# Section 2: Operating System Overview

## Question 1

Describe the difference between API and ABI services / functions.

Describe the difference between write() and println() as described in class.

### Answer

The Application Programming Interface (API) are libraries provided by the OS to perform common functions including process management, I/O, and others. Just like the user programs, API functions do not access the kernel directly but use the ABI.

The Application Binary Interface (ABI) are the functions used to invoke kernel services. ABI functions are primitive operations that a program can request of the OS e.g. open(), close(), read(), write(), seek() are operations against streams which are typically files but may be used to manipulate sockets and other types of I/O.

For example write() is an ABI System Call (aka SYSCALL) that writes (sends) a buffer of data to a file, socket, or other I/O device. Write() is a fundamental (primitive) service (write raw data) provided by the OS. The API call println() will format and write characters to a file, socket, etc. Println() will generate a string from its input parameters which is written. Println() is a convenience function and is built on top of the write() syscall.

## Question 2

Explain why a program’s execution must be ‘blocked’ when requesting certain operating system services i.e. executing a Blocking System Calls.

What is meant when a process is in a ‘blocked’ state?

### Answer

Some OS services (primarily input / output) require many processor cycles to complete e.g. reading a block of data from a drive requires many milliseconds. Other services are designed to wait for some external event to occur. For example, a process blocks while it waits for messages to arrive across a ‘socket’ network connection.

During these periods the OS does not schedule the execution of “blocked” process. Blocked processes will not be scheduled for processor execution until the blocking syscall completes or the needed event arrive. Blocking the execution of these processes provides additional processor cycles for other processes that are ‘ready’ or have work to do.

## Question 3

1. Name the two processor modes described in this section?
2. Describe the two restrictions placed on program execution when the processor is executing in user mode.
3. What triggers the processor to switch to the ‘kernel mode’ state?

### Answer

1. User Mode and Kernel Mode.
2. In user mode, the processor’s execution is restricted to memory addresses within the range of memory allocated to the currently executing program (process). If a user process attempts to address memory outside of its allocated range, the memory management hardware will generate a processor interrupt halting the execution of the offending process. Also in user mode, the processor is unable to execute its privileged instructions. Privileged instructions are those that are needed to perform process management (context switching) and access protected I/O devices (disks).

In kernel mode, the processor is able to access both kernel space memory and memory owned by user processes. Privileged instructions can only be executed in kernel mode.

1. Any type of interrupt (hardware or software) causes the process to switch to kernel mode.

## Question 4

1. Describe the purpose of the memory regions shown in Slide “Multiple Processes Maintained in Memory” i.e. Process Control Blocks / Process A & B.
2. Describe the purpose of the ‘Process Index Register”.

### Answer

1. Process Control Blocks are a list of processes that are currently active in the system. The PCB region is owned by the operating system (in kernel space), so cannot be accessed by the user programs. The Process List includes a reference to the start of the Process in user memory. The Process Context containing information used by the OS to manage the Process.  
     
   The regions Process i & j are the regions of system memory that have been allocated to these two processes. Each of these regions maintain the instructions, data, and control stacks (not shown) owned by the process.
2. The Process Index Register is a kernel data structure that identifies the currently executing process in the PCB list.

## Question 5

Describe the purpose of Base and Limit registers in Slide “Multiple Processes Maintained in Memory”.

### Answer

The Base register contains the start of the current process in memory and Limit contains the amount (length) of the memory allocated to the process. The memory management hardware will trigger an interrupt that will terminate the executing process if the process attempts to access memory outside the range defined by the base and limit registers.

## Question 6

What is the name of the mechanism used by a process to request a service from the operating system?

What are the four steps the processor executes when responding to a process’s service request in the Linux OS (from the slides)?

### Answer

The mechanism is called a system call or syscall.

The four steps in making a syscall are:

1. The process places the integer syscall number that identifies the OS request and its arguments into specific processor registers as defined by the syscall implementation.
2. The process executes a software interrupt (INT) instruction. The processor switches to Kernel Mode and saves the processor’s state on the kernel stack.
3. The interrupt handler uses the syscall number (found in the register) to determine the OS service being requested and executes the request-handling instructions in kernel mode in the kernel’s regions of memory.
4. When the syscall handler completes the system call, the last instruction it executes is the IRET instruction which restores the processor state and returns the process to User Mode where it continues execution.

## Question 7

Name and describe the three categories of information described in a process’s control block.

### Answer

Process Identity: A unique process identifier that is assigned by the system when the process is created.

Process State: Defines the process state including the process’s *state* (blocked, running, etc.), execution priority, Owner ID, Parent Process ID, and other OS house-keeping information. The process state also include the OS resources (buffers, files, sockets, windows, etc.) that have been allocated to the process.

Processor State: Defines the processor state that exists when the process’s execution is interrupted including register contents and other information that allows the processor state to be restored when the process is next scheduled for execution (running).

## Question 8

Describe the four innovations in computer systems design that make modern operating systems possible.

### Answer

1. Protected Memory: The OS kernel is loaded into a region of memory that is not accessible by the user’s program i.e. can only be accessed when the processor is in kernel mode.
2. Privileged Instructions: Processor instructions that manage processor hardware and other sensitive areas of operations. Privileged instructions can only be executed when the processor is in kernel mode.   
   Points 1 & 2 keeps faulty or malicious user programs from corrupting the state of the computer system or stealing sensitive protected data.
3. Interrupts: The ability to interrupt a program’s execution to process (handle) asynchronous events such as scheduling, fault handling, managing I/O, etc.
4. Hardware Timers: A periodic hardware interrupt that allows the kernel to take control of the processor. Used to manage and schedule user programs (processes) by ‘interrupting’ the execution of the currently executing program and installing the next ready program for execution.

## Question 9

1. Describe the principle of Preemptive Multitasking.
2. What are two triggers / methods of preempting a program’s execution described in the slides?

### Answer

Preemptive multitasking is a technique for allowing multiple programs / processes to share a single processor. That is, to allow multiple processes to take turns executing on the processor. A kernel service is responsible for scheduling each of the processes for execution. This is accomplished by periodically suspending the execution of the currently running process and resuming the execution of the next process for a slice of processor time. This is repeated giving every eligible process some amount of execution time.

The kernel blocks (pre-empts) the currently executing process when 1) the process makes a blocking system call or 2) when a timer interrupt occurs. Note: Understand the purpose and use of the Timer and Timer Interrupts for the exam.

## Question 10

Note: The materials for this question are found in the text book.

1. Describe ‘Modules’ in the Linux Operating System.
2. What are two important characteristics of Linux modules given in the text book?

### Answer

1. A module is an object code file (containing processor instructions) that can be loaded into the executing operating system. Modules allow new services to be added to the system without the need to recompile the kernel or even restart a running system. For example, new devices (disk, network, etc.) can be added to the system by loading the device’s driver as a module.
2. Dynamic Linking: A module can be loaded and installed (linked) into kernel memory without the need to recompile or restart the system. Modules can also be unloaded.

Stackable Modules: Modules can be arranged in a hierarchical (tree) structure of clients and servers. M1 can be loaded and provide services to client modules M2 & M3.