Question 2:

**Implementation:**

The system call is named sh\_task\_info(), for adding the system call I have added the line

(336 64 sh\_task\_info sys\_sh\_task\_info) to the syscall\_64.tbl file present in the directory whose path from the root directory is given by arch/x86/entry/syscalls.

Next I defined the system call inside the sys.c file present in the directory whose path form the root is given by /kernel.

The system call takes in 2 arguments thus I have written SYSCALL\_DEFINE2, furthermore I have added the error handling for less number of arguments or invalid arguments. After this the PID passed is used to find the task related to it using tow function find\_get\_pid and pid\_task then the path of the file is copied into a variable actualPath using strncpy\_from\_user as the memory space of the user space and the kernel space is different. After that the corresponding file is opened (created or possibly truncated, if it already exists).

Finally the following is written to the file using kernel\_write and the corresponding values are also printed using printk(can be seen dmesg | tail):

1)PID

2)Name

3)State

4)Processor used

5)Last Wakee’s name and PID

6) Priority

7)Parent’s name and PID

After the write is over the files are closed.

**Inputs and Outputs:**

When the executable of the compiled C program is run, it requires two inputs the first one is the PID of the process to be searched for and the second argument is the path/name of the file to be written to.

Example: ./runSysCall 2 Desktop/output.txt

./runSysCall 5 Desktop/output1.txt

The output would be the value returned by the system call, if the value is negative an error has occurred and the perror in the C program would display the error message and in using dmesg | tail you can see the actual error that occurred.

If the execution is successful, the attributes of the process would be written in the file specified, which can be seen using the cat command.

Example: cat output.txt

cat output2.txt

**Error values:**

EINVAL : This error occurs when the user provides insufficient arguments or invalid arguments

ENAMETOOLONG : This error occurs when the filename or path is more than 256 characters long

ESRCH: This error occurs when the task having the provided PID does not exist

EFAULT: This error occurs when the strncopy\_from\_user is not able to copy the string form the user space address into the kernel space memory address

EBADF: This error occurs if the filp\_open cannot open the file specified

EIO : This error occurs when the kernel\_write function is not able to write some attribute in the file specified.