### **Intro/procesos**

-¿Qué sucedería si llegara una interrupción durante el ciclo de ejecución de una instrucción? ¿En qué momento sería atendida? Indique también un ejemplo donde la misma sería ignorada y otro en la que no.

-¿Por qué se dice que realizar una syscall tiene un costo asociado en comparación con no realizar dicha llamada? ¿Tendría sentido permitir que algunas operaciones normalmente realizadas por syscalls puedan resolverse en modo usuario?.

-¿Cuál de las siguientes ventajas corresponden a procesos por sobre KLTs?

| 1. Permiten el multiprocesamiento ya que son administrados por el SO 2. Su creación e intercambio son más livianos 3. De forma inherente se pueden comunicar por tener memoria compartida 4. Son más estables y seguros 5. Todas las ventajas citadas son correctas 6. Ninguna de las ventajas citadas es correcta |
| --- |
| Correcta: D |

-¿Cuál de las siguientes afirmaciones es correcta sobre ULTs?

| 1. Los ULTs de distintos procesos siempre utilizan el mismo algoritmo de planificación 2. En caso de usar jacketing equivale a usar KLTs 3. Es más liviana su creación y switches y permite utilizar un algoritmo de planificación distinto al del SO 4. Al volver a ser elegido un KLT que había sido desalojado por el SO, su biblioteca de hilos siempre puede replanificar y elegir qué ULT ejecutar 5. Todas las afirmaciones sobre ULTs citadas son correctas 6. Ninguna afirmación sobre ULTs citada es correcta |
| --- |
| Correcta: C |

-Suponga un sistema que utiliza un planificador por prioridades (con desalojo) y venía ejecutándose un ULTA11 perteneciente al KLTA1 del proceso A.  
En un momento ocurren dos interrupciones, la primera (de máxima prioridad de atención) de un fallo de un dispositivo, la segunda una interrupción del DMA por fin de IO de otro KLTB (perteneciente a otro proceso B). Se sabe también que la prioridad de KLTB es mayor a la de KLTA1.  
Indique en detalle qué pasará en dicho momento teniendo en cuenta cambios de modo y de contexto.

### **Planificación**

-¿Cuál de los siguientes algoritmos de planificación podría generar starvation?

| 1. FIFO 2. RR 3. HRRN 4. VRR 5. Todos los mencionados 6. Ninguno de los mencionados |
| --- |
| Correcta: F |

-¿Cuál de los siguientes eventos podría tener en cuenta el algoritmo de planificación Feedback para replanificar?

| 1. Interrupción de Quantum 2. Un proceso nuevo llega al sistema 3. Interrupción de fin de IO 4. Un proceso se bloquea 5. B y C 6. A y D 7. Todos 8. Ninguno |
| --- |
| Correcta: G |

-Peter está pensando en implementar un algoritmo de planificación de corto plazo para su sistema, el cual cuenta con procesos muy variados que no fueron programados por él, por lo que desea que ninguno sufra starvation ni pueda monopolizar la CPU; además quiere priorizar aquellos que son interactivos (IO bound). ¿Cuál de estos algoritmos debería implementar?  
- FIFO | SJF (sin desalojo) | SJF (con desalojo) | HRRN | RR | VRR

-El supermercado SOTO en tiempos de COVID necesita dos algoritmos de planificación para organizar la atención en su local.   
Por un lado, para ingresar al supermercado se les preguntará a las personas que planean comprar: los que quieran comprar productos de primera necesidad podrán entrar según el orden de llegada, los que sólo quieran comprar snacks y alcohol deberán esperar más. Los jubilados y embarazadas podrán entrar apenas lleguen pero evitando sacar personas que ya entraron al local.  
Por otro lado, otra problemática a solucionar es el pesaje de frutas y verduras; se quiere evitar que algunas personas compren 20 productos distintos y hagan esperar mucho al resto en la cola.  
Ayude a SOTO con dos algoritmos de planificación para resolver estas problemáticas.

### **Sincronización**

-¿En qué caso podría ocurrir una condición de carrera?

| 1. Cuando hay varios procesos bloqueados esperando que ocurra un evento como resultado de otro del conjunto 2. Cuando tenemos distintos resultados sobre la ejecución de un sistema dependiendo de su orden de ejecución 3. Cuando dos o más procesos comparten recursos en modo lectura 4. Cuando varios procesos están en ejecución pero realmente no avanzan con ningún trabajo útil 5. Cuando dos o más procesos comparten recursos al menos uno en modo escritura en forma concurrente 6. C, D y E 7. B y E 8. Todos son correctos 9. Ninguno es correcto |
| --- |
| Correcta: G |

-Resolver problemas de mutua exclusión utilizando Test and Set:

* No produce espera activa.
* No es conveniente utilizarlo en sistemas con multiprocesamiento.
* Requiere la participación del SO para ejecutar la instrucción en modo kernel.
* Resuelve la mutua exclusión pero provoca alternancia en la ejecución de los procesos.
* Ninguna de las anteriores.

-¿A qué estado pasa un proceso cuando se bloquea por un semáforo? Cuando luego el mismo se desbloquea: ¿Pasa al estado Running o debe esperar? (justifique). Si uno quisiera proteger una mutua exclusión entre ULTs del mismo proceso, ¿traería algún problema utilizar semáforos mutex provistos por el sistema operativo?

-Compare las técnicas: deshabilitar interrupciones, uso de la instrucción test\_and\_set y uso de semáforos; usando los siguientes aspectos de comparación: espera activa, cambios de modo involucrados, posibles daños colaterales a todo el sistema.

-¿En qué casos no es necesario utilizar mutua exclusión con recursos compartidos entre varios procesos o hilos? Dé un ejemplo en pseudocódigo.

-Explique porque los problemas de condición de carrera son difíciles de detectar. Analice: ¿Podrían ocurrir en sistemas monoprogramados? ¿Podrían ocurrir en sistemas multiprogramados con un solo procesador?