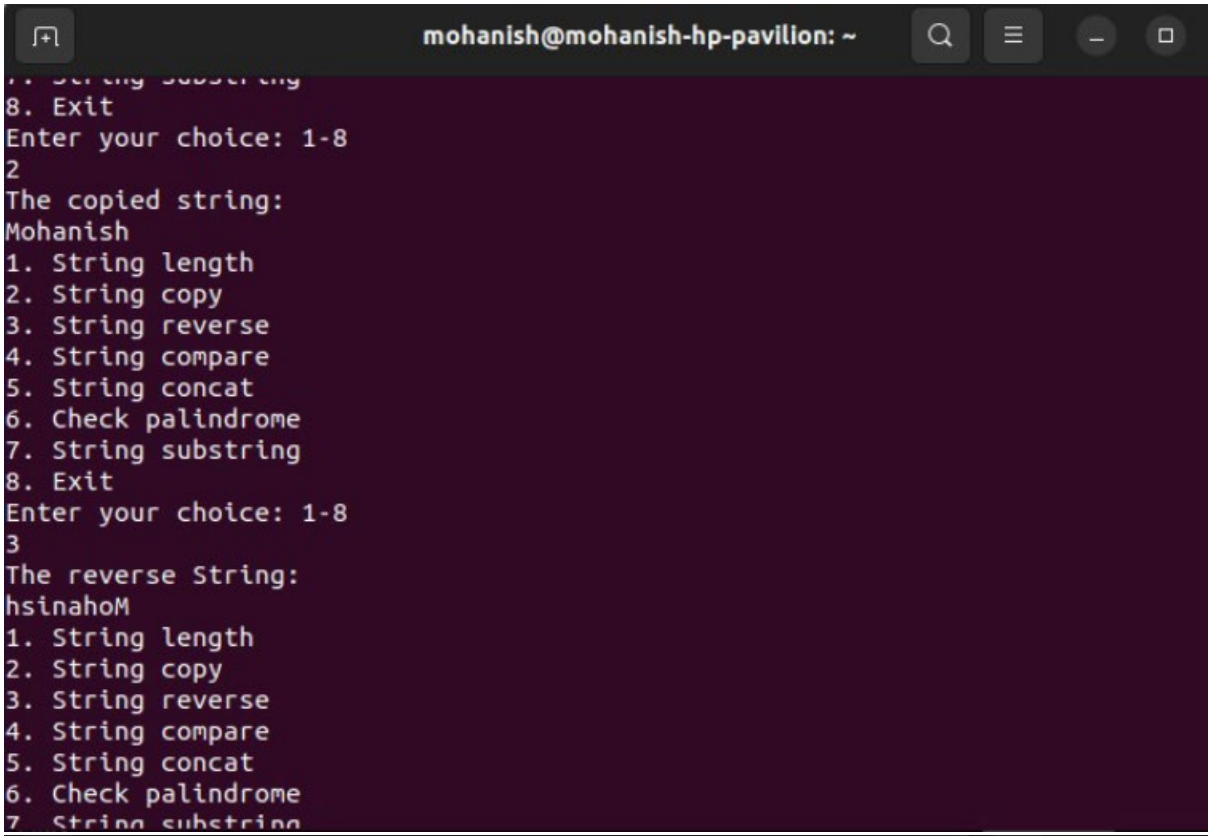
```
dec rcx
jnz up
write msg5, len5
write string3, [l1]
jmp printmenu
strcmp:
mov rbx, [l1]
cmp rbx, qword [l2]
jne nonequal
mov rsi, string1 ; source index
mov rdi, string2 ; destination index
mov rcx, [l1]
cld ; clears the direction flag to incrementing the pointer
repe cmpsb ; Comparison of two strings using REPE (repeat
till equal) prefix
jne nonequal
write msg6, len6
jmp printmenu
nonequal:
write msg7, len7
jmp printmenu
strcat:
mov rsi, string1 ; source
mov rdi, string3 ; destination
mov rcx, [l1]
cld ; clears the direction flag to incrementing the pointer
rep movsb ; to copy a data item byte by byte
mov rsi, string2 ; source
mov rcx, [l2]
rep movsb
mov rbx, [l1]
add rbx, [l2]
mov [l3], rbx
write msg8, len8
write string3, [l3]
jmp printmenu
strpalindrome:
write msg1, len1
read string1, 20
dec rax
mov [l1], rax
mov rsi, string1
add rsi, [l1]
dec rsi
mov rdi, string3
```

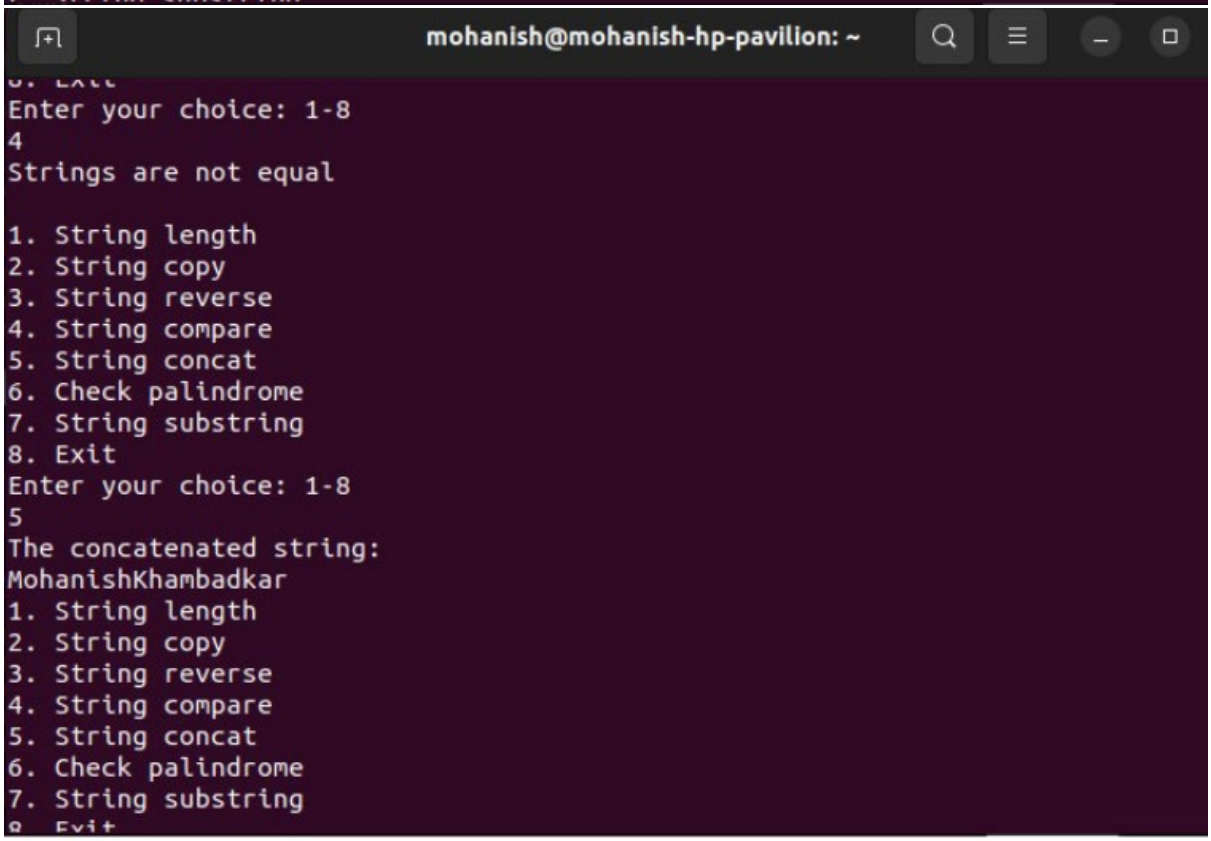
```
mov rcx, [l1]
up1:
mov dl, byte [rsi]
mov byte [rdi], dl
dec rsi
inc rdi
dec rcx
jnz up1
mov rsi, string1
mov rdi, string3
mov rcx, [l1]
cld
repe cmpsb ; repeat until equal to compare string byte
jne notequal1
write msg9, len9
jmp printmenu
notequal1:
write msg10, len10
jmp printmenu
strsub:
write msg11, len11
write msg1, len1
read string1, 20
dec rax
mov [l1], rax
mov rbx, qword [l1]
same:
inc rsi
inc rdi
dec rbx
dec qword [l1]
cmp rbx, 0 ; compare with l2 with 0
je st
cmp qword [l1], 0 ; compare till the last character of string
one
jne same
write msg12, len12
jmp printmenu
st:
write msg11, len11
jmp printmenu
exit:
mov rax, 60
xor rdi, rdi
syscall
```

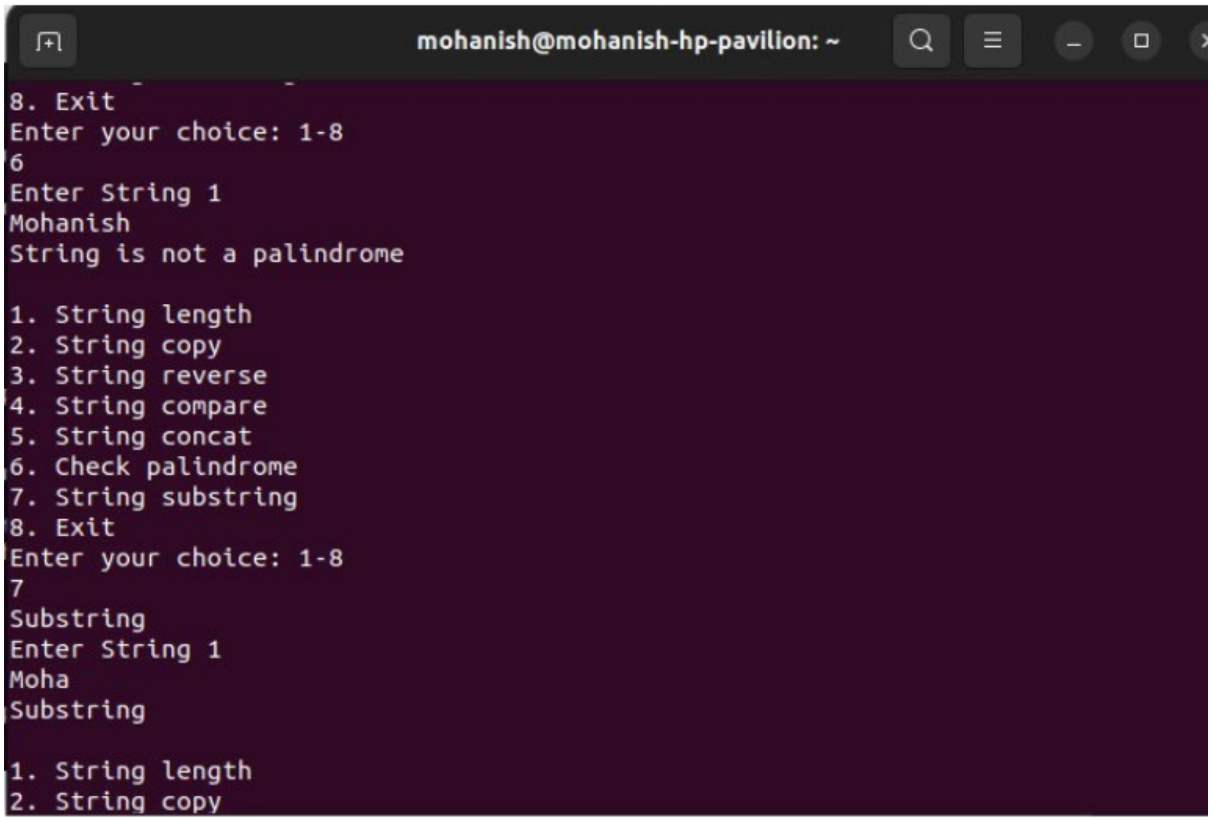
```
display:
mov rsi, char_buff
mov rcx, 16
up2:
rol rbx, 4
mov dl, bl
and dl, 0FH
cmp dl, 09H
jbe add30
add dl, 07H
add30:
add dl, 30H
mov byte [rsi], dl
inc rsi
dec rcx
jnz up2
write char_buff, 16
ret
```

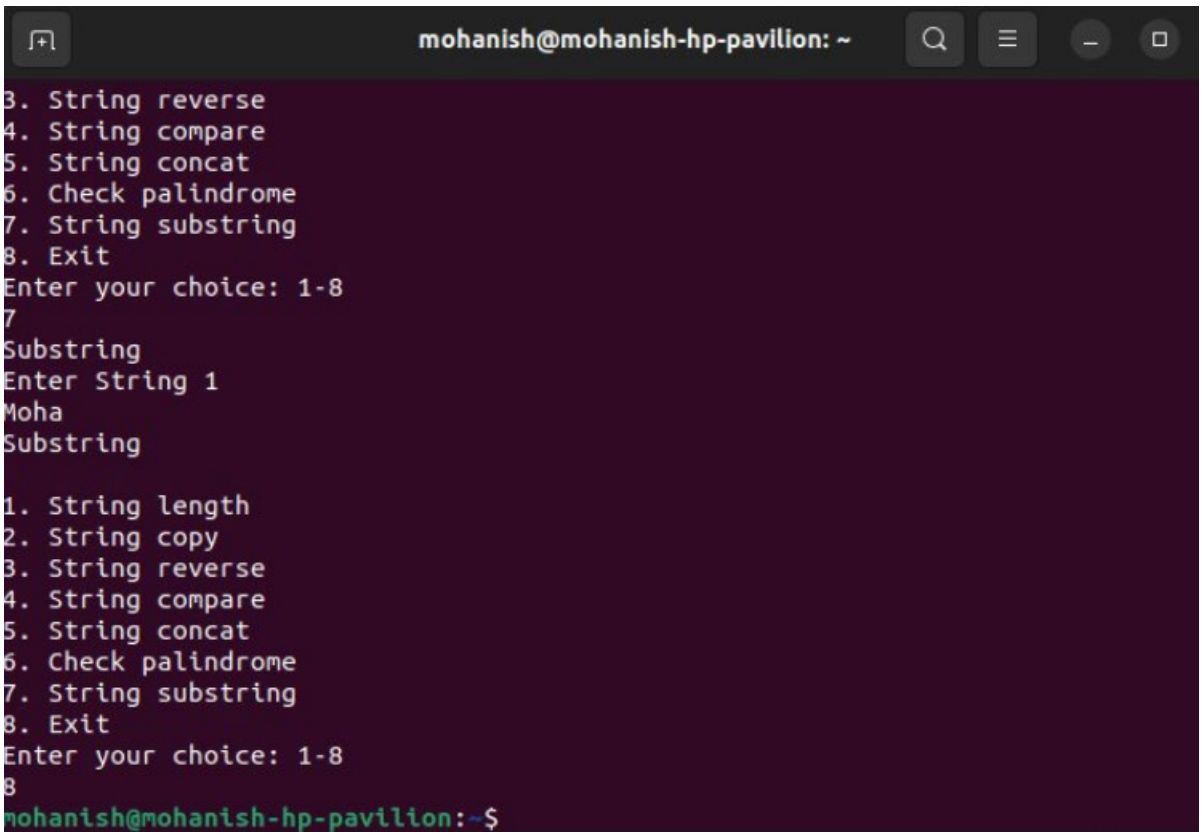
- **Output Screen:**




mohanish@mohanish-hp-pavilion: ~
8. Exit
Enter your choice: 1-8
2
The copied string:
Mohanish
1. String length
2. String copy
3. String reverse
4. String compare
5. String concat
6. Check palindrome
7. String substring
8. Exit
Enter your choice: 1-8
3
The reverse String:
hsinahoM
1. String length
2. String copy
3. String reverse
4. String compare
5. String concat
6. Check palindrome


mohanish@mohanish-hp-pavilion: ~
Enter your choice: 1-8
4
Strings are not equal
1. String length
2. String copy
3. String reverse
4. String compare
5. String concat
6. Check palindrome
7. String substring
8. Exit
Enter your choice: 1-8
5
The concatenated string:
MohanishKhambadkar
1. String length
2. String copy
3. String reverse
4. String compare
5. String concat
6. Check palindrome
7. String substring


mohanish@mohanish-hp-pavilion: ~
8. Exit
Enter your choice: 1-8
6
Enter String 1
Mohanish
String is not a palindrome
1. String length
2. String copy
3. String reverse
4. String compare
5. String concat
6. Check palindrome
7. String substring
8. Exit
Enter your choice: 1-8
7
Substring
Enter String 1
Moha
Substring
1. String length
2. String copy


mohanish@mohanish-hp-pavilion: ~
3. String reverse
4. String compare
5. String concat
6. Check palindrome
7. String substring
8. Exit
Enter your choice: 1-8
7
Substring
Enter String 1
Moha
Substring
1. String length
2. String copy
3. String reverse
4. String compare
5. String concat
6. Check palindrome
7. String substring
8. Exit
Enter your choice: 1-8
8
mohanish@mohanish-hp-pavilion:~$