Respuestas Orientadoras e incompletas (en algunos casos) a las preguntas de revisión. Favor tomarlas como orientadoras.

Clase 3 – Revisión

1. ¿Cuáles son las características de un proceso?

-Programa en ejecución
-"espíritu animado" de un programa
-entidad asignada a CPU y ejecutada por ésta
-instancia de programa en ejecución
-unidad de actividad caracterizada por la ejecución de una secuencia de instrucciones, un estado y un conjunto de recursos asociados
-entidad formada por n elementos (codigo del programa, conjunto de datos, PCB (process control block))
-entidad que describe un comportamiento o taza, listado de la secuencia de instrucciones del proceso

2. ¿Cuáles son los componentes de un proceso?

-Programa ejecutable
-Datos asociados al programa (variables, espacio de trabajo, buffers, etc.)
-Contexto de Ejecución: info. necesaria para que el SO pueda administrar el proceso, contenido de los registros del procesador, PC, registros de datos, prioridad, etc.

3. Teniendo en cuenta las dos preguntas anteriores y sumado al hecho de la existencia de sistemas con multiprogramación, ¿Qué problemas podrán causar n procesos ejecutándose al mismo tiempo en una misma CPU?

-Sincronización Incorrecta: mal manejo de interrupciones, diseño incorrecto de mecanismo de señalización.
-Fallos de Exclusión mutua: problemas de concurrencia: mecanismo de exclusión para que solo uno por vez pueda tomar el recurso.
-Funcionamiento no determinista del programa: la salida debe depender de la entrada y no de las actividades realizadas por otros programas.
-Interbloqueos: dos o más programas suspendidos, uno a la espera del otro (abrazo mortal), por ejemplo con asignación de dos dispositivos de E-S, forma imprevista de asignación y liberación de recursos.

4. ¿Qué ventajas cree Ud. que acarrea el hecho de trabajar con memoria virtual?

Permite a los programas direccionar memoria desde el punto de vista lógico, sin importar la cantidad de memoria principal física disponible, permite n procesos concurrentes. Las referencias se manejan por medio de una dirección virtual (a través de un sistema de paginación de memoria, se referencia como un número de página y un desplazamiento) en donde el sistema de paginación proporciona una proyección a una dirección real o física de la memoria principal.
Las páginas de un proceso en ejecución se guardan en el disco y sólo se tienen en memoria las mínimas requeridas para la ejecución del proceso, cuando se referencia una dirección que no esta en memoria principal, el hardware de gestión de memoria lo detecta y permite que la página que falta se cargue; a esto se lo llama área de memoria virtual.
La CPU y el SO dotan al usuario de un "CPU virtual" que accede a memoria virtual intercalando un traductor de direcciones (memory management unit, MMU, mapper) entre la CPU y la memoria principal.

5. ¿Qué diferencia hay entre un hilo (thread) y un proceso?

Un proceso es una unidad de asignación de recursos y de protección, la cual puede ser implementada bajo un modelo de un único hilo o bajo un modelo multihilo.

Cada hilo tiene su información contextual (BCP), su pila o stack, pero comparte el BCP del proceso al que pertenece el hilo y tiene acceso al espacio de direcciones del proceso común a todos los hilos, ver Fig. 4.2 Pag. 160.
Cada hilo tiene: su estado de ejecución, contexto, pila, espacio de direcciones para variables locales, acceso a memoria y recursos del proceso, compartido por todos los hilos de su mismo proceso.

6. ¿Qué diferencia hay entre la arquitectura micro-núcleo (microkernel) y la arquitectura multi-hilo (multithreading)?

Una Arquitectura micronúcleo (microkernel) se refiere a una decisión de diseño que se debe tomar en todo SO moderno, la idea es asignar sólo unas pocas funciones escenciales al núcleo del SO que permanecerá en memoria, tales como: -manejo de espacio de direcciones de memoria, comunicación entre procesos, planificación básica de procesos. Esta decisión de diseño está en contraposición con la aquitectura monolítica. La arquitectura multihilo (multithreading) se refiere a la habilidad de un SO para soportar múltiples y concurrentes flujos de ejecución dentro de un mismo proceso, tal como es el caso de windows y los unix modernos (linux, solaris, etc). Sin embargo, los antiguos unix soportaban n procesos concurrentes pero un solo hilo de ejecución por cada proceso. Se puede o no hacer uso de multihilo dentro de una arquitectura microkernel o monolítica, dependiendo si el SO soporta o no multihilos.

7. Responda Verdadero o Falso:
   En un sistema de multiprocesamiento simétrico (SMP):
   * existen n procesadores cada uno con su propia memoria (F)
   * existen 1 solo procesadores que comparte su memoria con n hilos o threads (F)
   * existen n procesadores que permiten ejecutar distinto set de instrucciones (F)
   * existen n procesadores que permiten ejecutar el mismo set instrucciones (V)
   * un procesador para poder ejecutar una instrucción debe esperar a que los otros procesadores no estén ejecutando ninguna instrucción (F)
8. Suponga que un usuario UNIX está usando la aplicación *vi* (editor) modificando un archivo de texto. Enumere las capas que se encuentran entre este usuario y el hardware de su computador.

1. Unix commands and Libraries (vi)
2. System call interface
3. Kernel
4. Hardware

9. ¿Por qué razón Ud. cree que los diseñadores han optado por un modelo por capas? ¿Qué ventajas cree Ud. que trae el programar de esta forma? ¿Podemos utilizar esta idea en la construcción de software aplicativo?

El desarrollo de software por capas requiere de capas de software con un objetivo específico e interfases bien definidas, toda capa de software debe estar *especificada*. Esto facilita el trabajo en grupo, en donde desarrolladores se concentren en determinada capa, desconociendo el resto, bajando el nivel de complejidad y acelerando la curva de aprendizaje. Las capas pueden ser provistas por distintos proveedores, pueden corregirse, actualizarse (hacia su interior) y hasta incluso ampliarse, sin interferir con el resto (encapsulación). Esta forma de desarrollo puede aplicarse a cualquier tipo de software.

10. Linux no utiliza la arquitectura de micro-núcleo (microkernel), describa brevemente cuál es la idea de su implementación en contraposición con la arquitectura de un UNIX tradicional.

Linux no tiene un diseño microkernel sino una aquitectura modular, se trata de una colección de módulos que se cargan en tiempo de ejecución (runtime) y se pueden vincular y desvincular (también en runtime) del kernel a través de los comandos *insmod , rmmod*. Los módulos son apilab les (stackable) de forma jerárquica. De forma tal, que un modulo puede servir como librería de otro módulo cliente. La idea es que un módulo pueda implementar el código común a n módulos, de esta

forma, se reduce la cantidad de código que el SO debe mantener en memoria, no hay duplicación de código, facilita el desarrollo de módulo más específicos, permite definir las dependencias entre módulos y el kernel esta seguro de contar con todos los módulos requeridos.