1. **Responde a las siguientes preguntas**

1.1. ¿En un sistema monotarea, cuántos procesos diferentes al SO están a la vez en la memoria?

Sólo uno, por eso mismo se llama MONOtarea.

1.2. ¿Qué diferencia existe entre un programa y un proceso?

Que un proceso es un programa en ejecución.

1.3. Basado en la respuesta anterior, ¿Puede existir un proceso antes de su programa?

1.4. ¿Qué número identifica de forma única a un proceso respecto al SO? Indica la sigla y su significado

En computación, PID es una abreviatura de process ID, o sea, Process Identification o bien identificador de procesos. El identificador de procesos es un número entero usado por el kernel de algunos sistemas operativos (como el de Unix o el de Windows NT) para identificar un proceso de forma unívoca.

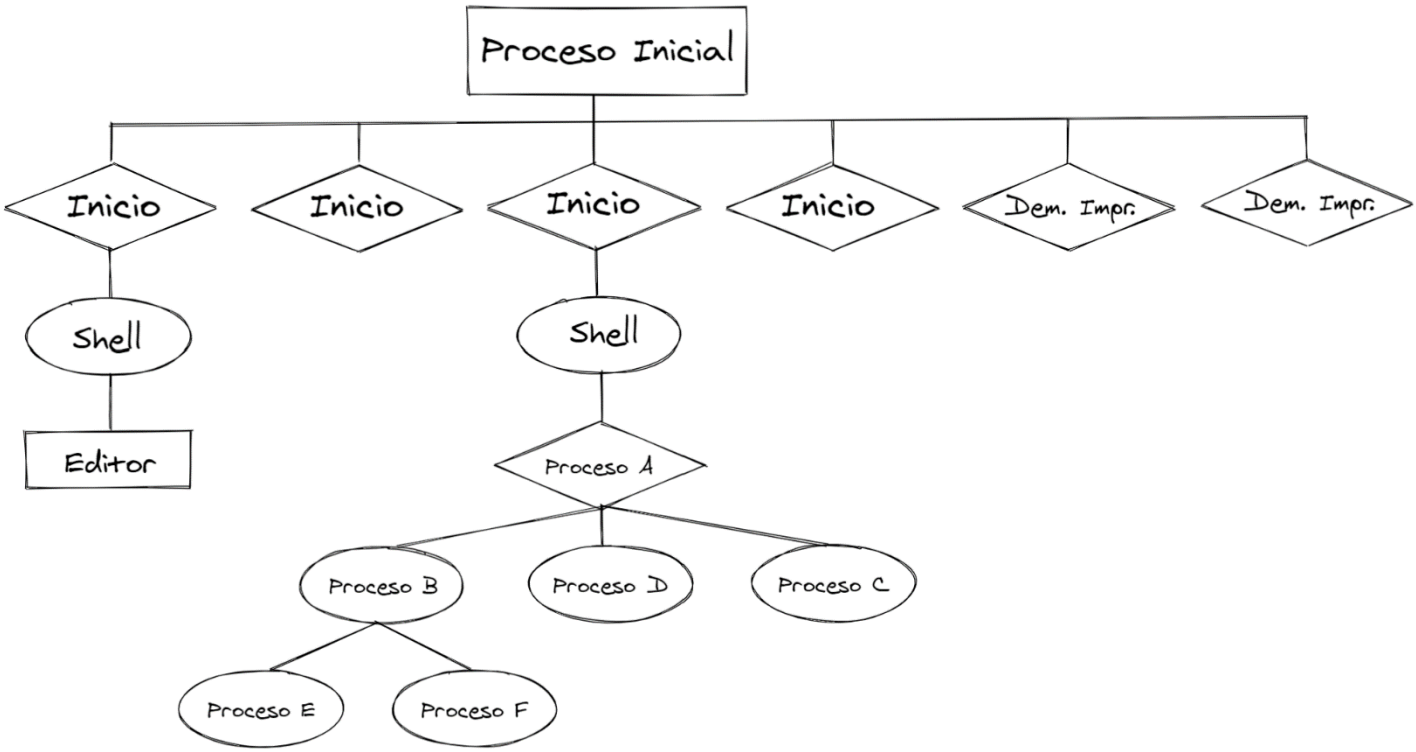
Para asignar el PID, el kernel utiliza internamente una variable global que se va incrementando con cada nuevo proceso creado con una llamada fork(). Cuando la variable alcanza un cierto valor límite se empieza otra vez desde 0, buscando números que no estén asignados ya a otro proceso en ejecución.

El valor 0 del PID está reservado a la tarea ociosa del sistema, un proceso que se crea al arrancar el sistema y nunca puede finalizar.

1.5. ¿Qué significa la jerarquía de procesos en un SO?
En este enlace tenéis más información. Haz un gráfico de este concepto (excalidraw)

En un sistema lo suficientemente sencillo es posible que todos los procesos que podrían ser necesarios en algún momento pueden estar presentes durante la inicialización del sistema, pero en la mayoría de los sistemas, es necesario una forma de crear y destruir procesos, cuando se requiera durante la operación.

Esta jerarquía muestra cinco niveles: nivel macroproceso, nivel proceso, nivel subproceso, nivel actividades y nivel de tareas específicas a realizar en un proceso concreto.



1.6. Los SO operativos multitarea tienen muchos procesos ejecutándose, pero, ¿cómo un SO es capaz de ejecutar múltiples procesos a la vez con un solo procesador?. Razona tu respuesta.

Porque en realidad los procesos NO se están ejecutando a la vez.
La simultaneidad es la sensación que la gran velocidad del procesador crea en la percepción humana.
Los procesos se ejecutan uno detrás de otro, en un orden previamente definido por un algoritmo (Planificador de Procesos).

2. **Responde a las siguientes preguntas sobre Gestión de Procesos**

2.1. ¿Cuándo hablamos de que un proceso muere de inanición, o se bloquea por este motivo, a qué nos estamos refiriendo?

En informática, inanición (starvation en inglés) es un problema relacionado con los sistemas multitarea, donde a un proceso o un hilo de ejecución se le deniega siempre el acceso a un recurso compartido. Sin este recurso, la tarea a ejecutar no puede ser nunca finalizada.

La inanición no es sinónimo de interbloqueo, aunque el interbloqueo produce la inanición de los procesos involucrados. La inanición puede (aunque no tiene porqué) acabar, mientras que un interbloqueo puede finalizar con una acción del exterior.

2.2. ¿Cómo se llama el componente del sistema operativo encargado de la gestión de los procesos?

El Planificador de Procesos.

2.3. ¿Este componente del SO es en sí mismo un proceso más, que se rige por las mismas reglas que los demás procesos? Razona tu respuesta

NO, es un proceso más cualquiera.

Aunque tenga la consideración de un proceso, es un proceso especial, ya que al iniciar el SO, éste coge todo el uso de la Memoria para luego administrárselo a los demás procesos en función de los programas que vayamos ejecutando y lo que vayamos haciendo con ellos.

2.4. Enumera los distintos estados de un proceso, e incluye una pequeña descripción de cada estado.

Los procesos pueden pasar a 3 estados diferentes:

- Listo.
- En ejecución.
- Bloqueado.

- Los procesos en estado listo son los que pueden pasar a estado de ejecución si el planificador del sistema operativo los selecciona, esto es, cuando llegue su turno (según el orden de llegada o prioridad).
- Los procesos en estado de ejecución son los que se están ejecutando en el procesador en un momento dado.
- Los procesos que se encuentran en estado bloqueado están esperando la respuesta de algún otro proceso para poder continuar con su ejecución, por ejemplo, una operación de entrada/salida.

2.5. Indica cuáles son los distintos estados de un proceso, y dibuja un gráfico de estados, incluyendo desde el inicio hasta el fin de un proceso.

2.6. ¿Cómo se denomina cuando un proceso pasa de estado en-ejecución a listo o viceversa?

En un sistema multiproceso o multihebra, cuando un proceso o hilo pasa de un estado a otro (por ejemplo, de espera a ejecución), lo que se producirá es un cambio de contexto.

El cambio de contexto puede ser parcial si se realiza entre hilos del mismo proceso. En caso de que el cambio de contexto sea entre hilos de diferentes procesos, se producirá un cambio de contexto completo, ya que el cambio afectará a memoria, hardware, ficheros comunes, etc.

2.7. Cuando se produce este hecho, el SO debe guardar una serie de información del proceso para poder volver a "revivirlo" en el mismo estado que antes de la congelación, ¿Qué nombre recibe este conjunto de datos?

Toda la información de un proceso que el SO necesita controlar se mantiene en una estructura de datos denominada Bloque de Control de Proceso (BC)

2.8. ¿Qué datos se deben guardar de un proceso para que se produzca el cambio de contexto sin afectar para nada al funcionamiento del proceso?

El BC almacena la siguiente información:

- Estado actual del proceso: Ejecución, preparado o bloqueado.
- Identificador del proceso (PID): Dependiendo del SO, a cada proceso se le asigna un PID.
- Prioridad: La asignada por el planificador.
- Ubicación en memoria: Dirección de memoria en la que se carga el proceso.
- Recursos utilizados: Recursos hardware y software para poder ejecutarse.

2.9. Para guardar toda esta información de cada proceso, y tener a todos los procesos controlados, el SO utiliza una estructura para almacenarlos ¿Cómo se llama esta estructura?

El sistema operativo almacena en una tabla denominada tabla de control de procesos con la información relativa a cada proceso que se está ejecutando en el procesador.

2.10. ¿Qué elementos se guardan en cada posición de esa tabla para tener control de cada proceso?

En cada posición de la tabla se guarda un Bloque de Control (BC).

2.11. Define los conceptos planificación apropiativa y no apropiativa.

Planificación Apropiativa	El sistema operativo puede arrebatar el uso de la CPU a un proceso que esté ejecutándose.
Planificación No Apropiativa	Cuando a un proceso le toca su turno de ejecución, ya no puede ser suspendido; es decir, no se le puede arrebatar el uso de la CPU (hasta que el proceso no lo determina no se podrá ejecutar otro proceso).

2.12 Enumera en la siguiente tabla los algoritmos de planificación de procesos vistos en clase.

Algoritmo	**Descripción**
de rueda	Asigna rotativamente tiempos de ejecución a los diferentes procesos. y en él la asignación de tiempos de ejecución a los procesos es la misma y de forma secuencial. A cada uno se le asigna el mismo quantum o intervalo de tiempo de ejecución.
FIFO (First In First Out) o FCFS (First Come First Service)	Al primer proceso que llega se le asignan tiempos o ciclos de CPU hasta que termina completamente. A continuación, se ejecuta completo el siguiente proceso que hay en la cola FIFO y así sucesivamente hasta terminar con el último proceso.
STR (Short Time Remainder)	Este algoritmo permite asignar el tiempo de ejecución de forma prioritaria a los procesos más cortos con el fin de ejecutarlos en el menor tiempo posible.
SRTF (Shortest Remaining Time First)	Variedad del STR donde la asignación de ciclos de CPU va en función del proceso que tenga menos ciclos pendientes de terminar.

3. Hebras.

3.1. ¿Qué diferencia existe entre un proceso y una hebra? Busca información en Internet para aclarar mejor ambos conceptos.

Un proceso es un programa en ejecución, y una hebra es un punto de ejecución de un proceso.

La diferencia en sí radica en que las hebras se ejecutan dentro del mismo espacio de memoria, mientras que los procesos se pueden ejecutar desde diferentes espacios de memoria.

3.2. ¿Cuántas hebras como mínimo puede tener un proceso? ¿Existe un máximo?

Puede tener como mínimo una hebra (el proceso actual que ejecuta el programa en sí), y el máximo dependerá de cómo esté hecho el programa, si éste está capacitado para aprovechar más recursos o no.

3.3. Un solo proceso puede estar a la vez en la CPU, pero si un ordenador tiene por ejemplo 4 cores, ¿Sigue teniendo esta limitación? Razona tu respuesta.

No, ya no tendría esa limitación, ya que podría ejecutar 4 procesos con 4 cores.

Los cores son, como un subprocesador en sí mismo.

Como un núcleo es un procesador en sí mismo, una CPU multinúcleo de dos núcleos pueda ejecutar dos tareas al mismo tiempo.

Una CPU con dos núcleos sí que podría realizar dos tareas al mismo tiempo, pero no más. Uno de cuatro, pues cuatro, y así de forma correlativa con tantos núcleos como incorpore.

Se puede definir como el flujo de control de programa.

Aunque un núcleo solamente pueda realizar una tarea al mismo tiempo, se pueden usar los hilos para hacer creer al usuario (y al propio ordenador) que sí se puede hacer más de una cosa al mismo tiempo.

En vez de realizar una tarea por completo, se divide la tarea en porciones (cada hilo se encarga de un aspecto concreto del programa), de modo que se va alternando entre porciones de tareas para que parezca que ambas se ejecutan al mismo tiempo.

Hacemos un poco de un proceso y otro poco de otro proceso; cada uno de esos trozos se corresponde con el hilo. Así, no tenemos que esperar a que una tarea acabe para comenzar otra.

El número de hilos corresponde de manera directa con el número de tareas que se pueden llevar a cabo de forma pseudoparalela (es decir, de forma ‘simultánea’).