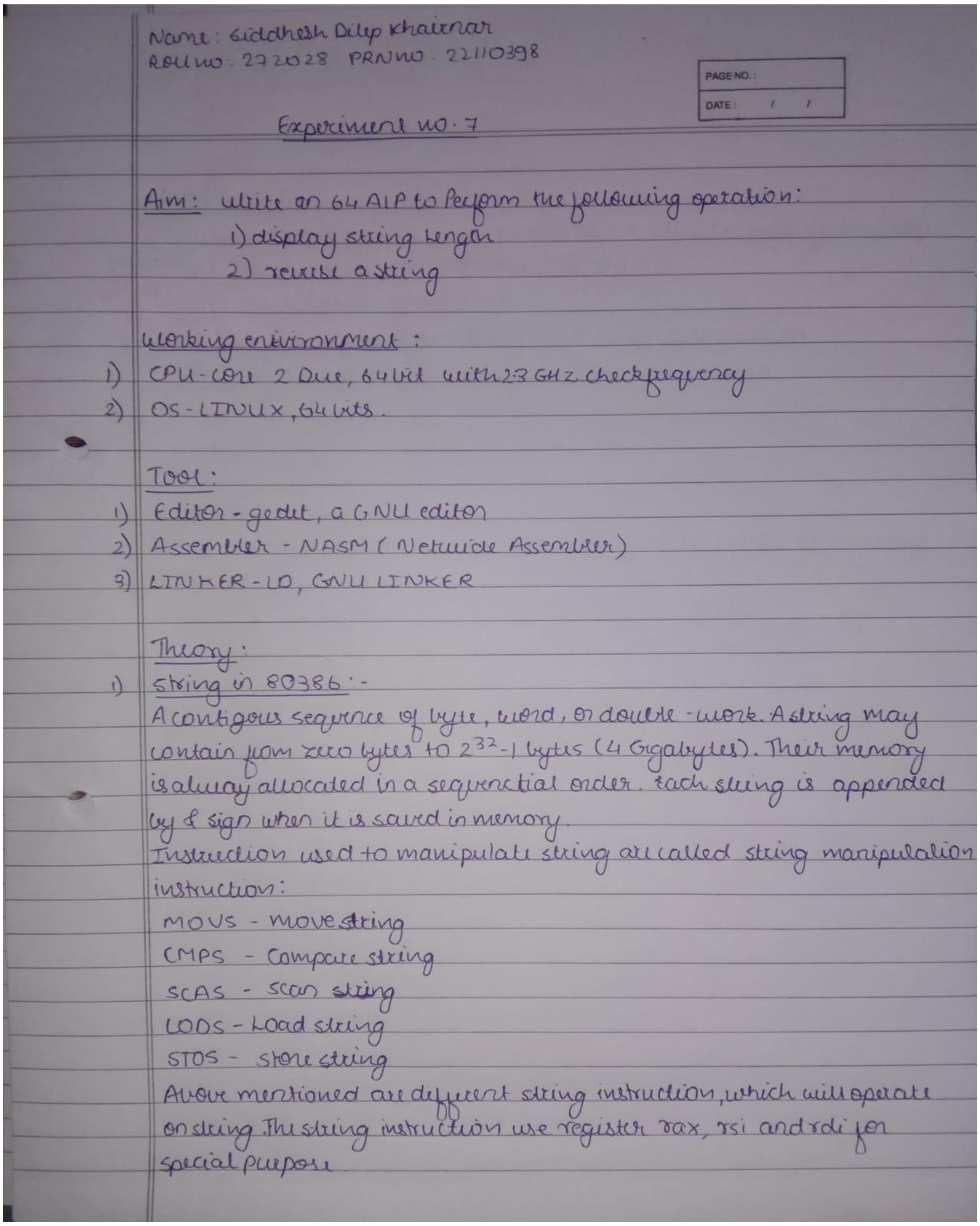
MP Practical- 7

Name- Siddhesh Dilip Khairnar

Roll No.- 272028

Batch- B2



Name: Siddhesh Dilip Khairnar
Roll no: 272028 PRN no: 22110398

PAGE NO.:
DATE: / /

Experiment no. 7

Aim: write an 64 ALP to Perform the following operation:
1) display string length
2) reverse a string

Working environment:
1) CPU-core 2 Duo, 64 bit with 2.3 GHz check frequency
2) OS - LINUX, 64 bits.

Tool:
1) Editor - gedit, a GNU editor
2) Assembler - NASM (Network Assembler)
3) LINKER - LD, GNU LINKER

Theory:
1) string in 80386:-
A contigous sequence of byte, word, or double-work. A string may contain from zero bytes to $2^{32}-1$ bytes (4 Gigabytes). Their memory is always allocated in a sequenctial order. Each string is appended by & sign when it is saved in memory.
Instruction used to manipulate string are called string manipulation instruction:
MOVS - move string
CMPS - Compare string
SCAS - Scan string
LODS - Load string
STOS - store string
Above mentioned are different string instruction, which will operate on string. The string instruction use register rax, rsi and rdi for special purpose

PAGE NO.:

DATE: / /

Algorithm :-

1) Initialize rax register, index register to execute read system call.
2) Accept the character read & store in a variable.
3) Perform ascii to hex conversion.
4) calculate the length of the string read using accumulator.
5) Display the string length using macros on the screen.
6) Exit.

Conclusion :-

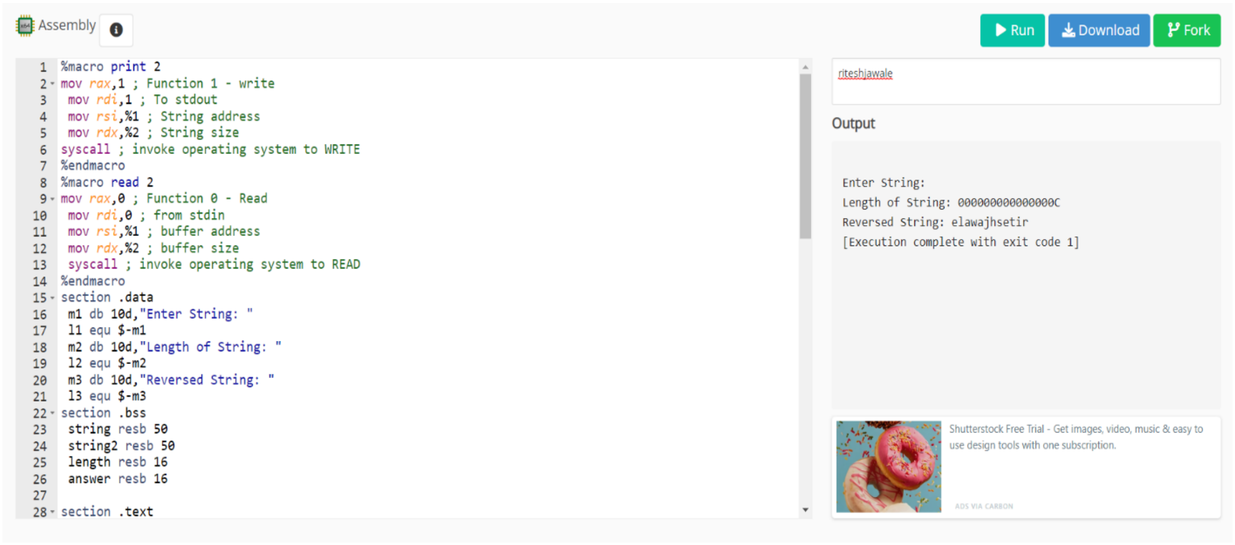
Hence we accepted the string and displayed the length of the string on the screen.

Code:

```
%macro print 2
mov rax,1 ; Function 1 - write
 mov rdi,1 ; To stdout
 mov rsi,%1 ; String address
 mov rdx,%2 ; String size
syscall ; invoke operating system to WRITE
%endmacro
%macro read 2
mov rax,0 ; Function 0 - Read
 mov rdi,0 ; from stdin
 mov rsi,%1 ; buffer address
 mov rdx,%2 ; buffer size
 syscall ; invoke operating system to READ
%endmacro
section .data
 m1 db 10d,"Enter String: "
 l1 equ $-m1
 m2 db 10d,"Length of String: "
 l2 equ $-m2
 m3 db 10d,"Reversed String: "
 l3 equ $-m3
section .bss
 string resb 50
 string2 resb 50
 length resb 16
 answer resb 16
```

```
section .text

 global _start

_start:

print m1,l1

read string,50

;length is returned in rax

;decrement once to remove count of Enter character

 ;dec rax

 mov [length],rax

 print m2,l2

 mov rax,[length]


 mov rsi,answer+15

 mov rcx,16

loop1: mov rdx,0

 mov rbx,16

 div rbx

 cmp dl,09h

 jbe skip1


 add dl,07h

skip1: add dl,30h

 mov [rsi],dl


 dec rsi

 dec rcx

 jnz loop1

 print answer,16

 mov rsi,string
 mov rdi,string2
 mov rcx,[length]
 add rdi,rcx
 dec rdi

loop2:
 mov al,[rsi]
 mov [rdi],al
 dec rdi
 inc rsi
 loop loop2
print m3,l3
 print string2,[length]
 mov rax,60
 mov rdx,0
 syscall
```

<u>Output:</u>


Assembly
Run
Download
Fork
Output
Shutterstock Free Trial - Get images, video, music & easy to use design tools with one subscription.
ADS VIA CARBON

```
%macro print 2
mov rax,1 ; Function 1 - write
 mov rdi,1 ; To stdout
 mov rsi,%1 ; String address
 mov rdx,%2 ; String size
syscall ; invoke operating system to WRITE
%endmacro
%macro read 2
mov rax,0 ; Function 0 - Read
 mov rdi,0 ; from stdin
 mov rsi,%1 ; buffer address
 mov rdx,%2 ; buffer size
 syscall ; invoke operating system to READ
%endmacro
section .data
 m1 db 10d,"Enter String: "
 l1 equ $-m1
 m2 db 10d,"Length of String: "
 l2 equ $-m2
 m3 db 10d,"Reversed String: "
 l3 equ $-m3
section .bss
 string resb 50
 string2 resb 50
 length resb 16
 answer resb 16

section .text
```

```
riteshjawale
```

```
Enter String:
Length of String: 000000000000000C
Reversed String: elawajhsetir
[Execution complete with exit code 1]
```