**COA ASSIGNMENT NO. :- 3**

**Aim:-** Write a 64-bit ALP to “Hello World” in NASM.

**Objective:-** To print “”Hello World” in NASM using assembly language programming.

**Theory :-**

The first important document is [the HYPERLINK "https://github.com/hjl-tools/x86-psABI/wiki/X86-psABI"x86-64 ABI specification](https://github.com/hjl-tools/x86-psABI/wiki/X86-psABI), maintained by Intel. (Weirdly, the official location for the ABI specification is some random dude’s personal GitHub account. Welcome to the sketchy world of assembly.) The ABI specification describes system calls in the abstract, as it applies to any operating system. Importantly:

* The system call number is put in rax.
* Arguments are put in the registers rdi, rsi, rdx, rcx, r8 and r9, in that order.
* The system is called with the syscall instruction.
* The return value of the system call is in rax. An error is signalled by returning -errno.

The second document is [the Linux 64-bit system call table](https://github.com/torvalds/linux/blob/master/arch/x86/entry/syscalls/syscall_64.tbl). This specifies the system call number for each Linux system call. For our example, the write system call is 1 and exit is 60.

Finally, you want the man pages for the system calls, which tell you their signature, e.g.:

#include <unistd.h>

ssize\_t write(int fd, const void \*buf, size\_t count);

Armed with this, we know to:

* put the system call number 1 in rax
* put the fd argument in rdi
* put the buf argument in rsi
* put the count argument in rdx
* finally, call syscall

**Sourcecode :-**

section .data

m1 db "Hello World!!"

l1 equ $-m1

section .text

global \_start

\_start:

mov rax, 1 ; system call for write

mov rdi, 1 ; file handle 1 is stdout

mov rsi, m1 ; address of string to output

mov rdx, l1 ; number of bytes

syscall ; invoke operating system to do the write

mov rax, 60 ; system call for exit

mov rdi, 0 ; exit code 0

syscall ; invoke operating system to exit

**Output:-**

user@user-Nitro-AN515-52:~/Desktop$ nasm -f elf64 -o hello.o hello.asm

user@user-Nitro-AN515-52:~/Desktop$ ld hello.o -o hello

user@user-Nitro-AN515-52:~/Desktop$ ./hello

Hello World!!