Ejercicios de planificación

1) Peter desarrolló un Sistema Operativo, donde las operaciones de semáforos duran 3 ms y cuya atomicidad está lograda mediante la deshabilitación de las interrupciones. Decidió ponerle un planificador Round Robin, con Q = 5 ms.
Leandro, quien es amigo de Peter, al analizar el sistema desde Pittsburgh (donde reside actualmente) le consulta si tiene sentido dicho diseño, dado que la mayor parte del tiempo se tendrán corriendo un programa formado por tres procesos, como se muestra a continuación:

Proceso A (llegada = 0)	**Proceso B (llegada = 1)**	**Proceso C (llegada = 5)**

wait(A);
wait(B);
calculos();
signal(A);
signal(B);
	wait(A);
	calculos();
	wait(C);
	signal(A);
	signal(C);
		wait(C);
		wait(A);
		calculos();
		signal(C);
		signal(A);

Peter decide entonces **realizar el diagrama de GANTT, e indicar como y cuando finalizaron los procesos**.

Nota: los semáforos se encuentran inicializados de la siguiente forma: A = 2, B = 0, C = 1, y la función cálculos() dura 3 ms

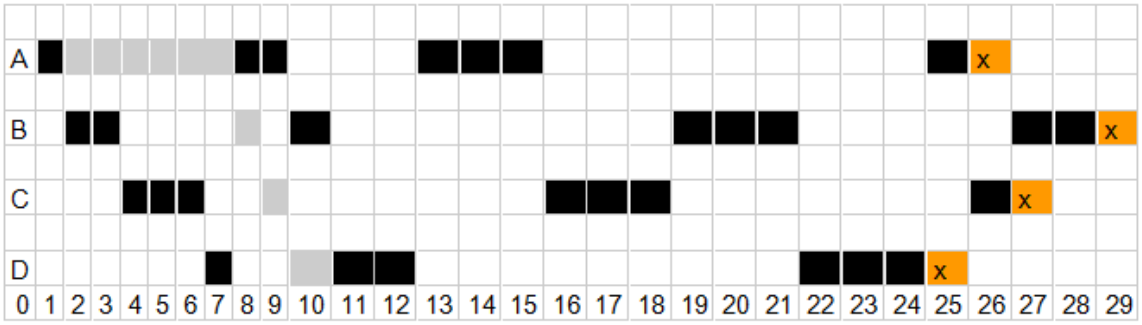
2) La oficina de la Junta Electoral está preparando el software necesario para dar los resultados de las elecciones PASO, ***anunciando los candidatos electos lo antes posible***, por lo que decide probar diferentes algoritmos de planificación para su sistema operativo. Los KLTs a utilizar están definidos de la siguiente forma:

1) Recuento: CPU(1) E/S (10) CPU(8) -> Realiza el recuento de votos y ***define los candidatos electos***
2) Estadísticas: CPU(2) E/S(3) CPU(5) -> Realiza estadísticas generales
3) Votantes: CPU(5) E/S(2) CPU (2) -> Revisa los padrones y envía e-multas a quienes no votaron

Los algoritmos para ser evaluados son SJF (con desalojo) y RR (con Q=2). Todos los KLTs están en la cola de listos, ordenados como se muestra más arriba, y deben finalizar para mostrar los resultados. ¿Cuál debe ser elegido? Justifique realizando diagramas de Gantt.

3) El siguiente diagrama representa la ejecución de 4 procesos planificados según un algoritmo. Observe detenidamente el gráfico e indique.

a) ¿Qué algoritmos se pueden estar utilizando si consideramos el Gantt hasta el instante 6 (antes de elegir a D)?
b) ¿Cuál fue el algoritmo utilizado considerando toda la ejecución de los procesos?
Justifique en ambos casos.

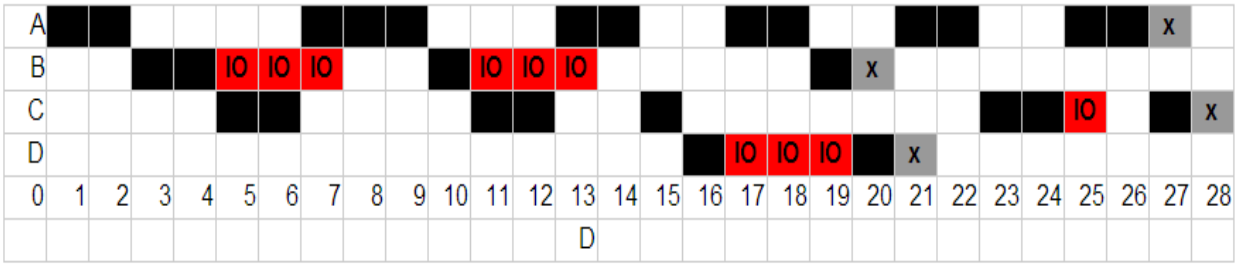


Nota: inicialmente los procesos llegan a una cola en el orden ABCD.

4) Un sistema utiliza un algoritmo Feedback de planificación de procesos. El mismo cuenta de 3 colas, preemptive entre ellas. Las reglas que sigue el algoritmo son las siguientes:

- Cola mayor prioridad RR Q = 2.
- Cola prioridad media RR Q = 3.
- Cola menor prioridad FIFO.
- Procesos nuevos van a la cola de mayor prioridad.
- Un proceso, al ser desalojado por fin de quantum, es penalizado y llevado a la cola inmediatamente inferior.
- Luego de una e/s los procesos son promovidos a la cola de mayor prioridad (RR Q = 2).

Considerando que inicialmente en la cola de listos de mayor prioridad se encuentran : A-B-C y que en el instante 13 llega el proceso D, encuentre los errores en la siguiente planificación:



5) En un Sistema Operativo que utiliza el algoritmo HRRN se están corriendo los siguientes ULTs, que pertenecen al KLT “Pregunta DOS” (que corre un juego de preguntas sobre comandos de consola):

ULT Buscador de participantes: CPU 4, E/S 4, CPU 4 -> Tiempo de llegada 0
ULT Buscador de preguntas: CPU 3, E/S 5, CPU 2 -> Tiempo de llegada 1
ULT Contador de monedas: CPU 1, E/S 5, CPU 1 -> Tiempo de llegada 5

Se sabe que todas las funciones de E/S utilizadas son propias de la biblioteca de hilos, y que ésta planifica usando el algoritmo SJF sin desalojo.

a) Realice el diagrama de Gantt correspondiente a esta ejecución.
b) El programador que realizó el juego decide utilizar la técnica de Jacketing para todas las E/S (convirtiéndolas en no-bloqueantes). ¿Mejorará esto la situación? Realice el nuevo diagrama de Gantt

6) Dada la siguiente traza de ejecución:

	CPU	**I/O**	**CPU**
Proceso A	10	2	5
Proceso B	8	3	7
Proceso C	9	4	7

a) Realice el diagrama de gant para los algoritmos SJF sin desalojo y SJF con desalojo.
b) Calcule para el proceso A el tiempo de ejecución y el tiempo de espera en ambos algoritmos.

Notas:

- Todos los procesos se encuentran en Ready
- Cada vez que se produce un cambio de proceso, el SO consume 2 unidades de tiempo
- Si el SO interviene pero eso no se traduce en un cambio de proceso, el SO consume 1 unidad de tiempo

7) Suponga que para un TP de la materia Técnicas de Gráficos por Computadora, usted grafica un Gaucho en 3D que anda lento (lo cual es raro porque su pc tiene dos procesadores). Para agilizar el proceso, al indagar descubre que este programa consta de un proceso, el cual tiene dos hilos de kernel (A y B) y que cada uno tiene dos hilos de usuario. Al ser un sistema operativo “UbUTNu”, su planificador es FIFO, y la biblioteca usa RR con Q=2.

Dada la siguiente traza de ejecución:

	Llegada	CPU	I/O	CPU
UTL-A1	0	3	1	1
UTL-A2	1	4	-	-
UTL-B1	4	2	-	-
UTL-B2	3	1	3	1

a) Realice un diagrama de GANTT para tener más información sobre la lentitud del Gaucho, sabiendo que toda petición de E/S pasa por la biblioteca de ULTs.
b) Calcule los tiempos de respuesta y ejecución, e indique cual debería merecer mayor importancia y porqué.

8) Un día cualquiera, Peter decide comprarse un iPhone 5S. Sin embargo, al experimentar la lentitud del sistema, decide sacarle iOS e instalarle Ubuntu Satanic. En una charla de ascensor, el vecino de su edificio le consulta si considera que realizó una buena elección. Para saberlo, decide ejecutar un par de procesos, registrar su ejecución y luego imprimir el GANTT para enviarle al vecino.
Sabiendo que el planificador es RR con Q=3, que existe una sola CPU y un solo dispositivo de E/S, grafique dicho diagrama de GANTT, sabiendo que la traza de los procesos PA y PB es la siguiente:

	Llegada	**CPU**	**I/O**	**CPU**
PA - KLT1	0	3	12	1
PA - KLT2	13	2	-	-
PB - KLT3	2	4	-	-
PB - KLT4	4	1	3	1

Nota: Cada vez que interviene el SO para un cambio de hilo, se consume 1 ms si se mantiene en el mismo proceso, y 2 ms en caso contrario.

9) Suponga la siguiente traza de ejecución de los procesos A y B, con un sistema operativo que planifica bajo RR con Q=4:

	Llegada	**CPU**	**I/O**	**CPU**
PA - UTL1	0	3	-	-
PA - UTL2	1	10	-	-
PB - UTL3	2	2	3	1
PB - UTL4	3	1	-	-

Se pide:

a) Realice el diagrama de GANTT de la ejecución de los mismos
b) Indicar en cuales instantes de tiempo se produjo un cambio de modo de ejecución
c) Indicar en cuales instantes de tiempo el procesador recibe una interrupción

Nota: La biblioteca de usuario planifica bajo el algoritmo FIFO. Todo pedido de E/S pasa por la biblioteca.

Ejercicios de sincronización

1) Peter necesita sincronizar los procesos que ocurren cuando se toman finales. Por cada mesa de examen hay un jefe de mesa que se encarga de recibir las libretas, pasar las notas a ellas y luego entregarlas.
La mesa cuenta con tres profesores que se ocupan de corregir los finales. La mesa de examen tiene que llegar a los treinta alumnos. Dado el siguiente pseudo-código:

Proceso jefe de mesa (1 instancias)	**Proceso profesor (3 instancias)**	**Proceso alumno (n instancias)**

entregar_examenes();
entregar_notas();
	corregir_final();	entregar_libreta();
		hacer_final();
		recibir_libreta();

Coloque adecuadamente en los tres procesos semáforos para que el jefe de mesa no entregue el examen a los alumnos hasta que los treinta no hayan entregado su libreta y además, para que las libretas se entreguen todas juntas, una vez que se corrigieron todos los finales.

Nota: indicar para cada semáforo utilizado, su tipo y su valor inicial.

2) Suponga que lo contrata Peter Rial, para sincronizar su sistema de gestión de chismes. Al parecer Peter Ventura lo programó, pero como no sabe nada de sincronización, se le pide a usted que lo ayude (a cambio de que luego salga en intrusos como el gran “sincronizador”). Dado el siguiente pseudo-código:

Proceso Mediática (n instancias)	**Proceso Rial (1 instancia)**

registrar_escándalo();
enviar_a_rial();
difundir_video();
asistir_a_instrusos();
	procesar_escándalo_nuevo();
	invitar_al_programa();

Sincronice la ejecución de ambos programas mediante el uso de semáforos exclusivamente, para garantizar que el proceso Rial solamente procesa escándalos cuando los mismos hayan sido enviados, y que el personaje mediático solo difunde su video cuando Rial ya lo invitó a su programa. Tenga en cuenta que si bien Rial invita a todos los mediáticos, solo dos de ellos puede asistir a la vez en un programa.

Nota: indicar para cada semáforo utilizado, su tipo y su valor inicial.

3) La AFIP atienden a contribuyentes todos los días hábiles, pero además realiza operativos anti-lavado constantemente. Dicha actividad ha sido modelada utilizando cuatro procesos, dos que simbolizan el actuar de la oficina, y dos que simbolizan a los contribuyentes y a los evasores (entre paréntesis figuran

las instancias de cada uno).

AFIP-Atencion (1)	**AFIP-Operativos (1)**	**Contribuyente (N)**	**Evasor (N)**
while (true) { *AtenderTramite();* *}*	*while (true) {* *RealizarOperativo();* *}*	*while (true) {* *HacerTramite();* *}*	*while (true) {* *Lavar("$1000");* *}*

La oficina de atención está ociosa hasta que alguien se acerca a realizar un trámite. Por otro lado, los contribuyentes deben esperar a que la oficina de atención se desocupe, para ser atendidos. Si la AFIP detecta una operación de lavado, realiza un operativo. Por otro lado los evasores prefieren no lavar más de $100.000 de manera continua, para evitar sospechas. Por último, dada la baja cantidad de personal que trabaja aquí, las oficinas de atención no reciben trámites durante los operativos, dejando que estos se acumulen infinitamente.

Sincronice estos procesos utilizando sólo semáforos y sin que sufran deadlock.

4) Suponga que Peter programó en Objective C para iPhone un código para salir a competir con la aplicación del momento: Pregunta2. En esta parte del mismo, existen muchos hilos que validan preguntas y las que son apropiadas las dejan listas para que un hilo servidor, del cual existen muchas instancias, la envíe a algun usuario ávido de contestar alguna pregunta.
El pseudo-código, que cada tanto falla por razones desconocidas, es el siguiente:

S1.valor = 1; S2.valor = 100; S3.valor = 0

Validador de Preguntas Nuevas (n instancias)	Servidor de preguntas (n instancias)

while (true){
wait(S1)
wait(S2)
pregunta = obtener_pregunta_validada()
depositar(cola, pregunta)
signal(S1)
signal(S3)
}
	while (true){
	wait(S1)
	wait(S3)
	pregunta2 = retirar(cola)
	enviar(pregunta2,)
	signal(S1)
	signal(S2)
	}

a) Indique paso a paso la secuencia de ejecución de ambos hilos que generaría un problema, y explique qué problema es.

b) Implemente en pseudo-código la función signal(), sabiendo que la estructura "semáforo" tiene el siguiente formato:

```
struct semáforo {
        int valor;
        LISTA* bloqueados;
        … // Otros atributos
};
```

Nota: para el punto b) se disponen de las funciones syscall_bloquear(proceso) y syscall_despertar(proceso), y se debe garantizar la atomicidad de la función a través de alguna solución provista por el hardware (se pueden agregar campos a la estructura si lo considera necesario)

5) Peter, quien es una persona que suele tener mala suerte, se esguinzó corriendo el colectivo. Por lo tanto, necesita ir a la guardia del Hospital OSD, donde deberá seguir el siguiente procedimiento:
En primer lugar, indicará la especialidad deseada en una máquina, dónde obtendrá un ticket con un número correspondiente al turno. Dicho valor es de 0-99, luego se reinicia.
El ticket generado por la máquina ingresa en forma virtual en una cola de tickets consultada por los recepcionistas. Para evitar que haya mucha espera en la sala, el sistema permite que haya sólo 60 tickets pendientes.
Cuando un recepcionista obtiene un ticket atenderá a la persona según el número que se indique en el mismo. Según lo que le explique el paciente, lo derivará con un especialista específico.

Peter (u otro paciente) (N)	Recepcionista (5)	Notas

turno= solicitarTicket()
sentarse();
explicarSituación()
atenderseConEspecialista()
	While(1) {
	turno= obtenerTicket()
	escucharPaciente()
	asignarEspecialista()
	}
		Nota 1: El paciente se levanta de la silla recién cuando el recepcionista lo llama.
		Nota 2: Hay 20 sillas

6) Peter está a punto de rendir el último final de su carrera. Sus amigos organizan los festejos en la puerta de la facultad, y para ello compran un maple con 96 huevos. Dadas las dimensiones del maple, los amigos tomarán los huevos de a uno a la vez. Peter recibirá el homenaje, para luego lavarse y secarse con la toalla que le ofrecerá su madre. Por último, se tomará una foto con todos sus amigos.

Peter	Amigo (4 instancias)	Madre de Peter

salir_de_la_facultad();
recibir_huevos();
lavarse();
secarse();
sacarse_foto();
	esperar();
	while(huevos > 0) {
	tomar_huevo(maple);
	huevos--;
	arrojar_huevo();
	}
	sacarse_foto();
		esperar();
		emocionarse();
		alcanzar_toalla();

Sincronice los siguientes procesos utilizando únicamente semáforos, sin caer en deadlock o inanición.

7) Peter decidió festejar su cumpleaños e invitar a muchos amigos, por lo que debe elaborar muchas pizzas. Como peter quiere terminar lo antes posible, consigue ayuda para poder paralelizar el trabajo mientras llegan los invitados. Sincronice (utilizando sólo semáforos) el trabajo de Peter y su equipo (amigos, abuelos, primos) para que la fiesta sea un éxito.

<table>
<tr><th>Amigo (10 Instancias)</th><th>Abuelo/a (4 Instancias)</th><th>Primo/a (7 Instancias)</th><th>Peter</th></tr>
<tr>
<td rowspan="3">WHILE(1){
pizzera= getPizzeraLibre();
estirarMasa(pizzera);
dejarLeudar();
ponerSalsaTomate();
cocinarEnHorno();
}</td>
<td rowspan="3">WHILE(1){
bowl= getBowlLibre();
mezclarIngredientes(bowl);
armarBollo();//Se amasa en la mesada
colocarEnBandeja();
dejarLeudar();
}</td>
<td>WHILE(1){
sacarPizzaHorno();
ponerMuzzarella();
cocinarEnHorno();
}</td>
<td rowspan="3">WHILE(1){
sacarPizzaHorno();
ponerPizzaEnTabla();
cortarPizza();
rica= testearPizza();
IF(rica){
ponerPorcionesMesa();
}
else{
tirarPizza();
}
}</td>
</tr>
<tr><th>Invitado (N instancias)</th></tr>
<tr>
<td>WHILE(1){
agarrarPorcion();//De la mesa
comerPizza();
}</td>
</tr>
</table>

Para esto tenga en cuenta que: hay 3 bowls, 5 pizzeras y un horno. Los abuelos van colocando los bollos en una única bandeja que tiene una capacidad máxima de 10 bollos (ya que si no se comienzan a pegar). Peter podrá colocar pizzas en el mismo momento en el que los invitados agarren porciones (aunque se pelee con Bernstein). Tenga en cuenta que Peter corta las pizzas en 4, pero que de cada una se come una porción. *Nota: en el proceso de elaboración de una pizza se la coloca dos veces en el horno, la primera, para preparar la prepizza, la segunda para que se derrita el queso.*

8) Peter está por hacer su primer salto en paracaídas. Como no quiere dejar ningún detalle librado al azar, decide simular su caída, incluyendo al fotógrafo que lo va a retratar y a su esposa. El salto se hará a 5000 pies, saltando primero él y después el fotógrafo. Antes de pasar los 3000 pies, se le sacará una foto desde arriba, y luego de pasar los 3000 pies una desde abajo (para esto, el fotógrafo primero *desafiará las leyes de la gravedad*, descendiendo más rápido que él). Luego abrirán el paracaídas y aterrizarán (no importa en qué orden).
En tierra lo esperará su esposa, quien se preocupará cuando Peter pase los 3000 pies, y lo abrazará luego de aterrizar.

Sincronice los siguientes procesos **utilizando únicamente semáforos**, sin provocar deadlock o starvation.

Peter	**Fotógrafo**	**Esposa**

while(1) {
  tirarseA5000Pies();
  pasarLos3000Pies();
  abrirParacaidas();
  aterrizar();
}
	while(1) {
	  tirarseA5000Pies();
	  sacarFotoDesdeArriba();
	  desafiarLeyesDeGravedad();
	  sacarFotoDesdeAbajo();
	  abrirParacaidas();
	  aterrizar();
	}
		while(1) {
		  preocuparse();
		  abrazarParacaidista();
		}

9) Los creadores de koopa decidieron que su implementación fuese multi-thread. Peter, quién programó la libmemoria.so entre otras cosas, va a realizar la modificación. La especificación de koopa dice: "*Ahora koopa realiza hasta tres operaciones de asignación de particiones en paralelo y a la hora de liberar particiones, hasta cinco operaciones*".

a) Ayudemos a Peter a modificar su implementación de memoria utilizando semáforos donde fuese necesario, indicando sus valores iniciales.
b) Implemente las primitivas de semáforos con espera activa, en caso que presenten sección crítica, sincronice (con un color diferente) dichas primitivas utilizando alguna técnica hardware de su elección.
Aclaración: Solo utilizar semáforos. No puede agregar o quitar código.

Las funciones particion_libre(...) y particion_ocupada(...) en caso de hallar una partición, la marcan para que no esté disponible. Mientras que asignar_particion(...) y liberar_particion(...) actualiza el listado de particiones.

t_list *particiones //Variable global con particiones de la memoria.

```
int almacenar_particion(...) {

   if(datos_invalidos( id, tamanio, contenido)) {
      return -1 ;
   }
   particion_libre = particion_libre(particiones, tamanio);

   if(particion_libre) {
      asignar_partición(particiones, particion_libre, id,
                              tamanio, contenido);
      return 1;
   } else {
      return 0;
   }

}
```

```
int eliminar_particion() {

   particion = particion_ocupada(particiones, id);

   if(particion) {
      liberar_particion(particiones, particion);
      return 1;
   }

   return 0;
}
```

10) Implemente las funciones wait() y signal(), utilizando la instrucción test_and_set para garantizar la concurrencia.

Suponga que tiene los procesos A, B y C. Sincronícelos con semáforos para que ejecuten siguiendo la secuencia BACABACA… (Utilice la menor cantidad de semáforos posible)

11) Sincronice el siguiente código, correspondiente a un proceso que genera procesos hijos, para evitar inconsistencias, deadlocks e inanición. Además debe tener en cuenta lo siguiente:

- El archivo donde se escriben los logs es único.
- No debe haber más de 50 procesos en ejecución

- El padre debe escribir en el log antes que el hijo recién creado.

```
int main () {
        while (true) {
                pid = fork();
                if (pid < 0) {
                        log('Error');
                } else if (pid == 0) {
                        log("Y yo soy el hijo");
                        realizarTarea();
                        exit(0); // Finaliza el proceso hijo
                }  else { // Padre
                        log(pid + " soy tu padre");
                }
        }
}
```

Ejercicios de deadlock

1) Dadas las matrices, defina: ¿qué enfoque utilizará para tratar el Deadlock?

Recursos Asignados

	R1	R2	R3	R4
P1	1	1	2	0
P2	2	1	0	1
P3	1	0	0	1
P4	1	0	1	1
P5	0	0	0	0

Peticiones Actuales

	R1	R2	R3	R4
P1	3	2	2	2
P2	3	1	0	0
P3	3	3	3	2
P4	1	1	1	0
P5	1	1	1	1

Recursos Totales

R1	R2	R3	R4
5	3	4	3

¿Qué procesos están en deadlock y cómo lo solucionaría? Tenga en cuenta que el proceso 3 debe finalizar correctamente. Indique en un grafo de asignación de recursos dónde podría existir deadlock si sólo grafica los procesos 2 y 4.

2) En un sistema cuyo estado se encuentra representado por las siguientes matrices se tienen como recursos máximos R1= 7, R2=4, R3=2,R4=3. Resuelva:

	R1	R2	R3	R4
P1	1	2	0	0
P2	3	0	1	2
P3	1	0	0	0
P4	0	2	0	1

Recursos Asignados

	R1	R2	R3	R4
P1	2	2	1	0
P2	4	1	1	2
P3	1	2	0	2
P4	1	3	2	1

Peticiones Máximas

a) ¿Qué algoritmo de tratamiento de deadlocks se puede utilizar con los datos presentes? Explique en qué se basa dicho mecanismo.

b) Indique si se satisfacen inmediatamente cada uno de los siguientes pedidos, utilizando el mecanismo indicado en b):

- P1 → 2 R1 1 R3
- P2 → 1 R1 2 R3
- P3 → 1 R1
- P4 → 1R1 1 R3

3) Dadas las siguientes matrices de asignación y necesidad de recursos, determine si el sistema se encuentra actualmente en estado de interbloqueo (deadlock). En caso de que hubiese interbloqueo, determine qué proceso/s se podría/n escoger como víctimas/s para solucionarlo. Si no hubiese deadlock, muestre una secuencia de ejecución en la cual todos los procesos terminen correctamente. Justifique.

Asignación

	R1	R2	R3	R4
P1	1	0	0	0
P2	0	1	0	1
P3	1	0	1	0
P4	0	0	1	0
P5	0	0	0	0
RT	2	1	3	1

Necesidad

R1	R2	R3	R4
1	0	1	0
0	0	1	0
1	0	0	0
1	1	0	0
2	0	3	0

4) Dadas las siguientes matrices de un sistema operativo que utiliza como política evadir el Deadlock:

	R1	R2	R3	R4
P1	1	2	0	1
P2	1	3	2	1
P3	2	2	2	2

Recursos Máximos

	R1	R2	R3	R4
P1	0	2	0	0
P2	1	1	0	0
P3	2	2	2	0

Recursos Asignados

a) Halle la matriz de recursos disponibles requeridos para que pueda otorgarse 2 instancias del Recurso 3 al Proceso 2. La cantidad de recursos debe ser la menor posible.

b) Indique la secuencia de finalización de los procesos.

5) Un grupo de vedettes intenta promocionar su nueva obra de teatro yendo a diferentes programas. Podemos diferenciarlas entre principal (hay una sola), vedettes secundarias (hay cinco) y mellizas griegas (hay dos). Aprovechando sus conocimientos de Sistemas Operativos, deciden simular esta situación utilizando procesos, para poder asistir a los programas sin descuidar la obra de teatro. El pseudo-código es el siguiente:

```
Programa de Chimentos() {

  Entrevistar(melliza, 1)
                ⇒
  if(bajo_rating) {
    Entrevistar(secundaria, 1)
  }
}
```

```
Programa de Cable() {

  Entrevistar(secundaria, 3)
  Entrevistar(melliza, 1)
                ⇒
  if(aparece_sponsor) {
    Entrevistar(principal, 1)
  }
}
```

```
Obra de Teatro() {

  Bailar(secundaria, 2)
  Monologo(principal, 1)
                ⇒
  if(el_publico_no_se_rie) {
    Bailar(melliza, 2)
  }
}
```

Habitualmente el programa de chimentos tiene un rating muy alto. Por otro lado, el programa de cable no suele tener ningún sponsor. Y un 90% de las veces, el público que va al teatro se ríe con el monólogo de la vedette principal.

a) En este instante los procesos están donde indica la flecha ¿Podrán promocionarse sin descuidar la obra de teatro? Justifique

b) Si la respuesta es afirmativa, indique en qué orden deberían finalizar los procesos para que esto ocurra. Si es negativa, indique qué sucedería si se contrata una vedette secundaria más.

6) Sobre un sistema operativo que emplea el algoritmo de detección de deadlock se ejecutará los siguientes programas: Valores de las variables globales: sem_t A=5, B=7, C=3;

Proceso X | PID 2345	Proceso Z | PID 3001	Proceso Y | PID 2352	Proceso W | PID 2398

wait(A, 2);
injectarSQL();
wait(B, 1);
wait(B, 6); ← T1
informarVulnerabilidad();
signal(B, 6);
wait(B, 1); ← T2
siguienteVictima();
...
	wait(C,2);
	wait(A,1);
	sleep(200); ← T1
	signal(A,1);
	kill(2398, SIGKILL);
	wait(A,1); ← T2
	...
		wait(C,1);
		wait(B,1);
		enviarMail();
		wait(A,1);
		wait(B, 1); ← T1
		wait(A, 1);
		spamearAlVecino();
		signal(B,1);
		wait(C,1); ← T2
		...
			recibirCorreo();
			wait(A,1);
			wait(B,5);
			wait(C,2); ← T1
			signal(C,1);
			signal(B,1);
			listarCorreo();
			wait(B,1); ← T2
			exit();
			...

Notas:
* wait(sem,n) Toma en forma atómica 'n' instancias del semáforo 'sem'. Se bloquea si no están disponibles.
* signal(sem, n) Libera 'n' instancias del semáforo 'sem'.
* ← Indica el instante donde el proceso se encuentra ejecutando. En caso de tratarse de un wait() significa que la función solicito un recurso al SO y este aun no lo entregó.

Se ejecutan los programas y después de un tiempo, el administrador de sistema activa el módulo de detección de deadlock. Indique el resultado que vería para los instantes T1 y T2, así como el estado de cada proceso. En T2 se sabe que W finalizó.

7) En un sistema que utiliza el algoritmo del banquero como método para evitar los deadlocks, indique cuáles de los siguientes pedidos serían satisfechos inmediatamente, suponiendo que cada uno se efectuará sobre el estado presentado inicialmente:

1) P1: una instancia de R1
2) P1: una instancia de R4
3) P3: 2 R2
4) P3: 2R1 y 3 R4

	R1	R2	R3	R4			R1	R2	R3	R4
P1	3	2	0	2		P1	3	2	0	0
P2	3	5	2	1		P2	2	3	1	1
P3	8	6	1	5		P3	1	2	1	2
P4	3	4	2	3		P4	2	2	2	1

PETICIONES MÁXIMAS RECURSOS ASIGNADOS

R1	R2	R3	R4
10	9	6	8

RECURSOS MÁXIMOS

8) Un sistema que evade el deadlock se encuentra en el siguiente estado:

Peticiones Máximas

	R1	R2	R3	R4
P1	2	1	2	4
P2	2	4	5	1
P3	0	1	2	4
P4	4	4	6	2

Recursos Asignados

	R1	R2	R3	R4
P1	2	1	2	1
P2	1	2	2	1
P3	0	0	0	3
P4	1	0	3	2

Halle el vector de recursos totales que tenga la menor cantidad posible de instancias de recursos y que además pueda aceptar la solicitud de una instancia de R1 por parte de P4.

Ejercicios de Memoria

1) Un sistema utiliza un esquema de paginación bajo demanda, con una política de sustitución global que utiliza el algoritmo LRU. En el mismo se dedican únicamente 4 frames para alojar páginas de procesos de usuario (frames: 1-2-3-4).
Dicho sistema posee en un momento dado únicamente dos procesos, A y B, que se encuentran ejecutando. El proceso A necesita ejecutar una instrucción la cual referencia a sus páginas 0, 1 y 2. A su vez, el proceso B necesita ejecutar otra instrucción para la cual necesita tener cargadas en memoria física a las páginas 0, 1, 2, 3 y 4.
Luego de cargar ciertas páginas el estado de los procesos es el siguiente:

TP Proceso A

Pág	Frame	P	m	T última referencia
0	1	1	1	9
1	4	1	1	12
2	2	0	0	6
3	-	0	0	-
4	-	0	0	-

TP Proceso B

Pág	Frame	P	m	T última referencia
0	3	1	0	10
1	1	0	0	4
2	1	0	0	5
3	4	0	0	8
4	2	1	0	11

Por lo tanto, con el fin de poder ejecutar las instrucciones, los procesos referencian a las siguientes páginas en el orden indicado: **1B – 3B – 2A – 2B**

a) Indique:
- La cantidad de PFs (page faults) que se generan.
- Las páginas que se leen de disco y las que se escriben de memoria a disco.
- El estado de las tablas de páginas luego de la última referencia.

b) ¿Podrán los procesos ejecutar cada uno su instrucción deseada? ¿Podría responder la pregunta anterior antes de resolver el punto 1? ¿Podría ejecutar algún proceso su instrucción si se modifica el orden de las referencias? Justifique en todos los casos**.**

2) Un sistema utiliza un esquema de paginación bajo demanda con un tamaño de frame de 2 KiB, direcciones lógicas de 16 bits y una política de asignación fija de 3 frames. En un momento se sabe que la tabla de páginas de un proceso es la siguiente (siendo la página 2 la página que hace más tiempo está cargada en memoria):

Pág	Frame	U	M	P
0	4	1	0	1
1	4	1	0	0
2	3	1	1	1
3	7	1	0	1
4	7	1	0	0
5	7	1	1	0

Posteriormente se referencian las direcciones lógicas: 0843h(R) - 2122h(R) - 027Ah(W) y aplicando un algoritmo de sustitución de páginas se llega al siguiente estado:

Fr	P
3	1
4	4
7	0

¿Cuál/es algoritmos se puede/n haber utilizado? Justifique

3) Un Sistema Operativo tiene 64KB de memoria y utiliza paginación por demanda, con páginas de 1 KB y direcciones de 20 bits. El proceso P, de 10KB, está cargado en memoria y quiere realizar la operacion int suma = a + b.

a) Indique cuál es el tamaño máximo que un proceso podría direccionar en este esquema.
b) Indique el estado final de la memoria, si P tiene asignados 3 frames (inicialmente vacíos) y el código ocupa la página 0 del proceso, “suma” está en la página 3, “a” en la página 6 y “b” en la página 6.
c) ¿Qué sucedería si “b” estuviese en la página 8?
d) ¿Qué sucedería si “a” y “b” estuviesen en la página 12?

4) Un sistema que trabaja con paginación bajo demanda utiliza 32 bits para el direccionamiento y divide toda su memoria en frames de 1KiB. En él se ejecuta un proceso que realiza las siguientes referencias a memoria: 3500, 2100, 1200, 500, 3501, 2101, 4500, 3508, 2101, 1050, 50, 4200. El proceso tiene asignados 3 frames (inicialmente vacíos).
a) Calcule la cantidad de fallos de páginas producidos por los algoritmos Clock y Fifo
b) Si se otorgara un frame más al proceso, ¿ocurriría la anomalía de Bélády?

5) Un esquema de memoria virtual tiene un tamaño de página de 1024 bytes y la memoria física tiene 4 marcos de página. La Tabla de páginas de un proceso es:

Página Virtual	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
Marco	**3**	**1**	**---**	**---**	**2**	**---**	**0**	**---**	**---**	**---**

a) Indique detalladamente ¿Cuáles son las direcciones físicas de las siguientes direcciones virtuales (expresadas en decimal): 1024, 0, 3728, 1025, 1240?
b) Suponiendo que en vez de tablas de páginas, se utilizará una tabla de páginas invertida, indique qué valores tendría la misma.

6) Una computadora que está ejecutando solamente dos procesos, utiliza un SO experimental que tiene una memoria de 256 KiB y utiliza tabla de páginas invertida con tamaño de página de 1024 bytes y direcciones de 20 bits. Se conocen las siguientes direcciones lógicas:
Proceso 1: (36;125) (35;256) (39;716)
Proceso 2: (40;26) (37;456) (32;951)

a) Grafique la tabla de páginas y complete las entradas que corresponda.
b) Indique cuántas entradas tiene la tabla de páginas invertida y cuántos frames tiene la memoria.
c) ¿Cuáles son las direcciones físicas de las direcciones (39;716) y (32;951)?

Notas:
- *Asuma que el resto de las páginas de los procesos no están en Memoria Principal.*
- *Utilice como función de hash: $f(x) = (x-36)^2$ (donde x es el número de página)*

7) Un Sistema Operativo administra sus procesos utilizando Segmentación simple, con direcciones de 18 bits y 64 KiB de memoria. El Sistema Operativo utiliza una lista de particiones para administrar la memoria disponible, con política FIFO, y está alojado en la segunda mitad de la memoria. La tabla de segmentos del único proceso corriendo en un instante determinado es la siguiente:

Segmento	Base	Límite / Tamaño
0	1000h	0111h
1	1200h	00FFh
2	1300h	0BEEh

a) Calcule qué dirección física produciría la dirección lógica 200EEh
b) Calcule qué dirección física produciría la dirección lógica 00211h
c) Si el proceso necesitara otro segmento de tamaño 15FFh ¿En qué lugar de la memoria lo ubicaría?
d) Si agregado al punto c), el proceso necesitara otro segmento de tamaño 4B00. ¿Habría espacio libre suficiente en la memoria para ubicarlo?

8) Lolo posee un sistema que utiliza paginación bajo demanda, con una asignación fija de 3 frames por proceso y una política de sustitución de frames local.
Al ejecutar un programa, en un momento determinado, el puntero a próxima instrucción es 123.
…
123 OPLOCA , OP1 , OP2 , OP3 , RES
…
Dicha instrucción, con los operandos OP1, OP2 y OP3, realiza una operación y guarda el resultado en RES.
Recordando que el ciclo de instrucción básico consta de: Búsqueda de instrucción ->Decodificación Instrucción -> Búsqueda de operandos -> ejecución de instrucción -> Guardar resultados. Y sabiendo que:

Ubicación de datos y código en la imagen Proceso

OPLOCA	PÁG 2
OP1	PÁG 6
OP2	PÁG 6
OP3	PÁG 10
RES	PÁG 5

Estado Frames asignados

PTR	Frame	Pág	U	M
	5	1	1	1
->	8	0	0	1
	11	10	0	0

a) Identifique la secuencia de referencias a páginas necesarias para poder completar el ciclo de dicha instrucción. Utilizando el algoritmo de sustitución de páginas Clock modificado mostrar el estado de la tabla de páginas luego de cada referencia.

b) En cada caso indicar (luego de cada referencia): si hay page fault, cuáles son las páginas leídas y escritas de/a disco (si corresponde).

c) ¿Se puede finalmente completar el ciclo de instrucción correctamente? En caso afirmativo justifique y en caso negativo relacione el problema a un concepto de memoria (explicando el mismo).

9) Se tiene una PC con procesador de 8 bits. La misma tiene instalado un SO que emplea paginación bajo demanda, que asigna 3 frames fijos por proceso y utiliza sustitución local basada en clock modificado.

Página	Dirección física
5	AEh
7	CEh
3	ACh
2	CCh
5	2Ch

Actualmente se está ejecutando un proceso del cual se sabe que la única página que tiene en memoria es la página 0 (cero) y reside en el frame 1. Dicha página tiene sus bits de modificado y uso ambos en 1.

Se sabe además que el proceso realizará una serie de lecturas en memoria, de las cuales sólo se conoce el número de página y la dirección física con la que se accederá a la misma:

Se pide:

1. Indicar qué frames le asignará el sistema operativo al proceso. Justifique
2. Indicar las direcciones lógicas (en hexadecimal) para cada una de las referencias a memoria

Nota: Para el punto 1 se deberá justificar realizando la traza de páginas con dicho algoritmo.

10) Un esquema de memoria virtual que utiliza paginación bajo demanda con política de asignación de frames fija y con alcance local, tiene un tamaño de página de 1024 bytes y la memoria física tiene 4 marcos de página por proceso. Un proceso de 12 KiB inicia su ejecución realizando las siguientes referencias a memoria:

En decimal: 1024 - 0 - 7000 - 6500 - 45 - 2999 - 2035 - 4018 - 4019 - 12800 - 11000 - 9500

Indique detalladamente el estado de los marcos luego de cada referencia utilizando:
a) Política de reemplazo LRU.
b) Política de reemplazo FIFO.

11) Se tiene un proceso de 10 Kb ejecutando en el sistema, realizando las siguientes referencias a memoria (en forma de página,desplazamiento): (1,20), (3,0), (5,200), (4,30), (3,20), (4,11), (7,20) (8,88) (3,10)

a) Indique el estado de la memoria en cada instante de tiempo para los algoritmos de reemplazo LRU y CLOCK
b) Indique qué algoritmo tuvo una mejor performance, justificando el criterio elegido
c) Suponga que la referencia a la página 5 era de escritura, ¿cambiaría en algo la solución del punto a)?
d) ¿Cuantas referencias a memoria hubieran existido para el punto a, en el algoritmo LRU, si se contaba con una TLB de 10 entradas?

Nota: el esquema de memoria tiene una asignación fija de 3 frames por proceso, con alcance local

Ejercicios de entrada/salida

1) Un disco está formado por 100 cilindros, 4 platos y 20 sectores por pista. El tiempo entre pistas es de 1ms. En el instante 0ms la cabeza se encuentra en la pista 50 y llegan los pedidos de pistas 60, 90, 25, 45 y 15. En el instante 20ms llega el pedido de pista 85 y en el instante 25 ms llega el pedido de pista 70.

Calcule el tiempo de atención e indique en qué orden serán atendidos los pedidos utilizando:

a) Algoritmo F-SCAN y la cabeza se encuentra subiendo.
b) Algoritmo C-SCAN y la cabeza se encuentra subiendo..

2) Peter desea planificar los pedidos que llegan a sus dos discos rígidos (D1 y D2) utilizando el mismo algoritmo (sin inanición y sin usar N-STEP, ya que no le agrada). Ambos discos tienen 50 pistas, pero el disco D1 tiene mayor capacidad y gira un poco más rápido. El tiempo entre pistas es de 1ms, y las cabezas de ambos discos están en la pista 0. Los siguientes pedidos (n° de pista) llegan en los instantes señalados:

D1: Tiempo 0ms: 40, 2. Tiempo 10ms: 49, 40
D2: Tiempo 0ms: 35, 5, 35, 5. Tiempo 20 ms: 10

a) Indique qué algoritmo se debería elegir para que el tiempo usado para recorrer las pistas sea el menor posible.
b) Si se deseara agregar un disco D3 de características similares a D1 para formar un RAID ¿Qué esquema sugeriría utilizar? Justifique

3) Peter tiene un disco rígido algo viejo, con capacidad total de 4 GB, sectores de 4KB (usando Advanced Format), 8 caras, 1024 sectores por pista y un tiempo entre pistas de 1ms. Su curiosidad lo lleva a investigar cómo se comportarían algunos algoritmos de planificación ante una cierta cantidad de pedidos.

Dentro de su investigación, cuando el brazo está en la pista 0, al disco llegan los siguientes pedidos (n° de pista):
Tiempo 0: 100, 15, 90
Tiempo 20: 110, 30, 35, 40

Calcule los tiempos que tomaría atenderlos utilizando FSCAN y NSTEP SCAN (N=2). Justifique sus respuestas.

4) Un disco que posee 100 pistas (0-99) tiene su cabezal en la pista 16 ascendiendo. Se requiere saber cuándo terminaría de atender los pedidos: 6-16-24-2-10, comenzando desde el instante 0, utilizando los algoritmos C-LOOK y FSCAN si el tiempo entre pistas es 2ms y si:

a) En el instante 15ms llega un pedido en la pista 27
b) En el instante 17ms llega un pedido en la pista 27

Nota 1: Sólo tenga en cuenta el tiempo de búsqueda (tiempo entre pistas).
Nota 2: Realice los puntos a) y b) como ejercicios separados.

5) Se tiene un disco de 100 cilindros 4 platos y 20 sectores por pista. El tiempo de búsqueda (seek time) es de 2 ms. El brazo se desplaza en forma descendente en este instante y acaba de leer el bloque 177. Tiene los siguientes pedidos encolados: 186, 2226, 1470, 7000, 154.
Ordene la cola y determine el tiempo necesario para atender todos los pedidos según: SSTF y N-STEP-SCAN(colas de 3 pedidos).

6) Peter necesita leer un conjunto de archivos lo más rápido posible, muchas veces al día. Como su presupuesto no es tan alto y conoce las limitaciones de escritura de los SSD decide dividir los archivos en dos discos independientes, cada uno de 1000 pistas, 10 caras y 10 sectores por pista.

En un momento dado se tienen los siguientes pedidos lógicos (D1: disco 1, D2: disco 2): 1000 D1, 1500 D2, 1800 D1, 5000 D1, 3300 D2, 1700 D2, 3200 D2.

a) Calcule el tiempo necesario para atenderlos a todos sabiendo que el algoritmo a utilizar es FIFO y que el tiempo entre pistas es de 1 ms. La pista inicial es la 0 en ambos discos.
b) ¿Mejoraría la situación si se atendieran todos los pedidos desde un único disco usando el algoritmo SCAN?

7) Un archivo comprimido con WinRawr está dividido en dos partes, guardadas en dos discos rígidos distintos. La primera parte se encuentra en un disco de 100 pistas (que el S.O. planifica utilizando el algoritmo SCAN), y para leerla necesita atender los siguientes pedidos de pistas: 88, 60, 89, 1, 20 (que llegan a la cola en el instante 0 ms), y 60, 20 (que llegan a la cola en el instante 50 ms). La segunda parte del archivo se encuentra en un disco de 50 pistas (que el S.O. planifica utilizando el algoritmo FSCAN), y para leerla necesita atender los siguientes pedidos de pistas: 40, 20, 1 (que llegan a la cola en el instante 0 ms), y 40, 20 (que llegan a la cola en el instante 20 ms). En ambos casos, el tiempo entre pistas es de 1ms, y el brazo se encuentra en la pista 0.

a) ¿Qué tiempo lleva abrir el archivo completo, si las partes se deben leer en simultáneo?
b) WinRawr escribe en un pequeño archivo de log, siempre en el instante 30 ms, en la pista 25 del primer disco. ¿Sería conveniente mover el archivo a la pista 35 del segundo disco?.

Ejercicios de file system

1) Peter se cansó de tener Windows en su PC y decide instalar Kubuntu. Al empezar a usarlo, descubre lo maravilloso que es, y decide desafiar a un amigo que tiene Windows a que su sistema es más rápido.
Para ello, deciden competir en la velocidad a la hora de jugar al "Angry Penguins", que como se la pasa operando con archivos, implica la necesidad de tener un desempeño muy bueno en su filesystem.
Luego de instalarse el programa "Linus Toolvars", descubre que su filesystem maneja inodos, con bloques de 4 KiB y punteros de 8 bytes. El inodo está conformado con 12 punteros directos, 1 indirecto, 1 indirecto doble y 1 indirecto triple.
Analice, suponiendo que el inodo de un archivo "/trash/pagefile.sys" se encuentra en memoria, la cantidad de accesos a disco para leer:

a) El byte nro 20000112 del archivo
b) Desde el byte 0 hasta el 191233 del archivo
c) El byte nro 20000112 de un hard link al mismo archivo
d) El byte nro 60001 de un symbolic link al mismo archivo

Nota: Tenga en cuenta que su disco es un HDD (Hard Disk Drive) de 3 Gb que al parecer posee 512 cilindros, 7200 rpm, 24 cabezas y 128 sectores por pista.

2) Un File System de tipo EXT2 maneja punteros de 32 bits, bloques de 4KiB y la conformación de su inodo es de 10 punteros directos, 1 indirecto simple, 1 indirecto doble y 1 indirecto triple. En dicho File System se aloja el archivo "Resumen.pdf", de 4 MiB, el cual se lee y comprime a la mitad de su tamaño (sin borrar el original), y luego se guarda en el mismo directorio como "Resumen.tar.gz".

a) Indique cuántos accesos a disco fueron necesarios para realizar la operación.
b) Indique cuántos accesos a disco serían necesarios para leer el primer bloque de Resumen.pdf si accediéramos desde un Symbolic Link ubicado en el Escritorio

3) Un volumen se encuentra formateado con FAT 12. El disco posee un tamaño de 1GiB. Si se sabe que con el FS actual se logra referenciar todo el disco:
a) ¿Qué tan bien se utilizaría el espacio bajo este esquema si la mayoría de los archivos son de 2KiB?
b) Encuentre el formato de FS tipo FAT que permita direccionar todo el disco y que mejor se adapte a archivos de 2KiB.
c) Suponiendo que la tabla FAT se cargue en su totalidad a memoria principal, compare dicho sistema en performance con EXT2, para casos de accesos concurrentes a distintos archivos.

4) Peter posee un disco de de 500 GiB formateado con un FS Unix que posee la siguiente estructura de inodos: 15 punteros directos , 2 punteros indirectos simples , 1 puntero indirecto doble
Los bloques de datos tienen un tamaño de 8 KiB y las direcciones son de 64 bits.

En su disco posee un directorio dedicado a series y películas. En esta oportunidad, Peter se quería bajar (legalmente) la 3er temporada de Game of Thrones en HD, la cual tiene un tamaño de 13,67 GiB. Sin embargo, en el proceso de escritura del archivo a su disco se encontró con un problema por lo que el archivo no se pudo guardar, sin importar que posee más de la mitad del disco vacío. Finalmente optó por una versión de menor calidad con un tamaño de 3,9 GiB.

a) Indique y justifique qué problema tuvo Peter. Proponga dos posibles soluciones para que pueda tener la versión HD.
b) ¿Cuántas operaciones de disco se realizarán para escribir el archivo de 3,9 GiB a disco? (considere escrituras de bloques de datos y de punteros).

Nota 1: Indique claramente de qué nivel son los punteros escritos.
Nota 2: Considere que toda la temporada se encuentra en un único archivo.

5) Se deben formatear 4 discos de 32 GiB cada uno, utilizando una configuración RAID 1. Las opciones de FS a elegir son:

- FAT16, con clusters de 1MiB.
- UFS con punteros de 4 bytes, bloques de 4KiB e inodos con el siguiente formato: 2 ptrs directos, 1 ptr indirecto simple, 1 ptr indirecto doble y 1 ptr indirecto triple.

a) Seleccione una de las opciones y justifique su elección teniendo en cuenta que: se quiere minimizar la cantidad de accesos a disco para leer un archivo, sin importar el tamaño del mismo; se quiere poder aprovechar todo el tamaño del volumen.

b) Luego de utilizar el sistema por un tiempo, uno de los discos falla. Por lo tanto, se decide eliminar el RAID 1 y reconfigurar los discos. Indique en cada caso la solución a proponer (especificando la opción de FS a utilizar en cada caso):

b1- Proveer seguridad de los datos y permitir escrituras eficientes
b2- Obtener el mayor volumen posible, sin importar la seguridad de los datos

6) Peter tiene dos volúmenes en su disco rígido; el primero se encuentra formateado con FAT16, con clusters de 8KiB, y el segundo con UFS, con inodos que poseen: 10 ptrs directos, 2 ptrs indirectos simples, 1 ptr indirecto doble y 1 ptr indirecto triple (el tamaño puntero es de 8bytes) y bloques de 2 KiB. Luego de tener problemas de espacio en su primera partición comenzó a pasar sus archivos a la partición con UFS. El primer archivo a “mover”, se encuentra comprimido y su tamaño es de 1MiB. Una vez que el archivo se encuentra en la segunda partición, el mismo se descomprime (alcanzando el doble del tamaño inicial).

Considerando que la tabla FAT y la tabla de inodos se encuentran en memoria:

a) Calcule la cantidad de accesos a clusters, a bloques de datos y bloques de punteros (según corresponda) necesarios para realizar las operaciones.

b) Indique, en cada uno de los filesystems, qué modificaciones sufrieron las estructuras administrativas al realizar dichas operaciones.

7) Dado un disco con FAT32 y clusters de 12 Kb

a) Indique el tamaño máximo teórico del filesystem
b) Indique el tamaño máximo real si se formatea sobre un pendrive de 4 TB
c) ¿Cuánto espacio en disco ocuparía la FAT?
d) ¿Cuál es el máximo de fragmentación interna que podría tener el filesystem?

Nota: Recuerde que dicho fs utiliza solamente 28 bits para direccionar clusters.

8) Un filesystem de tipo Unix maneja bloques de 4KiB, punteros de 32 bits. Además se sabe que el inodo cuenta con 10 punteros directos, un puntero indirecto simple, un puntero indirecto doble y un puntero indirecto triple. Se utiliza un disco que tiene 256 GiB con sectores de 1024 bytes.

a) Calcule el tamaño máximo teórico de un archivo.
b) Cuántos accesos a disco se requieren para leer los primeros 10 MiB de un archivo.
c) Calcule el tamaño máximo teórico y real del filesystem.

9) En la VM de Ubuntu que Peter usa para el trabajo práctico, tiene un sistema de archivos ext2 con bloques de 2 KiB y tamaño de punteros de 16 Bytes. Dentro de la VM tiene la canción “Lessons.mp3” (de la banda Rush) que se encuentra direccionada de la siguiente forma: 12 bloques directos, 1 bloque indirecto completo y un bloque indirecto doble con solo 62 bloques indirectos completos.

Como quiere escuchar esa canción en mi equipo iPod, que usa FAT16 con clusters de 256 bytes, aprovecha que tiene un pendrive vacío formateado en ese sistema de archivos con clusters de 256 Bytes para pasar la canción allí.

a) ¿Podrá pasar la canción al pendrive?
b) Independientemente del punto anterior, suponga que la canción se pudo pasar. Si luego ese pendrive se monta en otra PC, indique cuántos accesos al pendrive se necesitarán para leer el byte 875424 (asuma que la FAT está en memoria, y que los sectores son de 128 bytes)

Nota: El manual del iPod dice “Tamaño disponible: todo lo que direccione el filesystem menos lo que ocupen la tabla FAT y su copia”.