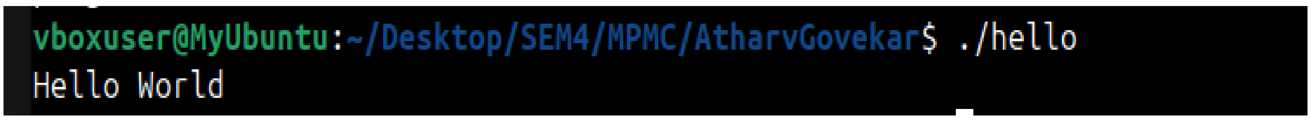
1]write an alp to display Hello world on the screen

Input:

```
SECTION .DATA

hello db 'Hello World', 10

hellolen equ $ - hello


SECTION .TEXT

GLOBAL _start


_start:

   mov eax, 4

   mov ebx, 1

   mov ecx, hello

   mov edx, hellolen

   int 80h


   mov eax, 1

   xor ebx, ebx

   int 80h
```

Ouput :



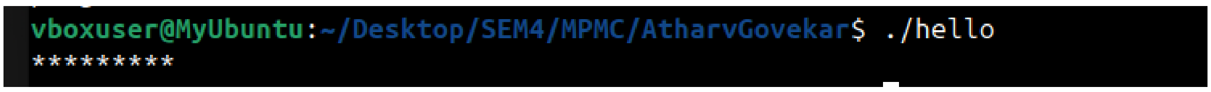
```
vboxuser@MyUbuntu:~/Desktop/SEM4/MPMC/AtharvGovekar$ ./hello
Hello World
```

2]write an alp to display nine stars on the screen using loop

Input:

```
section .data
    stars times 9 db '*'
    newline db 10
section .text
    global _start
_start:
    mov eax, 4
    mov ebx, 1
    mov ecx, stars
    mov edx, 9
    int 0x80

    mov eax, 4
    mov ebx, 1
    mov ecx, newline
    mov edx, 1
    int 0x80

    mov eax, 1
    xor ebx, ebx
    int 0x80
```

Ouput :



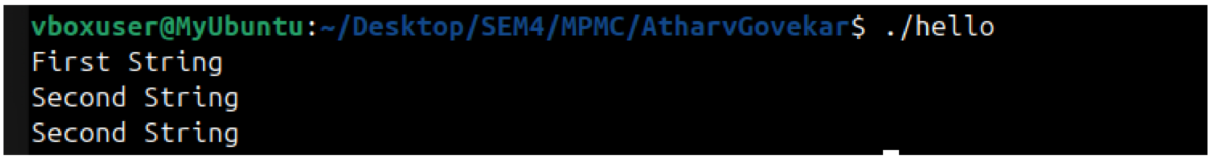
```
vboxuser@MyUbuntu:~/Desktop/SEM4/MPMC/AtharvGovekar$ ./hello
*********
```

3]write an alp to display two strings on the screen using EQU directory

Input:

```
SECTION .DATA
str1 db 'First String', 10
str2 db 'Second String', 10
str1len equ $ - str1
str2len equ $ - str2

SECTION .TEXT
GLOBAL _start

_start:
    ; Print the first string
    mov eax, 4
    mov ebx, 1
    mov ecx, str1
    mov edx, str1len
    int 80h

    ; Print the second string
    mov eax, 4
    mov ebx, 1
    mov ecx, str2
    mov edx, str2len
    int 80h

    ; Exit the program
    mov eax, 1
    xor ebx, ebx
    int 80h
```

Ouput :



```
vboxuser@MyUbuntu:~/Desktop/SEM4/MPMC/AtharvGovekar$ ./hello
First String
Second String
Second String
```

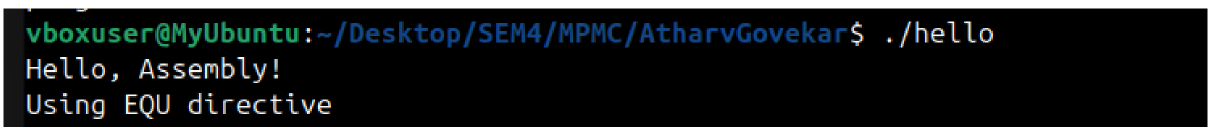
4]write an alp to display given word in a string on the screen

Input:

```
section .data
   LF        EQU    10
   NULL      EQU    0
   SYS_WRITE  EQU    4
   SYS_EXIT   EQU    1
   STDOUT     EQU    1

   msg1      db     'Hello, Assembly!', LF, NULL
   msg1_len   EQU    $ - msg1 - 1

   msg2      db     'Using EQU directive', LF, NULL
   msg2_len   EQU    $ - msg2 - 1

section .text
   global _start

_start:
   mov eax, SYS_WRITE
   mov ebx, STDOUT
   mov ecx, msg1
   mov edx, msg1_len
   int 0x80

   mov eax, SYS_WRITE
   mov ebx, STDOUT
   mov ecx, msg2
   mov edx, msg2_len
   int 0x80
```

```
mov eax, SYS_EXIT

xor ebx, ebx

int 0x80
```

Ouput :



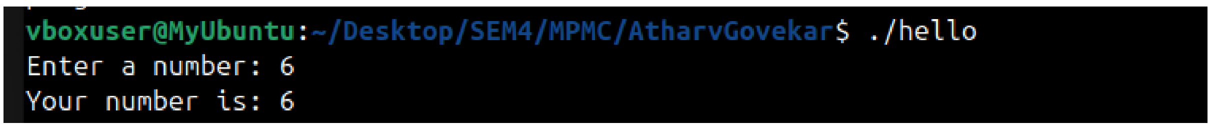
```
vboxuser@MyUbuntu:~/Desktop/SEM4/MPMC/AtharvGovekar$ ./hello
Hello, Assembly!
Using EQU directive
```

5]write an alp that reads a number from a keyboard and displays on the screen

Input:

```
section .data
    prompt db 'Enter a number: '
    plen equ $ - prompt
    msg db 'Your number is: '
    mlen equ $ - msg

section .bss
    num resb 5

section .text
    global _start

_start:
    mov eax, 4
    mov ebx, 1
    mov ecx, prompt
    mov edx, plen
    int 0x80

    mov eax, 3
    mov ebx, 0
    mov ecx, num
    mov edx, 5
    int 0x80

    mov eax, 4
    mov ebx, 1
    mov ecx, msg
    mov edx, mlen
    int 0x80

    mov eax, 4
    mov ebx, 1
    mov ecx, num
    mov edx, 5
    int 0x80

    mov eax, 1
    xor ebx, ebx
    int 0x80
```

Ouput :



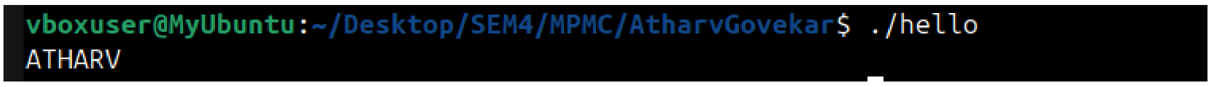
```
vboxuser@MyUbuntu:~/Desktop/SEM4/MPMC/AtharvGovekar$ ./hello
Enter a number: 6
Your number is: 6
```

6]write an alp prints name on the screen

Input:

```
SECTION .DATA
name db 'ATHARV', 10
namelen equ $ - name

SECTION .TEXT
GLOBAL _start

_start:
   mov eax, 4
   mov ebx, 1
   mov ecx, name
   mov edx, namelen
   int 80h

   mov eax, 1
   xor ebx, ebx
   int 80h
```

Ouput :



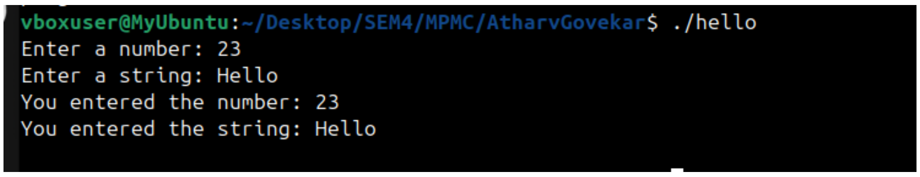
```
vboxuser@MyUbuntu:~/Desktop/SEM4/MPMC/AtharvGovekar$ ./hello
ATHARV
```

7]write an alp to accept a number and a string and diplay it on the screen

Input :

```
SECTION .bss
   num: resb 10
   str: resb 100

SECTION .data
   prompt_num db "Enter a number: "
   prompt_num_len equ $ - prompt_num
   prompt_str db "Enter a string: "
   prompt_str_len equ $ - prompt_str
   output_num db "You entered the number: "
   output_num_len equ $ - output_num
   output_str db "You entered the string: "
   output_str_len equ $ - output_str
   newline db 10

SECTION .text
   global _start

_start:
   ; Print prompt_num
   mov eax, 4
   mov ebx, 1
   mov ecx, prompt_num
   mov edx, prompt_num_len
   int 0x80

   ; Read input for num
   mov eax, 3
   mov ebx, 0
mov ecx, num
mov edx, 9
int 0x80

; Print prompt_str
mov eax, 4
mov ebx, 1
mov ecx, prompt_str
mov edx, prompt_str_len
int 0x80

; Read input for str
mov eax, 3
mov ebx, 0
mov ecx, str
mov edx, 99
int 0x80
push eax

; Print output_num
mov eax, 4
mov ebx, 1
mov ecx, output_num
mov edx, output_num_len
int 0x80

; Print num
mov eax, 4
mov ebx, 1
   mov ecx, num
   mov edx, 9
   int 0x80

   ; Print output_str
   mov eax, 4
   mov ebx, 1
   mov ecx, output_str
   mov edx, output_str_len
   int 0x80
   ; Print str
   mov eax, 4
   mov ebx, 1
   mov ecx, str
   pop edx
   int 0x80
   ; Print final newline
   mov eax, 4
   mov ebx, 1
   mov ecx, newline
   mov edx, 1
   int 0x80
   ; Exit program
   mov eax, 1
   mov ebx, 0
   int 0x80
```

Ouput :



```
vboxuser@MyUbuntu:~/Desktop/SEM4/MPMC/AtharvGovekar$ ./hello
Enter a number: 23
Enter a string: Hello
You entered the number: 23
You entered the string: Hello
```