EJERCICIO 1 **[4puntos].** Ordene los algoritmos de asignación de CPU siguientes con el criterio de menor a mayor penalización promedio a las ráfagas cortas:

a) algoritmo de asignación de barrido cíclico con quantum igual a dos milisegundos
b) FCFS
c) algoritmo descrito en el Ejercicio 4

Rodee con un círculo la opción elegida y justifique su respuesta:

(1) a<b< c (2) a<c<b (3) b<c<a (4) b<a<c (5) c<a<b (6) c<b<a

RESPUESTA: b<a<c Quien mas penaliza a las cortas es (b), luego (a) y luego (c)

EJERCICIO 4. Nos situamos en el ALGORITMO DE ASIGNACION DE CPU DE "COLAS MULTIPLES CON TRASPASO" que se concreta así:

a) hay tres colas Round Robin o Barrido Cíclico: cola 1, cola 2 y cola 3 (1 es la más prioritaria o de más peso, 3 es la menos prioritaria)
b) los valores de los quantum son 2, 4 y 8 milisegundos respectivamente para cola 1, 2 y 3.
c) al crearse los procesos entran en la cola 1.
d) un proceso pasa de la cola 1 a la cola 2 cuando ha consumido 1 quantum completo en la cola 1 (se entiende que ha consumido 1 quantum consecutivo de tiempo, sin bloquearse)
e) un proceso pasa de la cola 2 a la cola 3 cuando ha consumido 1 quantum completo en la cola 2(se entiende que ha consumido 1 quantum consecutivo de tiempo, sin bloquearse)
f) cuando los procesos pasan a la cola 3 allí permanecen hasta que terminan.
g) cuando un proceso se desbloquea vuelve a la cola en que estaba antes de bloquearse.
h) no hay desplazamiento (sin derecho preferente)

El comportamiento de los procesos es el siguiente: (unidades en milisegundos)

	Creación	Cpu	Bloqueo	Cpu	Bloqueo	Cpu	Bloqueo	Cpu
A	0	**15**						
B	1	**1**	1	**1**	1	**1**	1	**1**
C	2	**2**	1	**4**				
D	3	**2**	1	**2**				

Califique los siguientes enunciados en la hoja de respuestas:

1)	V*El proceso B permanece siempre en la cola 1 **(A=VERDADERO; B=FALSO)**
2)	F*El proceso C permanece siempre en la cola 1**(A=VERDADERO; B=FALSO)**
3)	B*Señale en la hoja de repuestas el proceso que es elegido para ejecutarse en t=2ms
4)	V*En t=2ms el proceso A ha satisfecho el criterio para pasar de la cola 1 a la cola 2**(A=VERDADERO; B=FALSO)**
5)	C*Señale en la hoja de repuestas el proceso que es elegido para ejecutarse en t=3ms
6)	D*Señale en la hoja de repuestas el proceso que es elegido para ejecutarse en t=5ms
7)	B*Señale en la hoja de repuestas el proceso que es elegido para ejecutarse en t=7ms
8)	A*Señale en la hoja de repuestas el proceso que es elegido para ejecutarse en t=8ms
9)	F*En t=12ms la cola 2 tiene como proceso mas antiguo el D y después el C **(A=VERDADERO; B=FALSO)**
10)	C*Señale en la hoja de repuestas el proceso que es elegido para ejecutarse en t=13ms
11)	B*Señale en la hoja de repuestas el proceso que es elegido para ejecutarse en t=17ms
12)	A*Señale en la hoja de repuestas el proceso que es elegido para ejecutarse en t=20ms

PREGUNTAS GENERALES SOBRE SISTEMAS OPERATIVOS.

Responda en la hoja de respuestas asignando A=VERDADERO; B=FALSO

13) V*Una parte del S.O. implementada como proceso puede sufrir que se le retire el control de la CPU y dependa del planificador a corto plazo para que se le vuelva a conceder.

14) V*En un sistema de tiempo compartido múltiples personas acceden simultáneamente al sistema para ejecutar sus trabajos.

15) F*Cuando un proceso realiza cualquier llamada al sistema pasa a estado bloqueado hasta que ésta termina.

16) F*En el algoritmo de Barrido Ciclico con quantum Q: si el proceso actual se bloquea antes de finalizar el quantum entoces es imposible que en este momento se asigne la CPU a otro proceso en este momento, debiendo quedar ociosa hasta que acabe el quantum para que pueda pasar a ejecutarse otro proceso.

17) V*En multiprogramación los distintos programas que estén en ejecución residen simultáneamente en memoria principal.

18) V*En un "Sistema Operativo en Red" el usuario ha de conocer la ubicación de los archivos

19) F*En un "Sistema Operativo Distribuido" el usuario ha de conocer dónde se ejecutan los procesos.

20) V*Generalmente, el software que compone el núcleo de un S.O. se mantiene permanentemente en memoria principal.

21) F*En el modelo "Cliente-Servidor" de S.O., cuando un proceso hace una solicitud a un proceso "Servidor de Archivos", se producen menos pérdida de tiempo en cambios de contexto que en la estructura de capas.

22) V*Algoritmo Round Robin: para valores de Q muy grandes en relación al tamaño de las ráfagas de CPU, este algoritmo se comporta como el FCFS (First Come First Served o FIFO)

23) F*Siempre que se ejecuta una llamada al sistema se produce un cambio de contexto.

24) V*Siempre que se ejecuta una llamada al sistema se produce cambio de modo usuario/privilegiado.

25) V*La estructura monolítica de S.O. es más rápida que la cliente-servidor.

26) V*Todas las partes de un S.O. con estructura monolítica se ejecutan en un único espacio de direcciones.

27) V*Siempre que el proceso actual no desea seguir ejecutándose ha de llamarse al planificador a corto plazo.

28) F*En la política de planificación FCFS, en promedio las ráfagas largas tienen un tiempo de espera mayor que las ráfagas cortas.

29) En la política de planificación de Barrido Cíclico con quantum Q, el tiempo de espera de una ráfaga es mayor para ráfagas largas que para ráfagas cortas.

ITEM ANULADO. En general, se ha dicho que el tiempo de espera es mayor para ráfagas largas, pues deben llegar a estar en primer lugar de la cola de ejecutables más veces, por lo que este item sería verdadero. Pero ciertamente hay que observar que en el caso de que Q muy grande llegando a ser mayor que el tiempo de CPU de las rafagas, entonces el algoritmo se comporta como el FCFS, en que el tiempo de espera es igual independientemente del tiempo de CPU de las ráfagas.

30) F*En la política de planificación FCFS (First Come First Served) o FIFO, la penalización es igual para ráfagas cortas que para ráfagas largas.

31) V*En un algoritmo de "Colas múltiples con realimentación" si tenemos la certeza de que las ráfagas de un proceso son de duración parecida, entonces es preferible que cuando un proceso se desbloquea prosiga en la misma cola en que estaba antes de bloquearse frente a que el proceso entre en la cola 1 (de mayor preferencia)

32) F*En el algoritmo SJF (Short Job First) sin desplazamiento queda resuelto el problema de inanición que podrían sufrir las ráfagas muy largas.

33) V*La instrucción máquina de leer de disco a nivel de pista/sector… debe ser siempre una instrucción privilegiada.

34) V*La estructura de S.O. llamada "Cliente-Servidor" tiene más tolerancia a fallos que la de "Capas".

35) F*La estructura de S.O. llamada "Cliente-Servidor" presenta en menor medida la característica de extensibilidad que la de "Capas".

36) F*En la estructura de S.O. "Cliente-Servidor" un proceso Servidor no puede hacer peticiones convirtiéndose en un cliente de otro proceso Servidor.

37) F*Si tenemos una política de planificación no apropiativa (sin desplazamiento), no se puede manejar una interrupción mientras se está ejecutando un proceso.

38) V*Modificar las posiciones de memoria donde se almacena la Tabla de Vectores de Interrupción solo se puede realizar en modo supervisor.

39) V*Nos situamos en una política de asignación de CPU de Barrido Cíclico con quantum Q en que tenemos un conjunto de procesos que no se bloquean y cuyas ráfagas de CPU son >Q, y sea S el tiempo necesario para realizar un cambio de contexto. La afirmación que debe calificarse como verdadera o falsa es: "en el caso en que Q=S, el 50% de tiempo de CPU se invierte en tiempo de SO para cambiar de contexto"

40) V*En un algoritmo de planificación de Barrido Cíclico no es necesario añadir una política de envejecimiento para evitar el problema de inanición.

41) F*En el algoritmo de Barrido Cíclico con quantum Q: se está ejecutando el proceso A y en el mismo momento en que ocurre fin de quantum se crea el proceso B, en este caso el proceso al que debe asignarse la CPU es al A.

PREGUNTAS SOBRE LA IMPLEMENTACION EN LINUX ESTUDIADA.

Responda en la hoja de respuestas asignando **A=VERDADERO; B=FALSO.**

42) F*Durante el periodo de tiempo en que se está creando un proceso nuevo, éste se califica en estado TASK_STOPPED hasta que concluyan todas las operaciones que conlleve la creación.

43) V*En la terminación de un proceso, para cada recurso que estuviera utilizando dicho proceso se decrementa el contador correspondiente que indica el número de procesos que lo están utilizando, y únicamente cuando este contador vale 0 se libera dicho recurso.

44) F*Es imposible que durante la terminación de un proceso el kernel encuentre que aún tiene procesos hijos qie aún no han terminado.

Sea el caso en que el proceso X creó únicamente al proceso Y todavía existente....

45) V*Cuando X realiza la operación ***`wait`*** es el momento en que la ***`task_struct`*** de proceso Y puede destruirse.

46) F*Cuando X realiza la operación ***`wait`*** es el momento en que el proceso Y pasa a estado EXIT_ZOMBIE.

47) V*Solo después de que X haya realizado la operación ***`wait`*** es cuando Y pasa a estado EXIT_DEAD

48) F*Dentro del planificador periódico, si se concluye que hay que replanificar se llama explicitamente al planificador principal ***`schedule`*** para que se elija un nuevo proceso a ejecutar.

49) V*El planificador principal ***`schedule`*** es invocado siempre que un proceso se bloquea

50) V*El planificador principal ***`schedule`*** es invocado siempre que un proceso termina

51) V*El planificador principal recorre las clases de planificación de mayor a menor prioridad (importancia), selecciona la clase de planificación de más alta prioridad (importancia) que no esté vacía y llama al procedimiento particular de dicha clase de planificación encargado de seleccionar el siguiente proceso a ejecutar.

52) F*En el planificador periódico nunca se activa el flag TIF_NEED_RESCHED
53) V*En la creación o desbloqueo de un proceso de más alta prioridad(importancia) que el actual se activa el flag TIF_NEED_RESCHED
54) V*El planificador periódico nunca cambia de clase de planificación
55) V*Mientras un proceso está ejecutando código de usuario es imposible que su contador preempt_count sea estrictamente positivo.
56) V*Es imposible que dos procesos tengan en un determinado momento de tiempo sus contadores preempt_count con valores estrictamente positivos.
57) F*Es posible que mientras un proceso está ejecutando código de usuario esté establecido el flag TIF_NEED_RESCHED