



UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL de BUENOS AIRES

Cátedra de Sistemas Operativos

Código: 95 - 2027

Director de Cátedra: Carlos R. NEETZEL;

Docentes: CASAS N., DE LUCA G., ESQUIVEL N; NEETZEL C.; SCARFIELLO J.L..



UNIVERSIDAD NACIONAL DE LA MATANZA
Departamento de Ingeniería
e Investigaciones Tecnológicas

Cátedra de Sistemas de Computación II

Jefe de Cátedra: Carlos Neetzel

Docentes: Casas N., De Luca G., Rivalta F., Boettner, F.; Catalano L.;
de Lizarralde R., Flores Arellano G.P.; Volker M.; Fera J.M.; Alvarez S.R.;
Franze E.; Loiacono F.; Alessandrello G.; Hirschfeld D.;

GUÍA DE EJERCICIOS TÉORICOS DE S.O. (Recopilación 2010)

Recopilador: Carlos Neetzel

V3

ÍNDICE

Módulo 3: Planificación de Procesos y del procesador.....	3
Parte teórica <i>planificación</i>	3
Ejemplos de ejercicios resueltos.....	5
Ejercicios sin resolver.....	
Módulo 4:	
Parte teórica	
Ejemplos de ejercicios resueltos.....	
Ejercicios sin resolver.....	
Módulo 5:	
Parte teórica	
Ejemplos de ejercicios resueltos.....	
Ejercicios sin resolver.....	
Módulo 6:	
Parte teórica	
Ejemplos de ejercicios resueltos.....	
Ejercicios sin resolver.....	
Módulo 7:	
Parte teórica	
Ejemplos de ejercicios resueltos.....	
Ejercicios sin resolver.....	
Módulo 8:	
Parte teórica	
Ejemplos de ejercicios resueltos.....	
Ejercicios sin resolver.....	
Ejercicios Combinados.....	

AGRADECIMIENTO:

La Cátedra de Sistemas Operativos agradece especialmente la colaboración de los alumnos, ayudantes y docentes de la Universidad Tecnológica Nacional (Fac. Regional Bs. As.) y de la Universidad Nacional de La Matanza, que han propuesto los ejercicios y las soluciones de los módulos del programa de estudio de Sistemas Operativos (UTN-FRBA) y Sistemas de Computación II (UNLaM). El recopilador agradece la valiosa colaboración y espera que esta sea de utilidad a nuestros alumnos.

No se asume la responsabilidad de que los ejercicios resueltos estén “bien resuelto” por lo que se recomienda resolverlos y comparar los resultados y frente a discrepancias se propone consultar con los docentes para su aclaración.

La presente guía está disponible para todos aquellos alumnos que previamente decidan realizar los ejercicios de la Cátedra, puedan comprobar sus resultados y así servir de guía de resolución y corrección de los mismos. Se propone como ejercitación práctica adicional a la de la bibliografía de la asignatura. Constituyen ejercicios modelos recopilados de guías anteriores y libros o exámenes finales ya tomados, por lo que es importante utilizarlos solo como una guía de estudios actualizada.

Por último, sugerimos que los alumnos traten de realizar un esfuerzo personal en resolver todos los ejercicios de la guía sin la ayuda de las soluciones propuestas, dado que el facilismo y el “copiarse” no aporta un aprendizaje.

ACLARACIÓN:

Esta Guía es confeccionada exclusivamente con la finalidad didáctica para ser usada por los alumnos como ejercitación de la teoría de Sistemas Operativos o Sistemas de Computación. Para cualquier otro uso, no se asume ninguna responsabilidad.

Resumen de Conceptos Básicos sobre S.O.

Los conceptos aquí explicados dan los conocimientos, términos y fórmulas necesarios para poder resolver los ejercicios que siguen a continuación. Estos son pequeños puntos teóricos que facilitan la resolución de los problemas; desde ya que se recomienda completar esta teoría con la de alguna bibliografía.

Módulo 3: Planificación de Procesos y del procesador.

PARTE TEORICA PLANIFICACIÓN:

Comenzando por este tema, podemos decir que en un sistema encontramos básicamente 4 tipos de planificaciones: Una planificación a largo plazo (la cual se encarga de determinar y organizar los nuevos programas que van a ser admitidos al sistema), una a mediano plazo (encargada del swapping (intercambio) de los procesos, entre la memoria central y la virtual), una planificación a corto plazo (el "dispatcher", quien determina cual es el próximo proceso que va a tomar la CPU), y finalmente la planificación de E/S.

En el largo plazo:

Para los ejercicios de planificación propiamente dicho, en el largo plazo tenemos los siguientes algoritmos:

FCFS: Es FIFO; es decir, los procesos entran a la cola READY en el orden que llegaron a la cola.

Shortest Job Next (SJN): Es un algoritmo para los procesos mas cortos que llegan a la cola de listos, éstos se ordenan según el tamaño (o Tiempo) mas corto queda a la cabeza de la cola.

Prioridades: Cada proceso se le asigna una prioridad y se ordena en la cola de mayor a menor prioridad para pasar a usar la CPU en ese orden. Cuando ingresan procesos de igual prioridad se colocan en la cola en orden de llegada (FIFO)

En el corto plazo:

Por otro lado, en la planificación a corto plazo considera el uso del procesador, que se basan en una serie de algoritmos, pero antes, una pequeña aclaración acerca de algunos términos a utilizar:

* **Non – preemptive = Apropiativo o NO EXPROPIATIVO en el uso del procesador**

* **Preemptive = No – Apropiativo o EXPROPIATIVO en el uso del procesador**

Los conflictos el Sistema Operativo los resolverá atendiendo en el siguiente orden de prioridad salvo que el enunciado indique expresamente otra cosa: 1º-Excepciones, luego 2º.-Interrupciones de Hardware y por último 3º.-Systems Call.

Cuando un proceso tiene asignado un quantum de tiempo y llega una interrupción. Esta puede ser del reloj o de un dispositivo de Entrada – Salida. Entonces tenemos las siguientes posibilidades:

1. El tiempo de atención de la interrupción es despreciable con respecto al Q → consideramos que el proceso tienen su Q completo y .
2. El tiempo de atención no es despreciable y los descuenta del Q del proceso, es decir el proceso pierde ese tiempo de ejecución.
3. El tiempo de atención no es despreciable, pero salva el registro del reloj cuando llega la int de hardware y no descuenta tiempo del Q al proceso, es decir el proceso no pierde ese tiempo de ejecución.

Estas consideraciones generalmente se aclaran en el enunciado

* **CPU Bursts** = Utilización del procesador entre dos E/S. Obviamente, cuando un proceso comienza, el tiempo utilizado hasta la primera E/S también es un Burst de CPU.

Bien, veamos entonces los algoritmos del corto plazo:

- **FCFS**: Es el famoso FIFO; es decir, los procesos usan la CPU en el orden que llegaron. Es un algoritmo apropiativo, es decir, a medida que llegan los procesos van usando la CPU y cuando hacen uso de ésta, no la abandonan hasta que se bloqueen por E/S o finalicen.
- **Round Robin**: El Round Robin viene siempre acompañado por un slice de tiempo (Quantum), es decir, un tiempo determinado que tienen los procesos para ejecutar; finalizado ese tiempo el proceso abandona la CPU. Debido a esta característica, el algoritmo es no apropiativo. Obviamente que el proceso puede abandonar antes la CPU en caso de que finalice o se bloquee por una E/S; caso contrario (que se le termina el tiempo), vuelve a la cola de listos.
- **Round Robin Virtual**: Es muy similar al anterior, pero hace esta diferencia: en vez de tener una cola de listo, tiene dos; una cola "clásica" para los procesos listos y una nueva cola de listos al que van a parar los procesos cuando finalizan su E/S. Esta segunda cola tiene mayor prioridad que la primera. El objetivo de este algoritmo es realizar un uso mas equitativo de la CPU cuando existen procesos que tienen mas burst de CPU que de E/S, y así mismo hay procesos que tienen mas burst de E/S que de procesador.
- **Shortest Process Next (SPN)**: Es un algoritmo apropiativo. Cuando los procesos llegan a la cola de listos, éstos se ordenan según quien tiene el menor burst de CPU. El que queda a la cabeza, es quien pasa a usar la CPU.
- **Shortest Remaining Time (SRT) First o Next**: Es muy similar al anterior, pero este sí es un algoritmo no apropiativo. En la cola se siguen ordenando según quien tenga el menor burst de CPU, pero si se da el caso de que en un momento hay un proceso ejecutando y llega uno nuevo, se determina si el proceso que está usando la CPU actualmente le falta menos del uso que va a hacer el nuevo proceso; si es menor, sigue ejecutando, si le queda más, el proceso es desalojado del procesador y llevado nuevamente a la cola de listos.
- **Prioridades**: Pude tomarse como apropiativo o no apropiativo, salvo que lo expresen específicamente. Cada proceso tiene una prioridad y se orden en la cola de mayor a menor prioridad y pasan a usar la CPU en ese orden. En el caso de ser apropiativo, el proceso que pasa a ser uso del procesador queda ahí hasta que se bloquee o finalice. En caso de ser no apropiativo, si llega un proceso con mayor prioridad, desaloja al que actualmente está haciendo uso del procesador, y este nuevo proceso toma la CPU.
- **Feedback**: Es como el algoritmo de prioridades, pero se tiene una cola por cada una de las prioridades. Cada cola puede manejarse por un algoritmo Round Robin, teniendo un time slice de tiempo para cada cola que aumenta a medida que disminuye la prioridad, es decir, la cola de mayor prioridad va a tener un quantum menor al de la prioridad inmediata inferior y así sucesivamente; Esto se realiza para que los procesos que llegan a la última cola son los mas viejos y por lo tanto se les da mas tiempo para que finalicen.

Ejemplo para el corto plazo:

Para **decidir** que proceso de la cola de procesos listos se ejecutará, se usan los llamados algoritmos de planificación del CPU, y mediante estos algoritmos queda determinado el orden de ejecución de dichos procesos.

Existen dos tipos de planificación según si el proceso en ejecución libera el CPU por "decisión propia" (ya sea porque se completó su ejecución o por necesidad de E/S) o sea los llamados Algoritmos non preemptive (sin reemplazo o apropiativos); o si la liberación del CPU se hace aunque el proceso en ejecución pueda seguir ejecutándose, es decir, se puede liberar el CPU por tics del reloj, por interrupciones, por la existencia de un proceso de menor de mayor prioridad en la cola de procesos listos, etc; o sea los llamados Algoritmos preemptive (con reemplazo o expropiativos).

Algoritmos non preemptive (sin reemplazo)

FCFS (First Come First Served)

- ✓ El primer proceso en solicitar la CPU es el primero en recibir la asignación de la misma.

- ✓ Los procesos que llegan al sistema se encolan en una cola FIFO de manera que se atiendan por orden de llegada.

Ventajas:

- ✓ Fácil de implementar.
- ✓ Tiene una política justa ya que se atienden los procesos a medida que van llegando.

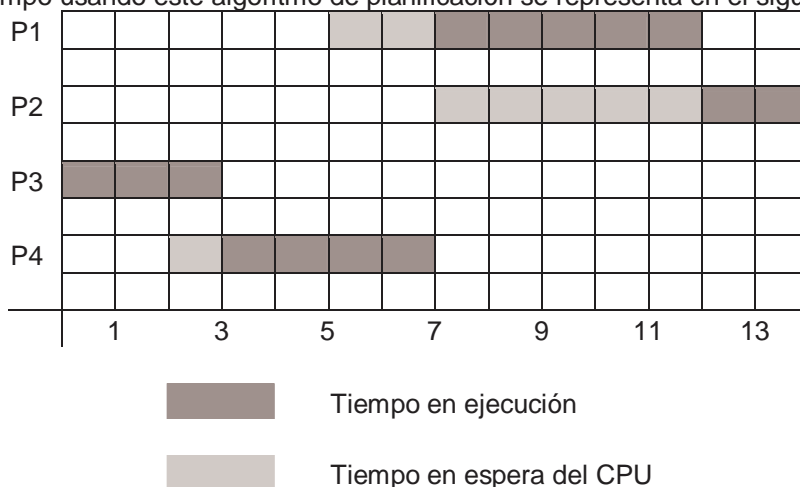
Desventajas:

- ✓ Tiene un tiempo medio de espera elevado.
- ✓ Tiene un bajo nivel de uso del CPU.
- ✓ Tiene un tiempo de respuesta pobre en procesos cortos con esquemas con mucha carga.

Supongamos que llegan al sistema 4 procesos en el orden que se presenta a continuación y con la longitud que se indica

Proceso	Duración	Tiempo de llegada
P3	3	0
P4	4	2
P1	5	5
P2	2	7

La ejecución en el tiempo usando este algoritmo de planificación se representa en el siguiente gráfico.



Como se puede ver en el ejemplo, se atiende a los procesos por orden de llegada y los procesos no liberan el CPU hasta que se complete su ejecución o hasta que ellos mismos soliciten E/S.

SPF (Shortest Process First)

- ✓ Asocia a cada trabajo la longitud de su siguiente ráfaga de procesador que se calcula generalmente tomando la media aritmética exponencial de los CPU burst (ráfagas de CPU) previos.
- ✓ Se asigna el procesador a aquel proceso que tenga el CPU burst más corto.

Ventajas:

- ✓ Se beneficia a los procesos cortos.

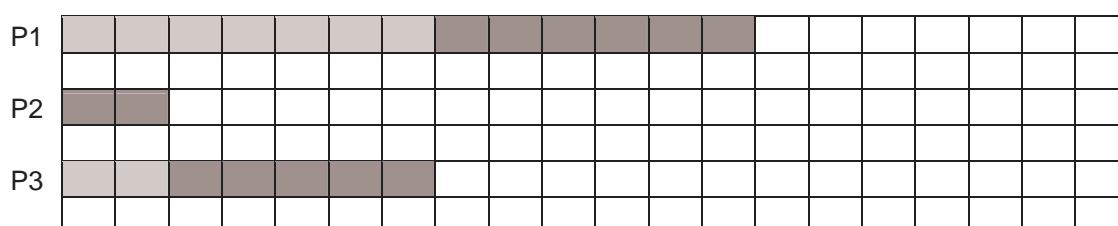
Desventajas:

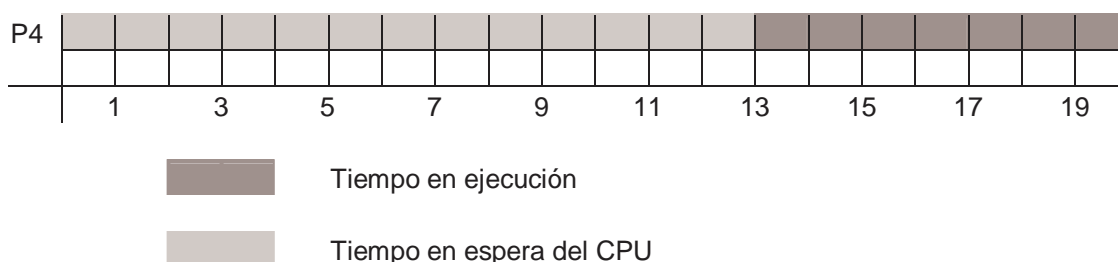
- ✓ Se pierde tiempo en efectuar el cálculo con el cual se obtiene el próximo CPU burst de los procesos.
- ✓ Puede haber inanición de los procesos largos que tiene CPU burst largos.

Supongamos que llegan al sistema 4 procesos en el orden que se presenta a continuación y con la longitud que se indica

Proceso	Duración	Tiempo de llegada
P1	6	0
P2	2	0
P3	5	0
P4	7	0

La ejecución en el tiempo usando este algoritmo de planificación se representa en el siguiente gráfico.





Como se puede ver en el ejemplo se ejecutan primero aquellos procesos más cortos.

Planificación por prioridad

- ✓ Cada proceso tiene asociada una prioridad, que puede ser interna (o dinámica) o externa (o estática). Las dinámicas son aquellas que son modificables por el sistema operativo en tiempo de ejecución. Las estáticas son puestas arbitrariamente por aquella persona que ingresó el trabajo.
- ✓ Se ejecuta primero el proceso con mayor prioridad.

Ventajas:

- ✓ Se puede dar mayor velocidad de ejecución a procesos con mayor prioridad.
- ✓ No beneficia sólo a procesos cortos como SPF, sino que un proceso largo puede ejecutarse antes que uno corto y viceversa.

Desventajas:

- ✓ Puede haber inanición de procesos con prioridades bajas si constantemente entran al sistema procesos con altas prioridades. Solución: Las prioridades de los procesos pueden irse subiendo, de manera que los de prioridad baja tengan una prioridad mayor y puedan ejecutarse.

Supongamos que llegan al sistema 4 procesos en el orden que se presenta a continuación y con la longitud y prioridad que se indica

Proceso	Duración	Tiempo de Llegada	Prioridad
P1	2	0	2
P2	3	0	1
P3	4	0	5
P4	5	0	3

La ejecución en el tiempo usando este algoritmo de planificación se representa en el siguiente gráfico.



Como se puede ver en el cuadro, los procesos se ejecutan de acuerdo a su prioridad, (aca se tomo como mayor prioridad, la prioridad con el mayor número, pero esto en otros sistemas puede ser inverso) y no se tuvo en cuenta la longitud del proceso, como se puede ver P4 y P3 son más largos que P1 y se ejecutaron antes que P1, sin embargo P2 es también más largo que P1 pero se ejecutó después debido a que tiene menor prioridad.

HRRN (Highest Response Ratio Next)

- ✓ Se tiene en cuenta la tasa de respuesta de los procesos que es una relación entre el tiempo de retorno y el tiempo de servicio.
- ✓ Se ejecuta primero el proceso que tenga la mayor tasa de respuesta.
- ✓ La tasa de respuesta se calcula de la siguiente manera

$$R = \text{tiempo consumido esperando por el CPU} + \text{Tiempo de servicio esperado}$$

Tiempo de servicio esperado

Ventajas:

- ✓ Si bien inicialmente favorece a los procesos cortos (ya que tienen menor tiempo de servicio) el envejecimiento de los procesos incrementa el valor de R

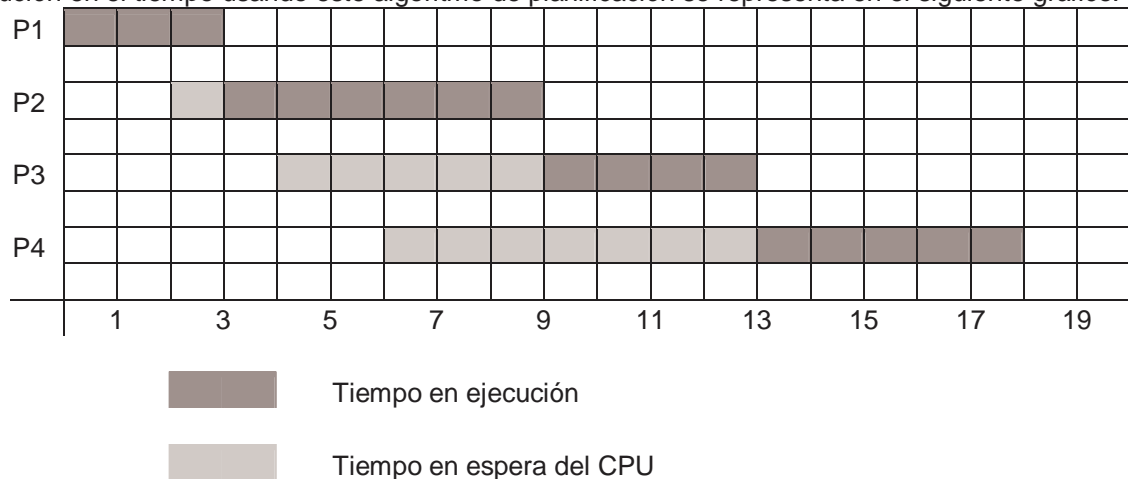
Desventajas:

- ✓ No posee grandes desventajas.

Supongamos que llegan al sistema 4 procesos en el orden que se presenta a continuación y con la longitud que se indica

Proceso	Duración	Tiempo de llegada
P1	3	0
P2	6	2
P3	4	4
P4	5	6

La ejecución en el tiempo usando este algoritmo de planificación se representa en el siguiente gráfico.



Algoritmos preemptive

RR (Round Robin)

- ✓ Los procesos son encolados en una cola circular, añadiéndose al final de la cola aquellos procesos que acaban de ingresar.
- ✓ Cuando se tiene que seleccionar un proceso, se lo hace siempre en un sentido de la cola circular.
- ✓ Existe lo que se llama el quantum que es el tiempo que los procesos podrán usar ininterrumpidamente el CPU, al vencerse ese quantum se elige otro proceso.

Ventajas:

- ✓ No existe inanición a menos que entren constantemente procesos nuevos en la cola.
- ✓ No se beneficia a ningún proceso en cuanto a longitud, vejez, tasa de respuesta.

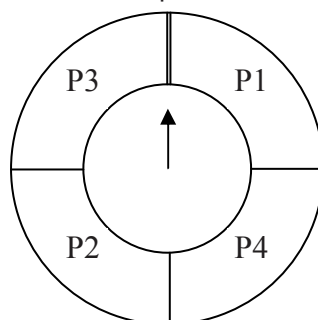
Desventajas:

- ✓ Si el quantum es muy grande se parece mucho a FCFS.
- ✓ Si el quantum es muy chico se produce mucho overhead.
- ✓ Si hay mezcla de procesos, aquellos limitados por E/S y los limitados por ráfagas del CPU, se beneficiará a los limitados por ráfagas de CPU.

Supongamos que llegan al sistema 4 procesos en el orden que se presenta a continuación y con la longitud que se indica

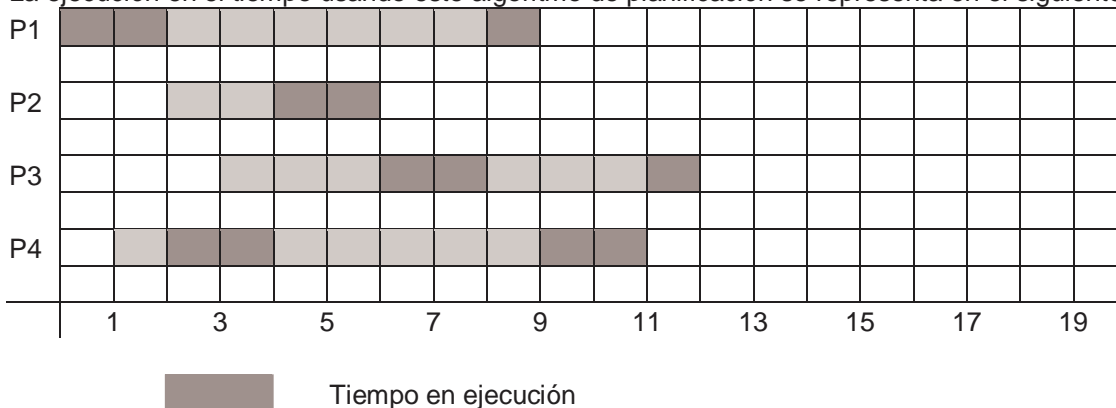
Proceso	Duración	Tiempo de llegada
P1	3	0
P2	2	2
P3	3	3
P4	4	1

Y se tiene un quantum de 2; la cola circular de procesos listos quedaría de la siguiente manera:



La flecha indica el próximo proceso a ejecutarse y gira en sentido de las manecillas del reloj, por lo que el primer proceso a ejecutarse será P1, el segundo P4 y así sucesivamente.

La ejecución en el tiempo usando este algoritmo de planificación se representa en el siguiente gráfico.



Aquí se ve como se alterna el uso del CPU entre los procesos, donde cada cambio, en este caso, es por vencimiento del quantum.

SRT (Shortest remaining time)

- ✓ Se ejecuta el proceso que tenga el menor tiempo restante de ejecución.
- ✓ Un proceso en ejecución puede ser “expulsado” del CPU si llega otro cuyo tiempo restante de ejecución es menor al que el que se está ejecutando actualmente.

Ventajas:

- ✓ Bueno performance.
- ✓ Es muy eficiente ya que no hay mucho overhead porque las interrupciones no son producidas por reloj.

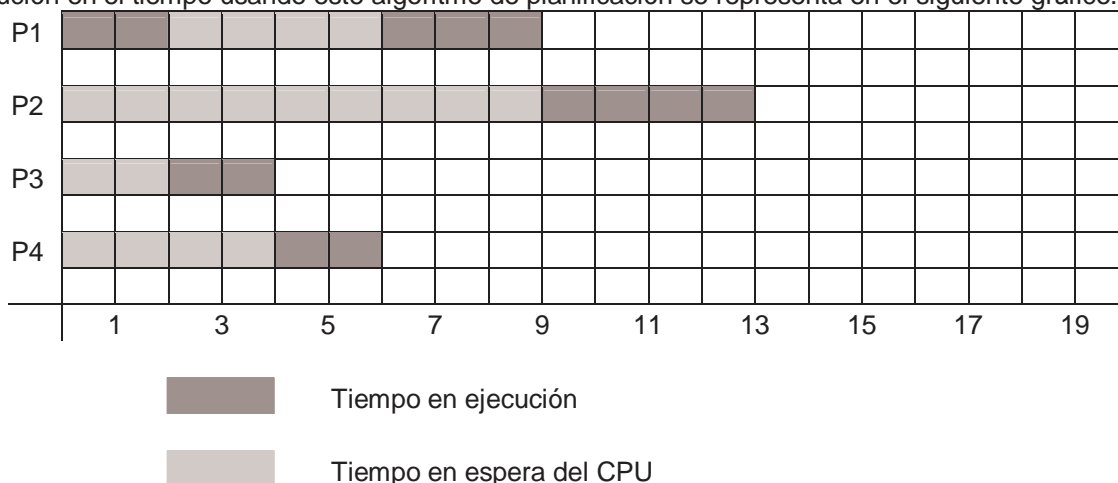
Desventajas:

- ✓ Produce inanición si llegan procesos con

Supongamos que llegan al sistema 4 procesos en el orden que se presenta a continuación y con la longitud que se indica

Proceso	Duración	Tiempo de llegada
P1	5	0
P2	5	1
P3	2	2
P4	2	3

La ejecución en el tiempo usando este algoritmo de planificación se representa en el siguiente gráfico.



Aquí se ve como al llegar un proceso cuyo tiempo de ejecución restante hasta su completitud es menor al del proceso en ejecución, el proceso de menor tiempo toma control del CPU.

HRRN (Highest Response Ratio Next)

La política de planificación es igual a la del HRRN non preemptive, sólo que en este caso, el proceso en ejecución puede ser expulsado del CPU por la llegada de algún proceso cuya tasa de respuesta sea mayor a la tasa del proceso que se está ejecutando actualmente.

Planificación con colas múltiples y realimentación.

- ✓ Muy útil cuando no es posible determinar el tiempo de ejecución de los procesos
- ✓ Posee quantums de tiempo.
- ✓ Tiene un mecanismo de prioridades dinámico.
- ✓ Cada proceso que llega se coloca en la cola de mayor prioridad y luego de cada ejecución de si mismo (es decir, después que se le venza el quantum o pida E/S), se coloca en la cola de menor prioridad que en la que estaba.
- ✓ A la cola de menor prioridad le sigue la cola de mayor prioridad.
- ✓ Dentro de las colas se usa el algoritmo FCFS, excepto en la última cola que se utiliza el algoritmo RR.
- ✓ Cada cola tiene su propio quantum de tiempo siendo menor el quantum de la cola de mayor prioridad.

Ventajas:

- ✓ Este esquema deja los procesos limitados por E/S y los procesos interactivos en la cola de prioridad más alta de manera que se ejecuten más rápidamente.

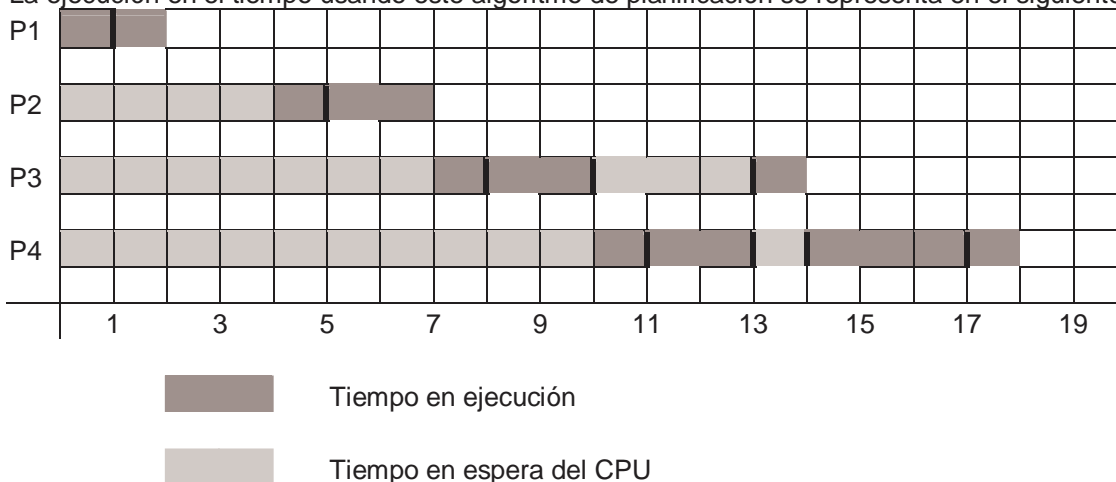
Desventajas:

- ✓ El tiempo de retorno de los procesos mayores puede alargarse de forma alarmante.
- ✓ Ocasión inanición si entran procesos nuevos frecuentemente.

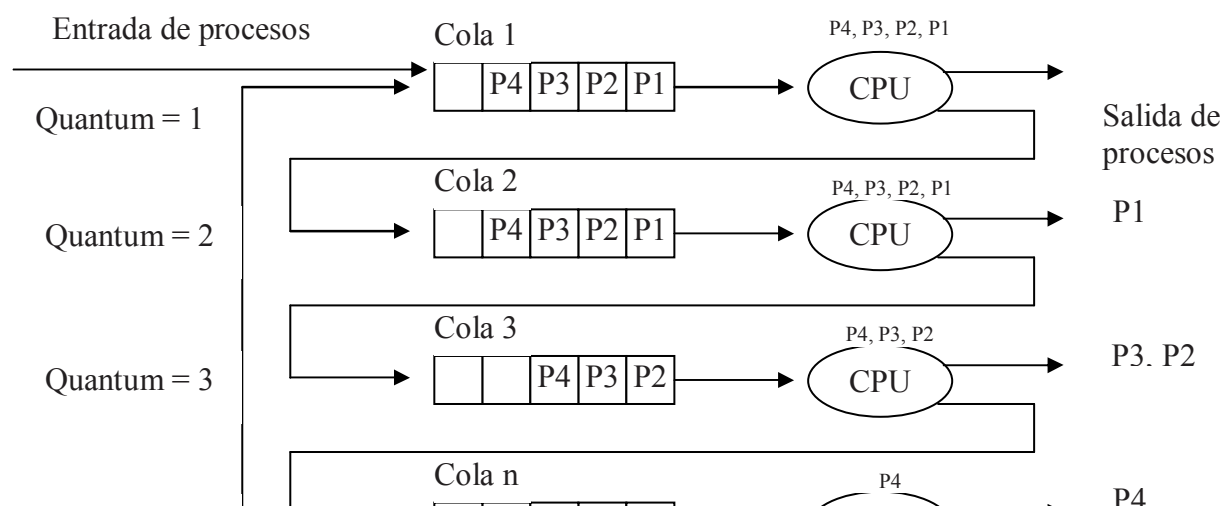
Supongamos que llegan al sistema 4 procesos en el orden que se presenta a continuación y con la longitud que se indica

Proceso	Duración	Tiempo de llegada
P1	2	0
P2	3	4
P3	4	6
P4	5	9

La ejecución en el tiempo usando este algoritmo de planificación se representa en el siguiente gráfico.



En el ejemplo anterior la línea más gruesa marca un vencimiento del quantum, en la gráfica de las colas se ve como los procesos van cambiando de cola, yendo cada vez a colas de menor prioridad. Las líneas más gruesas significan un vencimiento del quantum de tiempo de la cola en la que el proceso se encontraba antes de pasar a ejecución, y su correspondiente cambio de cola.



EJEMPLOS DE EJERCICIOS RESUELTOS:

Ejercicio RM3:1:

Sean los cinco procesos descritos en la tabla siguiente:

Proceso	Instancia de creación	Tiempo de CPU
A	3	1
B	0	5
C	1	4
D	6	3
E	9	2

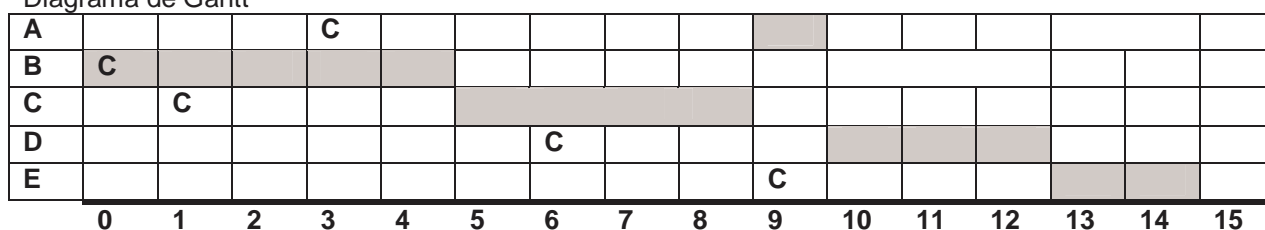
Para un algoritmo de planificación que utiliza una política FIFO

- Diagrama de Gantt
- Tiempo de respuesta medio
- Tiempo de espera medio
- Tiempo de ejecución medio
- Tiempo de servicio medio
- Índice de servicio medio

Resolución

Cola de Listos para FIFO: B, C, A, D, E

Diagrama de Gantt



C = Instante de llegada al sistema

b) T(de respuesta medio)

TfA=9, TfB=4, TfC=8, TfD=12, TfE=14

$$T(\text{de respuesta medio}) = (9+4+8+12+14)/5 = 9,4$$

c) T(espera medio)

TeA=6, TeB=0, TeC=4, TeD=4, TeE=4

$$T(\text{espera medio}) = \frac{6+0+4+4+4}{5} = 3,6$$

d) Tiempo de ejecución medio = $(1+5+4+3+2)/5 = 3$

e) Tiempo de servicio medio

n=5

$$\text{Tiempo de servicio medio} = \frac{1}{n} \sum_{i=1}^n (tf_i - ti_i)$$

$$= 0,2[(9-3)+(4-0)+(8-1)+(12-6)+(14-9)] = 0,2(6+4+7+6+5) = 5,6$$

e) Índice de servicio medio = (Tiempo de ejecución medio) / (Tiempo de servicio medio) =

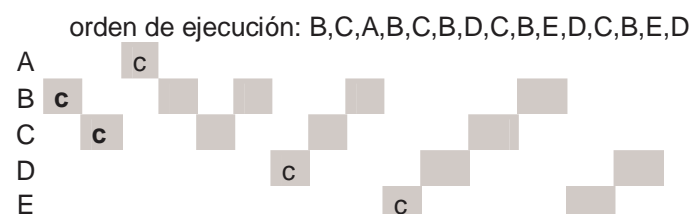
$$= 3 / 5,6 = 0,53$$

Ejercicio RM3: 2

Utilizando los valores del problema de la tabla anterior, calcular los tiempos medios de espera y respuesta para el algoritmo Round Robin con cuantums de 1 y 2. Considere que el tiempo de interrupción y context switch es cero:

Este es un algoritmo Preemptive, donde se otorga durante un quantum de tiempo, el procesador al proceso que sigue en la ronda

- Round-Robin con quantum q=1.



T 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19

T=0: la cola de listos (Ready) solo tiene a B y este pasa a ejecutar usando 1 quantum

T=1: se le acaba el quantum de tiempo al B y llega C. Como la interrupción de reloj tiene prioridad, entonces B pasará a la cola de listos y C pasa a ejecutar un quantum (este intercambio se conoce como context switch)

T=2: se produce el intercambio porque a C se termina su quantum entonces se produce la interrupción del reloj, entra a ejecutar el kernel haciendo el context switch poniendo a C en la cola de Ready y selecciona a A que llegó a la cola de Ready dándole un quantum y finaliza.

T=3: se selecciona a B que pasa a ejecutar nuevamente un quantum.

T=4: se produce el intercambio entre B y C quedando B en cola de listos y C ejecutando un quantum.

T=5 se produce el intercambio C y B, quedando C en cola de listos y B ejecutando.

T=6 se produce el intercambio B y D que llegó a la cola y es seleccionado. Quedando C, B en cola de listos.

T=7 se produce el intercambio D con C. Quedan B y D en cola de listos y C ejecutando.

T=8 se produce el intercambio C con B. Quedan C y D en cola de listos y B ejecutando.

T=9 Entra E y se produce el intercambio entre B y E. Queda B y D en cola de listos y E ejecuta un quantum.

T=10 se produce el intercambio E con D. Queda D ejecutando un Quantum y E en la cola de listos junto con B,

T=11 se produce el intercambio D con C. Quedan B y D en cola de listos y C ejecutando un quantum hasta finalizar.

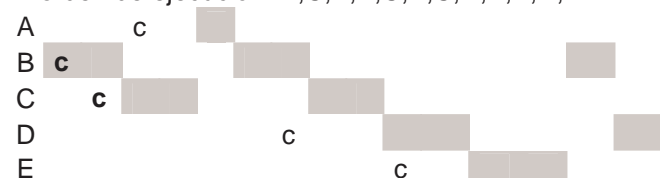
T=12 se le acaba el quantum de tiempo a C y finaliza intercambiando con B que ejecuta un quantum y finaliza. Quedan en la cola D y E

T=13 se selecciona a E que ejecuta un quantum y finaliza.

T=14 se selecciona a D que ejecuta un quantum y finaliza. Queda vacía la cola de listos y no se continúa ejecutando ya que no hay procesos en cola de listos.

b) Rond-Robin con quantum q=2.

orden de ejecución: B,C,A,B,C,B,C,D,E,B,D,



T 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19

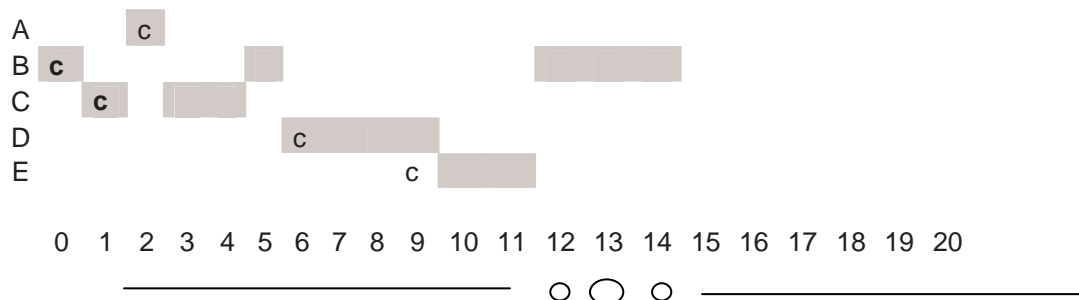
Ejercicio RM3: 3

Dada la siguiente tabla:

Proceso	Instancia de creación	Tiempo de CPU
A	3	1
B	0	5
C	1	4
D	6	3
E	9	2

Dibujar un Diagrama de Gantt para los procesos de la tabla utilizando el algoritmo "Primero el de tiempo restante menor" (SRTF).

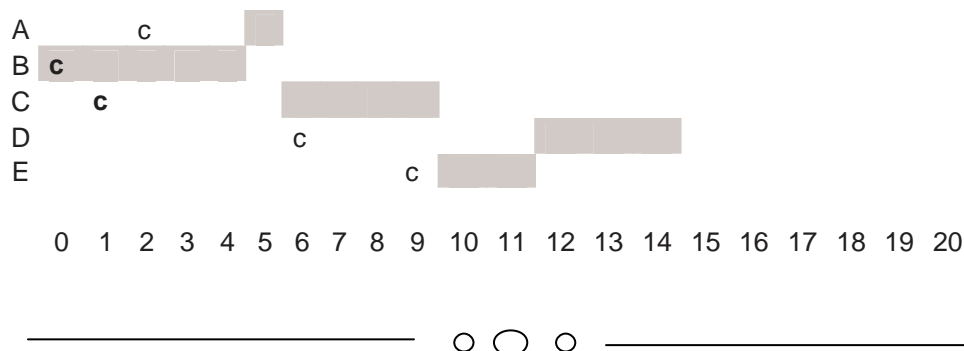
Este es un algoritmo Preemptive, donde se otorga el procesador al que menor tiempo restante estimado que tiene el proceso. Considere que el tiempo de interrupción, la ejecución del S.O. y context switch es cero



Ejercicio RM3: 4

Utilizando la tabla siguiente, dibujar un diagrama de GANTT para el algoritmo con prioridades fijas
OBSERVACIÓN: Como las prioridades son fijas asumimos que es un algoritmo NonPreemptive y Considere que el menor valor de prioridad es la de maxima prioridad en el sistema. Considere que el tiempo de interrupcion, la ejecución del S.O. y context switch es cero

Proceso	Instancia de creación	Tiempo de CPU	Prioridad
A	3	1	2
B	0	5	1
C	1	4	4
D	6	3	5
E	9	2	3



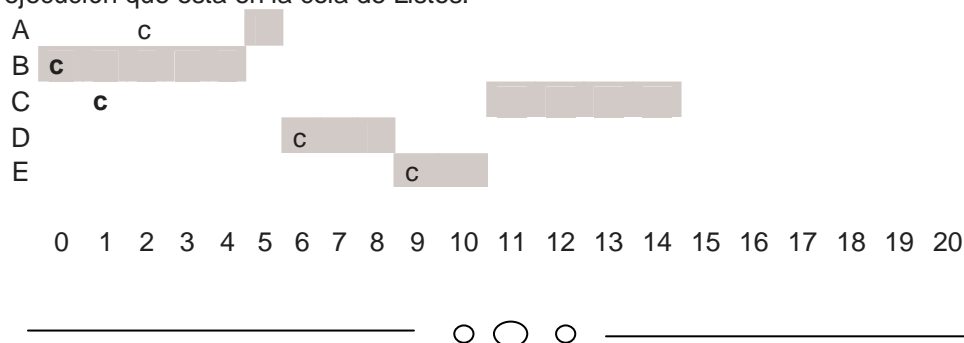
Ejercicio RM3: 5

Dada la siguiente tabla:

Proceso	Instancia de creación	Tiempo de CPU
A	3	1
B	0	5
C	1	4
D	6	3
E	9	2

Dibujar un Diagrama de Gantt para los procesos de la tabla utilizando el algoritmo "Primero el Proceso más corto (SPF). Considere que el tiempo de interrupcion, la ejecución del S.O. y context switch es cero

Este es un algoritmo Non Preemptive, donde se otorga el procesador al proceso que menor tiempo de ejecución que esta en la cola de Listos.



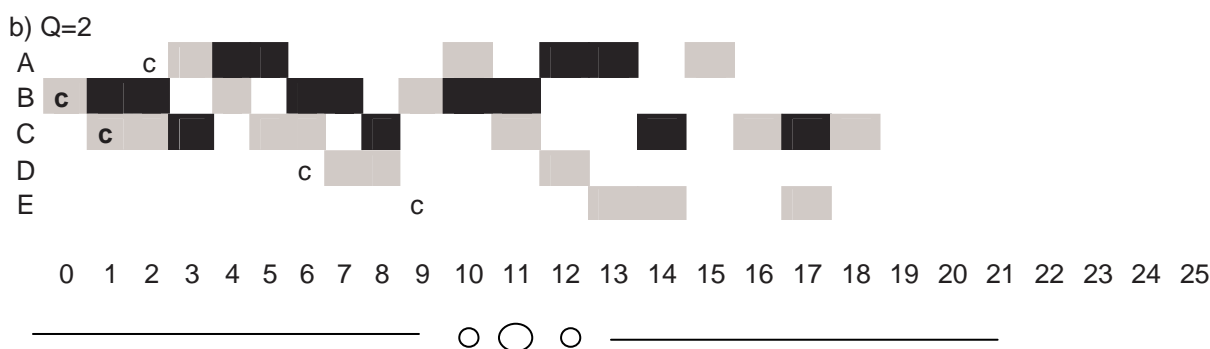
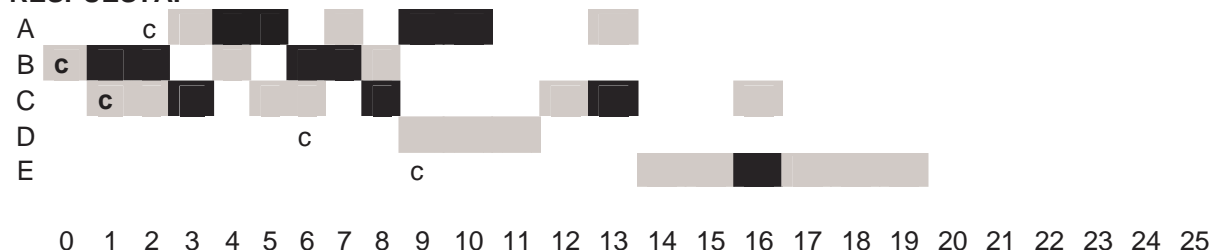
Ejercicio RM3: 6

Consideremos los procesos cuyo comportamiento se recoge en la tabla siguiente:

Proceso	Tiempo creación	Comportamiento						
		CPU	Bloqueo	CPU	Bloqueo	CPU	Bloqueo	CPU
A	2	1	2	1	2	1	-	-
B	0	1	2	1	2	1	-	-
C	1	2	1	2	1	1	1	1
D	6	3	-	-	-	-	-	-
E	9	2	1	3	-	-	-	-

Dibujar el diagrama de Gantt para los algoritmos a) FIFO (Non Preemptive) y b) Round-Robin con q=2 preemptive) y considere que el bloqueo se produce por una Entrada – Salida sobre una impresora. Considere que el tiempo de interrupcion, la ejecución del S.O. y context switch es cero

RESPUESTA:



Ejercicio RM3: 7

HRRN (Highest Response Ratio Next)

- ✓ Se tiene en cuenta la tasa de respuesta de los procesos que es una relación entre el tiempo de retorno y el tiempo de servicio.
- ✓ Se ejecuta primero el proceso que tenga la mayor tasa de respuesta.
- ✓ La tasa de respuesta se calcula de la siguiente manera

$$R = \frac{\text{tiempo consumido esperando por el CPU} + \text{Tiempo de servicio esperado}}{\text{Tiempo de servicio esperado}}$$

Ventajas:

- ✓ Si bien inicialmente favorece a los procesos cortos (ya que tienen menor tiempo de servicio) el envejecimiento de los procesos incrementa el valor de R

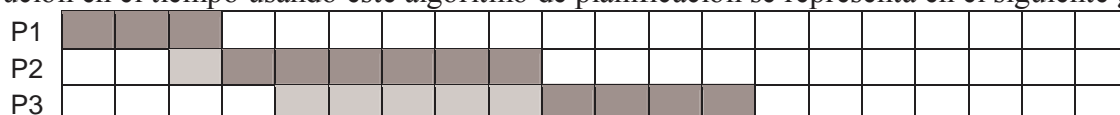
Desventajas:

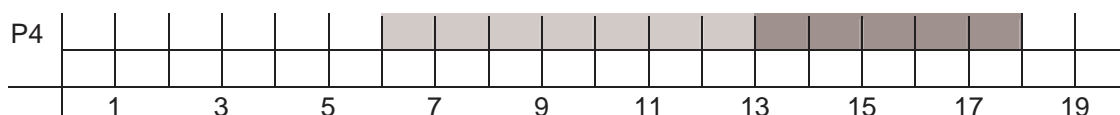
- ✓ No posee grandes desventajas.


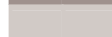
Supongamos que llegan al sistema 4 procesos en el orden que se presenta a continuación y con la longitud que se indica

PROCESO	Tiempo de llegada	Duración
P1	0	3
P2	2	6
P3	4	4
P4	6	5

La ejecución en el tiempo usando este algoritmo de planificación se representa en el siguiente gráfico.





 Tiempo en ejecución
 Tiempo en espera del CPU

Ejercicio RM3: 8

Planificación con colas múltiples o realimentación.

- ✓ Muy útil cuando no es posible determinar el tiempo de ejecución de los procesos
- ✓ Posee quantums de tiempo.
- ✓ Tiene un mecanismo de prioridades dinámico.
- ✓ Cada proceso que llega se coloca en la cola de mayor prioridad y luego de cada ejecución de si mismo (es decir, después que se le venza el quantum o pida E\S), se coloca en la cola de menor prioridad que en la que estaba.
- ✓ A la cola de menor prioridad le sigue la cola de mayor prioridad.
- ✓ Dentro de las colas se usa el algoritmo FCFS, excepto en la última cola que se utiliza el algoritmo RR.
- ✓ Cada cola tiene su propio quantum de tiempo siendo menor el quantum de la cola de mayor prioridad.

Ventajas:

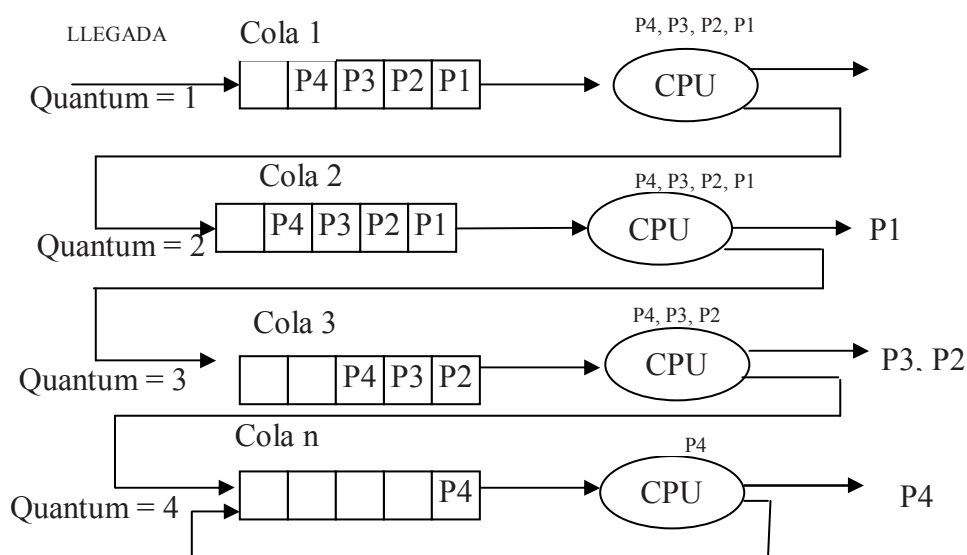
- ✓ Este esquema deja los procesos limitados por E\S y los procesos interactivos en la cola de prioridad más alta de manera que se ejecuten más rápidamente.

Desventajas:

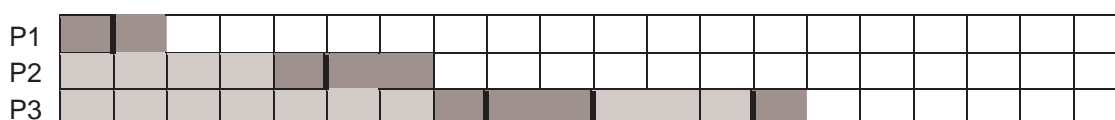
- ✓ El tiempo de retorno de los procesos mayores puede alargarse de forma alarmante.
- ✓ Ocasión inanición si entran procesos nuevos frecuentemente.

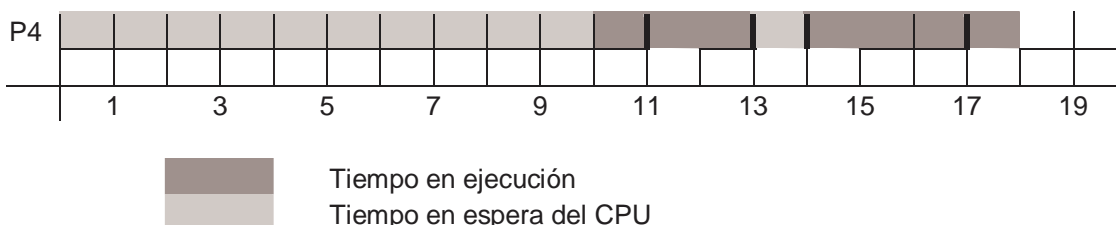
Supongamos que llegan al sistema 4 procesos en el orden que se presenta a continuación y con la longitud que se indica

PROCESO	Tiempo de llegada	Duración
P1	0	2
P2	4	3
P3	6	4
P4	9	5



La ejecución en el tiempo usando este algoritmo de planificación se representa en el siguiente gráfico.





En el ejemplo anterior la línea más gruesa marca un vencimiento del quantum, en la gráfica de las colas se ve como los procesos van cambiando de cola, yendo cada vez a colas de menor prioridad. Las líneas más gruesas significan un vencimiento del quantum de tiempo de la cola en la que el proceso se encontraba antes de pasar a ejecución, y su correspondiente cambio de cola.

Ejercicio RM3: 9 Final 22-07-2006

19 de Julio de 2006 – 9 PM, en una importante empresa de telefonía celular en EEUU. Nick, el Administrador de servidores, estuvo monitoreando un nuevo servidor que se compró para dar servicio de mensajes de texto a los clientes, cuyo software fue desarrollado por Tom. Durante dicho monitoreo, Nick ve que los indicadores de performance del sistema no son los correctos, por lo que decide realizar un análisis de la situación. Durante los primeros 15 minutos de análisis nota que está corriendo en el sistema operativo XP un proceso que no es fácilmente identificable, ya que el mismo ejecuta de forma esporádica, lo que hace que afecte a los procesos que se encuentran corriendo en el servidor. Dicha situación es preocupante debido que la Compañía no podrá brindar un buen servicio a los clientes, justo cuando mayor es la utilización de este tipo de servicio, debido a la víspera del Día del Amigo.

Es por ello, que Nick decide llamar a Tom para ver si él le puede dar una solución. Lo que él le indica es que el sistema operativo utiliza un planificador Round Robin con un quantum de 3 ciclos con una única cola de bloqueados que atiende por FIFO. Esta cola de bloqueados tiene prioridad por sobre la de listos. A su vez, Tom le envía por fax a Nick el siguiente detalle con la ejecución de los 4 procesos que ejecuta el SW:

P1	P2	P3		RCA	Referencias
		ULT A	ULT B		
0	1	2		-----	T. Llegada en Ciclos
3	4	1	3	4	CPU (En Ciclos)
6	2	2	2		IO (En Ciclos)
1	2	1	1	4	CPU (En Ciclos)
2	3				IO (En Ciclos)
1	1				CPU (En Ciclos)

[illegible]

 Ejecución  I/O

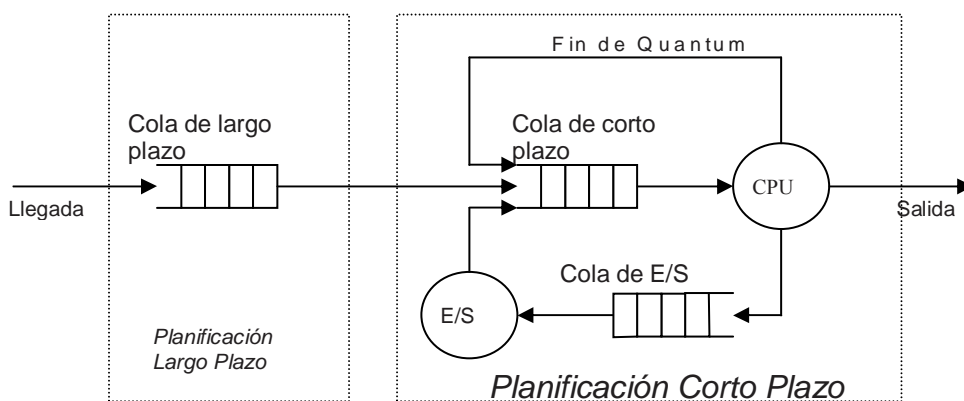
Condiciones Q=3 ciclos para el Round Robin DEL S.O. COM MAXIMA PRIORIDAD y q = 2 ciclos biblioteca de Threads

- 1) T=0 el único procesos en el sistema es **P1** → pasa a ejecución.
- 2) T=1 **P1** está en ejecución , Ready P2
- 3) T=2 Llega el proceso 3 con sus 2 hilos, **P1** está en ejecución . Ready P2 y P3
- 4) T=3 **P2** está en ejecución, Ready P3, Bloqueados I/O P1
- 5) T=6 **P3** está en ejecución, HILO A (HA) ejecuta 1 ciclo y pide I/O, Ready P2, Bloqueados I/O P1
- 6) T=7 **P2** está en ejecución, Ready P2, Bloqueados I/O P1 y P3:H1
- 7) T=8 **RCA** pasa a ejecución durante 3 ciclos, Ready P2 y Bloqueados I/O P1, P3:HA y P2
- 8) T=9 **RCA** en ejecución , Ready P1, Bloqueados I/O P3:H3 y P2
- 9) T=11 **P1** pasa a ejecución, P3 vuelve de I/O y RCA termina su quantum, como I/O tiene prioridad entonces queda : Ready P3 y RCA, Bloqueados I/O P2
- 10) T=12 **P3:HA** pasa a ejecutar 1 ciclo y finaliza. Ready RCA, Bloqueados I/O: P2 y P1
- 11) T=13 **P3:HB** durante 2 quantums, Ready RCA y P2; Bloqueados I/O P1.
- 12) T=15 **RCA** ejecuta 1 ciclo y finaliza, Ready: P2, P1 y P3:HB y no hay bloqueados por I/O
- 13)T=16 **P2** ejecuta, Ready: P1 y P3:HB y no hay bloqueados por I/O
- 14)T=18 Ejecuta **P1** y finaliza, Ready: P3:HB y bloqueado por I/O P2.

- 15) T=19 Ejecuta **P3:HB**, procesos en Ready RCA y en I/O P2 y P3:HB
 16) T=20 Ejecuta **RCA**, Ready no hay procesos y bloqueados por I/O: P2 y P3:HB
 17) T=21 Ejecuta RCA, Ready P2; Bloqueados I/O: P3:HB
 18) T=23 Ejecuta P2 y finaliza, Ready P3:HB, RCA y no hay bloqueados por I/O
 19) T=24 Ejecuta P3:HB, Ready RCA y no hay bloqueados por I/O
 20) T=25 Ejecuta RCA y finaliza y no hay en Ready y bloqueados por I/O

Ejercicio RM3: 10 FINAL 06/12/2003

Un sistema computacional de un solo procesador planifica el procesamiento de trabajos según el siguiente modelo:



La planificación de **largo plazo** se encarga de mantener el grado de multiprogramación en tres procesos, usando una política **SJF** (Shortest Job First). En el **corto plazo** el procesador es asignado usando una política de **Round-Robin** con un quantum de 2 unidades de tiempo. Considere los siguientes datos:

Proceso	Tiempo de llegada	CPU	E/S	CPU	Tiempo estimado de proceso (SJF)
P ₁	0	5	3	3	12
P ₂	0	2	4	3	10
P ₃	2	3	4	2	8
P ₄	4	6	2	2	11
P ₅	10	2	3	2	7
P ₆	15	3	4	4	12

Para la resolución del ejercicio, si existe coincidencia de tiempos en los eventos de entrada a la cola de corto plazo, ordénelos arbitrariamente en el siguiente orden: 1º fin de E/S, 2º Cola de Largo Plazo y 3º fin de quantum.

Suponiendo que el overhead para el cambio de contexto es despreciable y que existe un único dispositivo de E/S (el cual planifica FIFO) se pide la traza de ejecución de los procesos mediante un Diagrama de Gantt.

SOLUCIÓN

CPU	P2	P2	P1	P1	P3	P3	P1	P1	P2	P2	P3	P1	P2	P5	P5	P3	P3	P4	P4	P1	P1	P4	P4	P5	P5
E/S			P2	P2	P2	P2						P3	P3	P3	P3	P1	P1	P1	P5	P5	P5				
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

CPU	P1	P4	P4	P6	P6	P4	P4	P6					P6	P6	P6	P6
E/S				P4	P4				P6	P6	P6	P6				
	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41

Proceso	Instante en que ingresa	Instante en que finaliza
P1	0	26
P2	0 (encola primero por SJF)	13
P3	2	17
P4	17	32
P5	13	25
P6	25	41

Instante	Job Queue	Ready Queue	CPU	I/O Process	End Process
T0		P1	P2		
T2		P3	P1	P2	
T4	P4	P1-P5	P3	P2	
T6	P4	P3-P2-P1	P5		
T8	P4	P3-P2	P1	P5	
T10	P6-P4	P1-P3	P2	P5	
T11	P6-P4	P5-P1-P3	P2		
T12	P6-P4	P2-P5-P1	P3		
T13	P6-P4	P2-P5	P1	P3	
T14	P6-P4	P2	P5	P1-P3	
T16	P6	P4	P2	P1-P3	P5
T17		P6-P3	P4	P1	P2
T19		P4-P6	P3	P1	
T20		P1-P4-P6	P3		
T21		P1-P4	P6		P3
T23		P6-P1	P4		
T25		P6-P4	P1		
T27		P1-P4	P6		
T28		P1	P4	P6	
T30			P1	P4-P6	
T31				P4-P6	P1
T32			P6	P4	
T34		P6	P4		
T36			P6		P4
T38					P6



Ejercicio RM3: 11

En un mismo instante llegan a un centro de calculo cinco trabajos por lotes, los trabajos de A al E. Teniendo estimados como tiempo de ejecución 15, 9, 3, 6 y 12 minutos respectivamente. Sus prioridades son 6, 3, 7, 9 y 4 respectivamente, donde un valor menor corresponde a una prioridad mas alta. Determine el tiempo de retorno de cada proceso y el tiempo medio de retorno para cada uno de los siguientes algoritmos de planificación:

- Planificación por prioridades
- FCFS
- SJF

Solución

PRIORIDAD			FCFS		
ORDEN	TRABAJO	TIEMPO	TRABAJO	TIEMPO	
7	B	9	A	15	
6	E	9 + 12 = 21	B	15 + 9 = 24	
4	A	21 + 15 = 36	C	24 + 3 = 27	
3	C	36 + 3 = 39	D	27 + 6 = 33	
1	D	39 + 6 = 45	E	33 + 12 = 45	

SJF		
ORDEN	TRABAJO	TIEMPO
3	C	3
6	D	6 + 6 = 12
9	B	12 + 9 = 21
12	E	21 + 12 = 33
15	A	33 + 15 = 48

El tiempo de retorno de todos los procesos es 45 minutos.

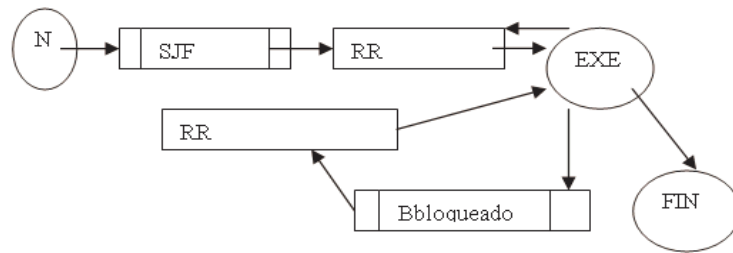
El tiempo medio de retorno de cada planificación es:

- PRIORIDAD $(9 + 21 + 36 + 39 + 45) / 5 = 30$ minutos
- FCFS $(15 + 24 + 27 + 33 + 45) / 5 = 28,8$ minutos
- SJF $(3 + 12 + 21 + 33 + 48) / 5 = 23$ minutos



Ejercicio RM3: 12

FINAL UTN del 06/12/2003

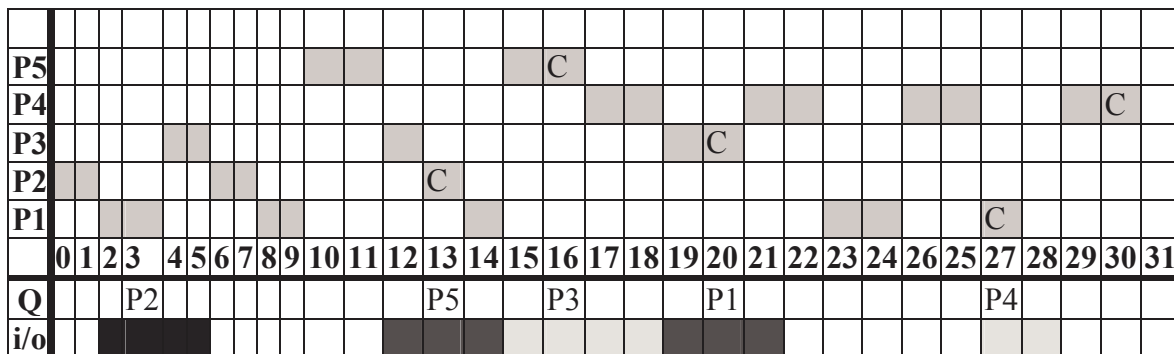


Considere los siguientes datos:

Proceso	Tiempo de Llegada	CPU	E/S	CPU	Tiempo estimado de proceso (SJF)
P1	0	5	3	3	12
P2	0	2	4	3	10
P3	2	3	4	2	8
P4	4	6	2	2	11
P5	4	2	3	2	7

Suponiendo que el overhead para el cambio de contexto es despreciable y que existe un único dispositivo de E/S (el cual planifica FIFO) se pide la traza de ejecución de los procesos mediante un Diagrama de Gantt.

RESPUESTA:



Tiempo	Cola SJF	Cola RR	Cola RR FIN E/S	CPU	BLOQ E/S
0		P1		P2	
2		P3		P1	P2
4	P4	P5,P1		P3	P2
6	P4	P3,P5,P1	P2	P2	
8	P4	P2,P3,P5		P1	
10	P4	P1,P2,P3		P5	
12	P4	P1,P2		P3	P5
13	P4	P1		P2	P3,P5
14		P4		P1	P3,P5
15		P4	P5	P5	P1,P3
17				P4	P1,P3
19		P4	P3	P3	P1
21				P4	P1
22			P1	P4	
23		P4		P1	
25		P1		P4	
27				P1	P4
29			P4	P4	
31					

Se deben ejecutar dos procesos. Uno implementa hilos y el otro procesos. El S.O. maneja la cola de listos con un algoritmo Round Robin con quantums iguales a 3, aunque para los hilos el quantum es igual a 1. Estos procesos poseen semáforos $SA = 0$, $SB = 0$, que se encuentran inicializados en 0. Es importante aclarar que es necesario un ReadFile para leer todo el archivo, que cuando finaliza un hilo automáticamente se une y que el orden de llegada de los procesos es PA en el instante 0 y PB en el instante 1. Se pide la traza de ejecución de los procesos mediante un Diagrama de Gantt.

Función	CPU	I/O	Detalle
Fork	2	0	atómico
Crear_hilo	1	0	
Up / Down (V() / P())	2	0	atómico
Readfile	1	2	
Exit	1	0	
Finalizar_hilo	2	0	atómico
Unir_hilo	2	0	atómico

Proceso A	Proceso B
<pre> main() { Crear_hilo1_biblioteca(Hilo1); Crear_hilo1_biblioteca(Hilo2); Unir_hilo(Hilo1); Unir_hilo(Hilo2); exit (0); } // main Crear_hilo1_biblioteca(void *nombre_hilo) { ReadFile("River.txt", lsize); Finalizar_hilo(NULL); } Crear_hilo2_biblioteca (void *nombre_hilo) { ReadFile("Boca.txt", lsize); Finalizar_hilo(NULL); } </pre>	<pre> main() { if ((iPid1 = fork()) == 0) { ReadFile("River.txt", lsize); up(PA); exit(0); } if ((iPid2 = fork()) == 0) { ReadFile("Boca.txt", lsize); up(PB); exit(0); } if (padre) { down(PA); down(PB); exit(0); } } // main </pre>

(*) En este ejercicio no solo se evalúan las trazas de ejecución sino que también es necesario conocer la teoría de hilos y procesos para poder desarrollarlo correctamente.

RESOLUCIÓN

[illegible]

Ejercicio RM3: 16

EJERCICIOS SIN RESOLVER DE PLANIFICACIÓN

Ejercicio M3: 1

En un determinado momento de un sistema se cuenta con la siguiente lista de procesos, y los tiempos de su próxima ráfaga de CPU, en la cola de listos, Se pide que ordene la cola de listos según el método de planificación a usar, simule la ejecución de los procesos según dicho método de administración, luego debe calcular los tiempos de regreso, espera, e índice de servicio, de cada proceso, y los tiempos medios de servicio, de espera, y la eficiencia.

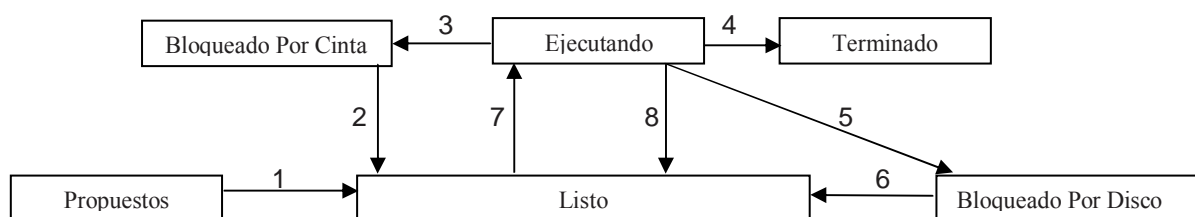
Proceso	Tamaño	T. Llegada	Prioridad
A	20	1	4
B	15	2	6
C	25	0	7
D	20	3	3
E	5	5	6
F	15	4	7

Métodos de planificación:

- F.C.F.S.
- S.P.F.
- Prioridades (0 es la menor).
- R.R. (Quantum 5).

Ejercicio M3: 2

Dado el siguiente diagrama de Transición de Procesos:



Considerar: que la estrategia de ejecución es RR y llega primero el proceso A.

Además se supone:

- Las rutinas que produce la Transición 1 y 4, demoran 10 mseg..
- El resto de las rutinas demoran 5 mseg..
- El método de selección de la Cola de Listos es RR, asignándole a cada proceso 20 mseg. de quantum.
- El sistema tiene dos Canales (Disco y Cinta), que son administrados por semáforos.
- Una operación de Entrada / Salida sobre Cinta tarda 50 mseg. y sobre Disco 40 mseg.

Los procesos realizan los siguientes trabajos:

- Proceso A:** Ejecuta 30, Realiza una operación de I/O sobre cinta, ejecuta 10, realiza una E/S sobre disco, ejecuta 10 mseg y termina.
- Proceso B:** Ejecuta 10, realiza una I/O sobre cinta, ejecuta 10 mseg, realiza E/S sobre disco, ejecuta 30 mseg y termina.

Se pide:

Realizar un gráfico con lo que realiza cada proceso. Indicando sobre el diagrama temporal en que instante se producen las interrupciones (y su clasificación completa), que rutina del S.O. se activa en cada caso (1 a 8), y en caso de haber conflicto de interrupciones explique como se evalúo cual atender primero. Analice el gráfico resultante e indique el tiempo en el que se estuvo ejecutando el proceso nulo.

Ejercicio M3: 3

En un determinado sistema operativo que planifica el uso del procesador a través del método R.R con Q=1, y el orden dentro de la cola es dado a través de prioridades variables, se tienen cuatro procesos distintos con las siguientes prioridades asignadas estáticamente (denominada p_nice, que es la clase de prioridad determinada

por el usuario, teniendo valores posibles de 0 a 39, estando reservados los primeros 20 para el superusuario, y la default es 20), $A=20$, $B=30$, $C=20$, $D=0$ (proceso del superusuario). Las prioridades finales (denominadas p_pri), se calculan a través de la siguiente formula (se realiza el cálculo cada un segundo): $p_pri = p_cpu/2 + p_user + (p_nice - nzero)$. Los valores de p_user , y $nzero$ son estáticos, y son por el usuario (el dueño de los procesos), la diferencia de los valores p_user y $nzero$ dan un rango entre 0 y 119, los primeros 40 están reservados para el sistema operativo, el resto son para los usuarios comunes. Los valores por default son $p_user = 60$, y $nzero = 20$. Cuando un proceso se encuentra ejecutando el sistema operativo le asigna un Clock Tick cada un sesentavos de segundo en el valor de p_cpu (éste valor no puede ser superior a 80). Se pide que realice el análisis de que proceso debe ingresar al procesador, analizando por lo menos los primeros 4 segundos.

Ejercicio M3: 4

Un sistema distribuido cuenta con dos maquinas Pentium 200 MMX con 65 Mb de memoria.

La maquina A usa algoritmo SRT (Shortest Remaining time) para dispatchar y solo posee un disco rígido de 1.8 Gb y la maquina B usa algoritmo FIFO y tiene una impresora láser blanco y negro. Solo se migran los procesos por la falta de recursos, y esto consume un ciclo de reloj. Dibuje cada cola de Listos, ejecutados y bloqueado para cada ciclo. Los relojes de ambas maquinas están perfectamente sincronizados.

Proceso	Maquina	Ciclo inicio	Recurso	Ciclo duración
1	A	0	CPU	3
			Disco	1
			CPU	2
2	B	0	CPU	1
			Disco	2
			CPU	4
3	A	1	CPU	1
			Impresora	2
			CPU	1

Ejercicio M3:5

Dada una serie de trabajos a realizarse utilizando los algoritmos Non Preemptive FCFS y SJF se pide :

- Calcular el tiempo medio de espera de los procesos para según ambos algoritmos.
- Calcular desde un tiempo $x=0$ en el que todos los procesos están en la cola de listos para ejecución el tiempo de finalización de todos los procesos
- Suponiendo que estos algoritmos pudieran intercambiar procesos que se encuentran listos para ejecutar cuando el que está en ejecución se bloquea por I/O.

¿ Puede darse el caso de que la misma serie de trabajos se realice en menos tiempo para uno de estos algoritmos en comparación con el otro?. De ser así de un ejemplo y explique cual es el motivo.

Procesos dados por orden de llegada

Nombre del Proceso	Tiempo de ejecución
A	40 ms
B	5 ms
C	25 ms
D	30 ms
E	12 ms

Ejercicio M3: 6

Sea la siguiente carga de trabajo:

Realizar el diagrama de Gantt y evolución del estado de la cola de espera para los algoritmos de planificación por prioridades y SJF en los casos apropiativo (non preemptive) y no apropiativo preemptive), y RR (Quantum o Time Slice = 3).

Trabajo	Instante llegada	Tiempo de CPU	Prioridad
A	0	20	4
B	3	4	5
C	4	2	4
D	6	3	5
E	13	6	2

Ejercicio M3:7

Teniendo en cuenta el cuadro del punto (2).

Imagine un sistema compuesto por dos procesadores, con una cola de listos (ready queue) compartida y cola de espera individual.

Realizar el diagrama de Gantt y evolución del estado de la cola de espera para los algoritmos de planificación:

- Ready: FCFS, Procesador 1 (P1): RR (Q = 3), Procesador 2 (P2): Prioridades (no apropiativas).
- Ready: RR (Q = 3), P1: SJF, P2: Prioridades (no apropiativas).
- Ready: Prioridades, P1: Prioridades (no apropiativas), P2: SJF.
- Ready: Colas Multinivel (niveles por prioridades – si no se llega a ejecutar ningún proceso de una cola en 4 unidades de tiempo se sube a todas las colas menores de nivel), P1: SJF, P2: RR (Q = 2).

→ Calcular tiempo de respuesta y tiempo de retorno de cada trabajo, tiempo medio y tiempo promedio de respuesta y retorno.

Ejercicio M3: 8

En cada activación del S.O. se produce una de las siguientes alternativas:

- No cambia el estado de ningún proceso.
- Cambia el estado de algún proceso.
- Cambia el estado de un proceso pero no hay cambio de contexto.
- Hay un cambio de contexto voluntario.
- Hay un cambio de contexto involuntario.

Se pide que analice cuáles de 5 alternativas pueden ocurrir para cada uno de los tipos de activación del S.O. que se detallan a continuación, planteando en cada caso una situación que sirva como ejemplo. Supóngase que en el sistema se usa un algoritmo de planificación expulsivo basado en prioridades.

- Interrupción de reloj para un proceso.
- Interrupción de disco.
- Llamadas al sistema.
- Error de proceso.
- Creación de un proceso hijo.
- Creación de un proceso hilo o thread.
- Interrupción de reloj para un hilo.

Ejercicio M3: 9

Un sistema distribuido cuenta con:

Máquina A:

- Pentium 200 MMX con 65 Mb de memoria.
- Algoritmo de planificación SRT (Shortest Remaining time) para dispatcher.
- Disco rígido de 1.8 Gb

Máquina B:

- Pentium 200 MMX con 65 Mb de memoria.
- Algoritmo de planificación FCFS (First Come First Served) para dispatcher.
- Impresora láser blanco y negro.

Solo se migran los procesos por la falta de recursos, y esto consume un ciclo de reloj.

Dibuje cada cola de Listos, ejecutados y bloqueado para cada ciclo.

Los relojes de ambas maquinas están perfectamente sincronizados.

Proceso	Maquina	Ciclo inicio	Recurso	Ciclo duración
1	A	0	CPU	3
			Disco	1
			CPU	2
2	B	0	CPU	1
			Disco	2
			CPU	4
3	A	1	CPU	1
			Impresora	2
			CPU	1
4	A	2	CPU	2
			Disco	2

5	B	2	Impresora	2
			CPU	3
			CPU	1
			Disco	2
			Impresora	3
			CPU	1

Ejercicio M3: 10

Cuatro procesos de A-D tienen tiempo de ejecución aproximados de 6, 4, 8, 2 segundos respectivamente, con prioridades (determinadas en forma externa) de 2, 1, 3 y 4 respectivamente siendo 4 la prioridad máxima.

Determinar el tiempo promedio de retorno a cada proceso para cada uno de los siguientes algoritmos de planificación. (Se ignoran los tiempos excesivos de la alternancia entre procesos).

- Raund Robin (Quantum 2 segundos, sistema multiprogramado, cada tarea obtiene porción justa de CPU.)
- Planificación por Prioridad
- First come first served
- Shortest job first

Ejercicio M3: 11

En todos los casos de algoritmos de planificación se pide : obtener la grafica Gantt y el tiempo medio de finalización.

- SJF, por prioridades. (Preemptive).
- SJF, FIFO (No preemptive)

Trabajo	Inst. de llegada	Tiempo de CPU	Prioridad
A	0	4,6	1
B	0,2	5,2	3
C	1,8	1,2	2
D	3	2,8	4
E	3,4	4	3
F	3,8	2,8	0

Nota : este S.O. le asigna la mayor prioridad al numero mas alto.

Ejercicio M3: 12

Trabajo	Inst. de llegada	Tiempo de CPU
I	2	5
LL	4	2
X	6	6
T	5	5
H	0	9
R	4	1

- RR $q=3$.
- Los trabajos llegan a la cola 1, cuando son ejecutados pasan a la cola 2 con diferente quantum. La primera cola de planificación tiene mayor prioridad que la cola 2. A la cola 1 llegan los trabajos nuevos .En la cola 2 llegan los trabajos de la cola 2 y también llegan de la cola 1.
El S.O. tiene instalado a BILL WIN LAUCHA y hace que todos los trabajos que se ejecutaron en la cola 2 pasen a la cola 3 pero solo si sus tiempos restantes de CPU son impares. Esta cola tiene mayor prioridad que la cola 1y utiliza el algoritmo de planificación LIFO.
- ¿Cuánto duró todo el procesamiento?

Ejercicio M3: 13

Obtener la grafica Gantt del siguiente trabajo teniendo en cuenta que hasta $t=20$ el S.O. Utiliza el algoritmo de planificación FCFS y luego el S.O. utiliza el algoritmo de planificación S.P.F. (Non Preemptive). El S.O. asigna la mayor prioridad al menor numero.

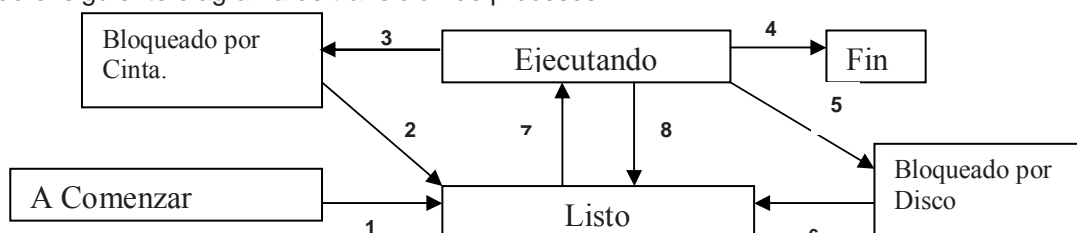
Trabajo	Inst. de llegada	Tiempo de CPU	Prioridad
A	0	4	0
B	2	5,5	1
C	5	6	3

Procesos dados por orden de llegada:

Proceso	Tiempo de ejecución
A	40ms
B	5ms
C	25ms
D	30ms
E	12ms

Ejercicio M3: 17

Dado el siguiente diagrama de transición de procesos:



Supongamos que el sistema ejecuta dos procesos de las siguientes características:

Proceso A: Ejecuta 30ms, efectúa una E/S sobre cinta, ejecuta 10ms y termina.

Proceso B: Ejecuta 10ms, efectúa una E/S sobre cinta, ejecuta 10ms, efectúa una E/S sobre disco, ejecuta 10ms y termina.

Considere:

1. La estrategia de ejecución es FCFS y llega primero el proceso A.
2. Cada rutina que produce la transición (1 a 8) se ejecuta 10 ms ante cualquier evento.
3. El método de selección de la cola de listos es FIFO, asignándole a cada proceso 20 ms.
4. El sistema tiene dos canales (disco y cinta) administrado por semáforos.
5. Una operación de E/S sobre cinta tarda 50 ms y sobre disco 40 ms.

Se pide:

- a) Realizar un diagrama temporal de ejecución.
- b) Indicar sobre el diagrama temporal en que instante se producen las interrupciones y cual es la rutina del S.O. (1 a 8) que es activada en cada caso.
- c) Cuanto tiempo insume la ejecución de ambos procesos:
 - Tiempo de servicio.
 - Tiempo de espera.
 - Índice de servicio.
- d) Cuanto tiempo ejecuta el S.O

Ejercicio M3: 18

Consideremos los procesos en la siguiente tabla:

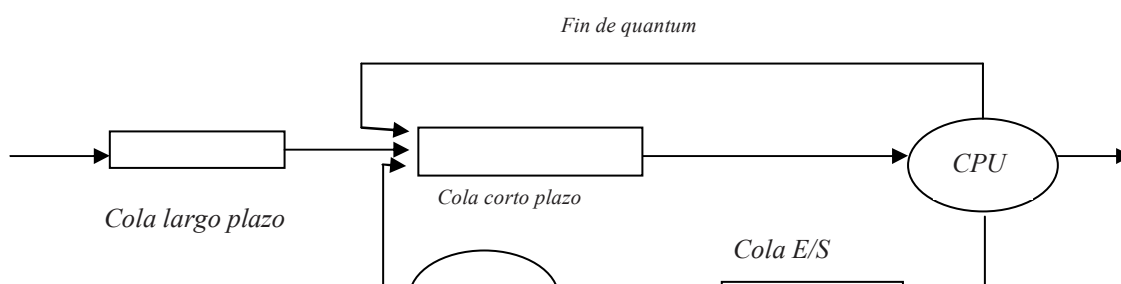
Proceso	Instante de llegada	CPU	Bloqueo	CPU	Bloqueo	CPU	Bloqueo	CPU
A	0	2	1	2	2	1	1	1
B	1	2	1	2	2	1	-	-
C	2	1	1	1	2	1	1	1
D	4	3	2	1	2	1	-	-

Dibujar el diagrama de Gantt para los Algoritmos de planificación de CPU:

- a) FIFO
- b) RR $q=1$
- c) Prioridades suponiendo que las prioridades son 3 para A y B 2 para C y 1 para D, siendo la de menor valor la mas prioritaria
- d) Considere que el bloqueo es por una E/S sobre Impresora.

Ejercicio M3: 19

Un sistema computacional de un solo procesador planifica el procesamiento de trabajos según el siguiente modelo:



La planificación de largo plazo se encarga de mantener el grado de multiprogramación en tres procesos, usando una política SJF (Shortest Job First). En el corto plazo el procesador es asignado usando una política Round Robin con un quantum de 2 unidades de tiempo

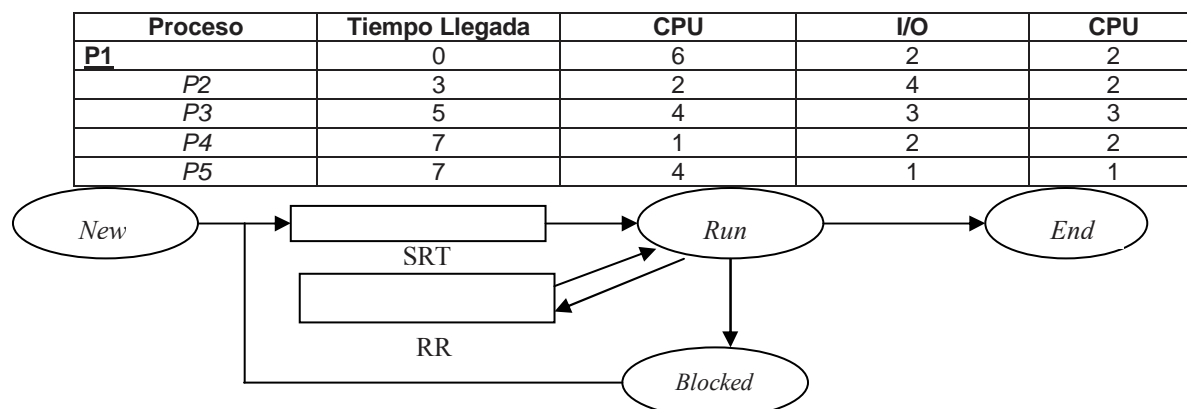
Proceso	T. Llegada	CPU	E/S	CPU	Tiempo Est.
P1	0	5	3	3	12
P2	0	2	4	3	10
P3	2	3	4	2	8
P4	4	6	2	2	11
P5	10	2	3	2	7
P6	15	3	4	4	12

Para la resolución del ejercicio, si existe coincidencia de tiempos en los eventos de entrada a la cola de corto plazo, ordénelos arbitrariamente en el siguiente orden: 1) Fin de E/S 2) Cola de largo plazo 3) fin de quantum.

Suponiendo que el overhead para el cambio de contexto es despreciable y que existe un único dispositivo de E/S (planificado por FIFO) se pide la traza de ejecución de los procesos.

Ejercicio M3: 20

Suponga un planificador que utiliza dos colas con distintos algoritmos de planificación: la primera, Shortest Remaining Time First (SRT), y la segunda, Round Robin (RR) con un quantum de 3. Sabiendo que la primera cola tiene mayor prioridad y que existe un único dispositivo de E/S, se pide realizar la traza de ejecución de los procesos con el siguiente esquema de estados:



Ejercicio M3: 21

Suponga un sistema que posee un procesador, el cual posee una planificación SJF (Shortest Job First), y dos dispositivos que planifican según FIFO; uno es una impresora y otro un disco. Suponiendo que al SO le lleva un ciclo de CPU realizar el cambio de proceso, se pide realizar la traza de los mismo con los siguientes datos:

Proceso	T. Llegada	CPU	E/S	CPU	E/S	CPU
P1	0	5	Imp: 2	2	Disco: 5	3
P2	0	4	Disco: 3	3	Disco: 2	1
P3	3	2	Disco: 5	4	Imp: 7	2
P4	5	7	Imp: 1	1	Imp: 2	3
P5	9	1	Disco: 3	5	Imp: 1	2

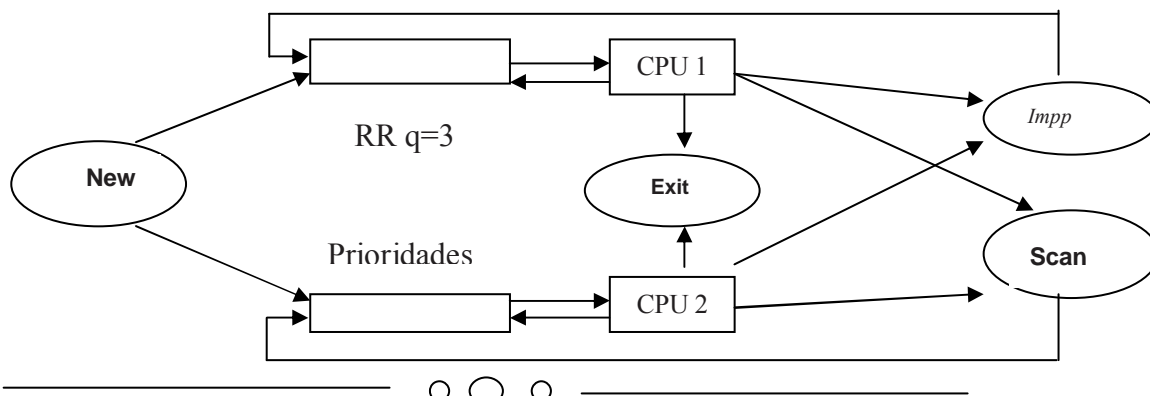
Ejercicio M3: 22

Un sistema de multiprocesamiento posee dos CPU, la primera de ellas planifica según un algoritmo no apropiativo (Round Robin con quantum=3), mientras que la segunda CPU planifica según un algoritmo basado en prioridades preemptive; a menor número mayor prioridad. Además se tienen dos dispositivos de E/S que planifican FIFO, una impresora y un escáner. Según el siguiente esquema y los datos de los procesos:

Proceso	T. llegada	CPU	E/S	CPU
P1 Pri: 2	0 (CPU1)	10	Scanner: 5	4
P2 Pri: 3	0 (CPU2)	5	Imp: 1	3
P3 Pri: 5	1 (CPU2)	3	Scanner: 6	3
P4 Pri: 1	3 (CPU2)	1	Imp: 2	8

Se pide

- Diagrama de Gantt de la ejecución
- Indicar Tiempo de Finalización de cada proceso
- Turnaround time de cada proceso



Ejercicio M3: 21

Un sistema operativo planifica según un algoritmo de RR=4. Además se tiene un disco que encola los pedidos según un algoritmo FIFO. Se pide la traza de ejecución de los procesos teniendo en cuenta el siguiente cuadro:

(KLT = Hilo a nivel Kernel - ULT = Hilo a nivel Usuario)

Proceso		Llegada	CPU	I/O	CPU	I/O	CPU
P1	KLT	0	3	3	5	2	1
	ULT		5	2	7		
P2	KLT	2	6	1	2	3	1
	KLT		5	4	1		
P3	ULT	4	2	2	5		
	ULT		7	5	3	1	3

Ejercicio M3: 22

Suponga un sistema distribuido, en el que se cuenta con dos máquinas:

Máquina A

Algoritmo SRT

Disco rígido 1.8 Gb

Máquina B

Algoritmo FCFS

Impresora láser Blanco y Negro

Solo se migran los procesos por falta de recursos, y eso consume un ciclo reloj. Dibuje la traza de los procesos, teniendo en cuenta que ambas máquinas están perfectamente sincronizadas.

Proceso	Máquina	Ciclo Inicio	Recurso	Ciclos duración
P1	A	0	CPU	3
			Disco	1
			CPU	2
P2	B	0	CPU	1
			Disco	2
			CPU	4
P3	A	1	CPU	1
			Impresora	2
			CPU	1
P4	A	2	CPU	2
			Disco	2
			Impresora	2
P5	B	2	CPU	3
			CPU	1
			DISCO	2
			Impresora	3
			CPU	1

Ejercicio M3: 23

Considere los procesos cuyo comportamiento se recoge en la siguiente tabla:

Proceso	Llegada	CPU	I/O	CPU	I/O	CPU	I/O	CPU
A	0	1	2	1	2	1	4	2
B	1	1	1	5	1	7	-	-

C	2	2	1	2	3	2	-	-
D	4	4	3	3	-	-	-	-

Sabiendo que hay una sola cola de bloqueados, se pide realizar la traza de ejecución según:

- FIFO
- RR (q=1)
- SRT
- Prioridades Apropiativo (A=3, B=2, C=D=1)
- Prioridades NO-apropiativo (las mismas que para d.)

** Considerar que a menor número, mayor prioridad

○ ○ ○

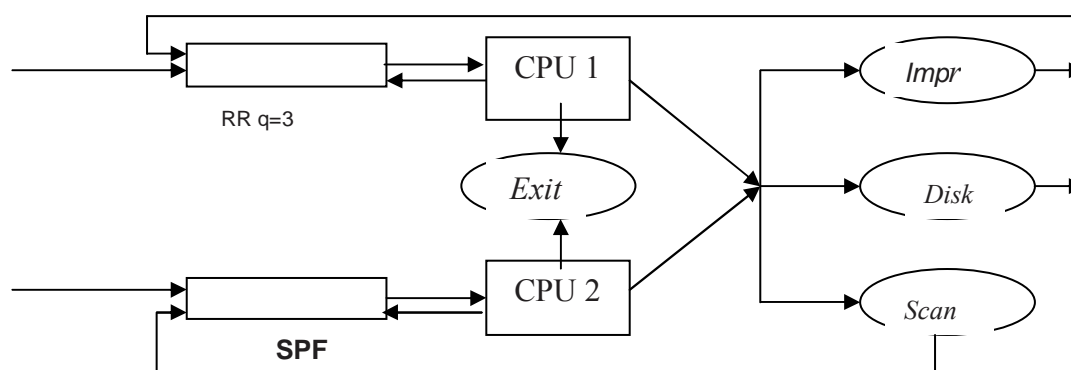
Ejercicio M3: 24

Suponga un sistema con el siguiente diagrama, en el cual existen dos procesadores, cada uno de los cuales tiene su propia cola de listos, la primera planifica según el algoritmo no-apropiativo Round Robin (q=3) y la segunda mediante un algoritmo apropiativo; el Shortest Process Next.

Además existen 3 dispositivos de E/S: una impresora, un disco y un scanner; cada uno con su propia cola planificando mediante FIFO.

Teniendo en cuenta que para ambos procesadores se consume un ciclo de CPU cuando se debe realizar un cambio de proceso; se pide realizar el diagrama de Gantt.

Proceso	T. Llegada	CPU	E/S	CPU	E/S	CPU
P1	0 (Proc: 1)	4	Imp: 3	3	Sca: 2	3
P2	0 (Proc: 2)	8	Sca: 1	6	Disco: 5	1
P3	1 (Proc: 1)	2	Sca: 2	1	Sca: 1	4
P4	2 (Proc: 1)	7	Disco: 3	2	Disco: 3	5
P5	3 (Proc: 2)	3	Disco: 5	5	Imp: 7	2
P6	5 (Proc: 2)	1	Imp: 2	4	Imp: 3	2



(En caso de superposición de tiempos, dar prioridad a los procesos que vuelven de los dispositivos)

○ ○ ○

Ejercicio M3: 25

Suponga un sistema operativo que realiza una planificación no-apropiativa con un Round Robin cuyo quantum es igual a 2 (q=2). Se tiene además una impresora que ordena sus pedidos según un algoritmo FIFO. Sabiendo que se consume un ciclo de procesador cada vez que el sistema operativo debe asignar un nuevo proceso a la CPU, se pide la traza de los mismos.

Proceso		Llegada	CPU	E/S	CPU	E/S	CPU
P1	ULT	0	1	6	2	5	2
	ULT		1	2	3	-	-
	KLT		3	2	1	-	-
P2	ULT	5	3	3	3	1	3
	ULT		2	1	4	-	-

○ ○ ○

Ejercicio M3: 26

Se cuenta con dos máquinas que poseen las siguientes características:

	Máquina A	Máquina B
Planificación	SRT	SPN
Recursos	Disco rígido de 20 Gb	Impresora Scanner

Teniendo en cuenta que se consume 1 ciclo de CPU cada vez que se necesita migra un proceso (solamente por falta de recursos), se pide la siguiente traza de procesos:

Proceso	Máquina	Ciclo Inicio	Recurso	Duración
P1	A	0	CPU	7
			Disco	5
			CPU	3
			Scanner	4
			CPU	1
P2	B	1	CPU	2
			Disco	3
			CPU	5
P3	A	2	CPU	3
			Impresora	5
			CPU	2
P4	B	4	CPU	3
			Scanner	2
			CPU	1
			Impresora	3
			CPU	3

Ejercicio M3: 27

Dada la siguiente tabla de procesos:

Id. de Proceso	Tiempo de arribo a la Cola de Ready en ut.	Necesidad de procesador en ut	Instante de solicitud de E/S en ut.
P1	0	10	3, 4, 8
P2	3	5	4
P3	4	8	2
P4	6	18	5, 10, 15
P5	10	8	3, 7

Calcular el Tiempo de espera promedio para un S.O. cuyo algoritmo de Scheduling de corto plazo es Round Robin con un Quantum de 3 ut.(unidades de Tiempo), sabiendo que la atención de una E/S dura 2 ut. y hace que el proceso libere el procesador y se encole nuevamente en la cola de Listos al cabo de ser satisfecha

Ejercicio M3: 28

Se tiene la siguiente tabla de procesos a ejecutar:

Proceso	T. Llegada	CPU	IO	CPU	IO	CPU
P1	0	4	3	6	2	8
P2	5	3	2	7	4	3
P3	8	2	1	4	2	4

Sabiendo que existe sólo una instancia del dispositivo de Entrada / Salida y que el algoritmo de planificación es del FIFO (First in – First Out), se pide que indique la traza de ejecución de los procesos, mediante la realización de un diagrama de GANTT.

Ejercicio M3: 29

Un Sistema Operativo posee un STS (Short Term Scheduler) Round Robin con quantum de 5 unidades. Los procesos que van a ejecutar en este Sistema se encuentran representados en la siguiente tabla:

Proceso	T. Llegada	CPU	IMPRESORA	CPU	DISCO	CPU
P1	0	4	3	6	2	8
P2	5	3	2	7	4	3
P3	8	2	1	4	2	4

Si se considera que los únicos dispositivos de Entrada / Salida que posee este Sistema son el Disco y la Impresora, los cuáles planifican FIFO, se pide, mediante la confección en forma clara y detallada de un **diagrama de GANTT**, los tiempos de finalización de los procesos y sus respectivos Turn Around Time.

2. b. Realice el ejercicio anterior considerando el algoritmo de planificación VRR (Virtual Round Robin) con quantum de 3 unidades.

OPCIONAL: Si se aumentara el grado de multiprogramación en este Sistema ¿Cree usted que el porcentaje de utilización de la CPU también aumentaría? **Justifique claramente su respuesta.**

Ejercicio M3: 30

Se tiene un Sistema el cual utiliza un Short Term Scheduler por prioridades preemptive. En un momento dado se va a ejecutar el siguiente set de procesos:

Proceso	Prioridad	T. Llegada	CPU	I/O	CPU	I/O	CPU
P1	40	0	2	IMP (4)	3	DISCO (4)	2
P2	30	0	3	PLOTTER (2)	X		
P3	20	1	5	PLOTTER (3)	6	DISCO (2)	2
P4	10	3	2	IMP (2)	1		

Considerando que se tienen tres dispositivos de entrada / salida, los cuales planifican FIFO, que el tiempo que insume el Sistema Operativo en realizar un process switch es de 1 ciclo de CPU y que a menor número mayor prioridad, se pide hallar el valor de **X** y los tiempos de finalización de todos los procesos para el valor hallado, sabiendo que el Turnaround Time del proceso 2 es menor que el Turnaround Time del proceso 3. **Justifique su respuesta.** ¿En que modifica el resultado si se cambia el algoritmo a prioridades nonpreemptive? **Justifique.**

NOTA: Para la resolución de este ejercicio tenga en cuenta que no se considera el inicio del mismo como un process switch y que ante la simultaneidad de eventos deberá elegir por FIFO al proceso a ejecutar.

Ejercicio M3: 31

Sea un Sistema con multiprocesamiento, en el que se poseen dos procesadores formando una arquitectura SMP (Simetric Multi Processing). Para planificar los procesos el Sistema Operativo utiliza un algoritmo SRT (Shortest Remaining Time), el cual es utilizado para ambos procesadores.

En el Sistema se va a ejecutar el siguiente conjunto de procesos:

Proceso	Tiempo de Llegada	CPU	I/O	CPU
P1	0	6	2	2
P2	3	2	4	2
P3	5	4	3	3
P4	7	1	2	2
P5	7	4	1	1

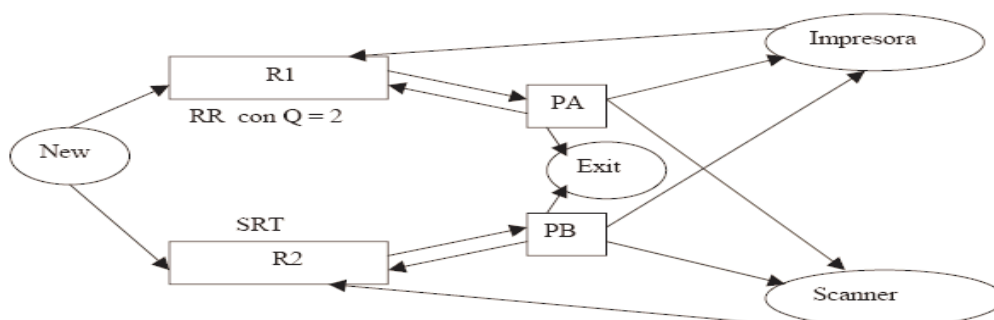
Teniendo en cuenta los datos mencionados, se pide que realice un diagrama de Gantt indicando la utilización de la CPU y los instantes en los cuales finalizan los procesos.

Ejercicio M3: 32 Final de Sistemas Operativos –13/12/2003 (Ejercicio 2)

Sea un sistema que implementa el diagrama de transiciones para la administración de los procesos, indicado en el siguiente grafico, siendo PA y PB dos procesadores.

Se sabe que todo proceso nuevo que ingresa al sistema se ubica en la respectiva cola de Ready según su prioridad y en función de las velocidades de los procesadores, según la siguiente tabla:

C o l a R 1	Procesos con Prioridad 0 , 1 , 2 ó 3
C o l a R 2	Procesos con Prioridad 4 , 5 , 6 ó 7



En un instante dado la carga del sistema es la siguiente:

Proceso	Prioridad	T. Llegada	CPU	I/O	CPU	I/O	CPU
PW	0	0	5	Scanner: 2	2	Scanner: 3	5
PX	4	2	3	Impresora :1	1	Scanner: 1	3
PY	2	1	1	Impresora :6	4	Impresora :1	2
PZ	5	4	4	Impresora :3	2	Impresora :4	1

Teniendo en cuenta que cada tipo de dispositivo de E / S tiene una única instancia, y que planifican FIFO, se pide el orden de ejecución de los procesos y en que instantes finalizan, mediante la confección, en forma clara y detallada de un diagrama de GANTT.

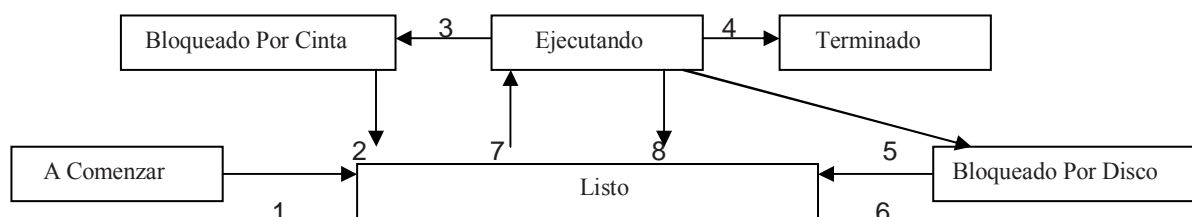
Ejercicio M3: 33

Cinco trabajos por lote, A-E, llegan a un centro de cómputos casi al mismo tiempo (considerar que es el mismo). Tienen un tiempo estimado de ejecución de 12, 4, 1, 6 y 7 minutos. Sus prioridades (determinadas en forma externa) son de: 3, 5, 2, 1 y 4, respectivamente, siendo 5 la mínima prioridad. Para cada uno de los siguientes algoritmos de planificación, realizar el diagrama de ejecución, determina el tiempo de retorno de cada proceso, ignorando el costo excesivo del context switch entre procesos.

- Round Robin (Q=3)
- Planificación por prioridad
- First Come First Served
- Shortest Job First

Ejercicio M3: 34

Dado el siguiente diagrama de Transición de Procesos:

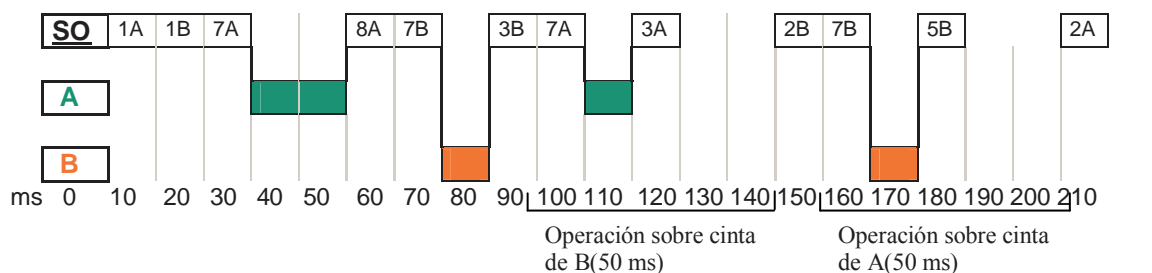


Considerar: que la estrategia de ejecución es RR y llega primero el proceso A.

Además se supone:

- Cada rutina que produce la Transición (1 a 8) se ejecuta 10 mseg ante cualquier evento.
- El método de selección de la Cola de Listos es RR, asignándole a cada proceso 20 mseg.
- El sistema tiene dos Canales (Disco y Cinta) administrados por semáforos.
- Una operación de Entrada / Salida sobre Cinta tarda 50 mseg y sobre Disco 40 mseg.

El diagrama de ejecución de los procesos es el siguiente:



Los procesos realizan las siguientes tareas:

Proceso A: Ejecuta 30, realiza una I/O sobre cinta, ejecuta 10 mseg, realiza una E/S sobre disco, ejecuta 10 mseg y termina.

Proceso B: Ejecuta 10, realiza una I/O sobre cinta, ejecuta 10 mseg, realiza E/S sobre disco, ejecuta 30 mseg y termina.

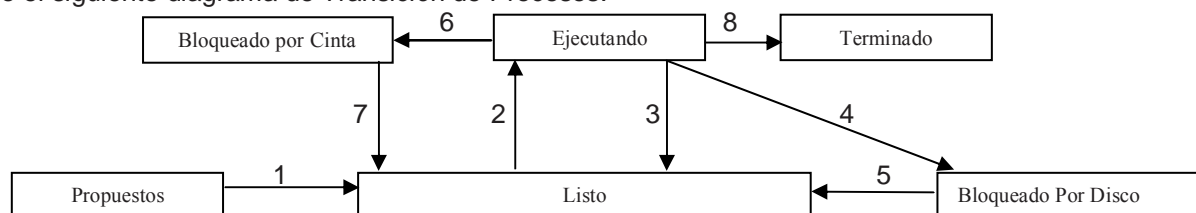
Se pide:

Continuar el gráfico con lo que realiza cada proceso. Indicar sobre el diagrama temporal en que instante se producen las interrupciones (y su clasificación completa), y en caso de haber conflicto de interrupciones explique como se evaluó cual atender primero. Analice el gráfico resultante e indique el tiempo en el que

se estuvo ejecutando el proceso nulo, marque y calcule todos los tiempos de espera para cada uno de los recursos existentes.

Ejercicio M3: 35

Dado el siguiente diagrama de Transición de Procesos:



Considerar: que la estrategia de ejecución es RR y llega primero el proceso A.

Además se supone:

- Las rutinas que producen la Transición 1 y 8, demoran 10 mseg..
- El resto de las rutinas demoran 5 mseg..
- El método de selección de la Cola de Listos es RR con prioridades variables (fija por proceso: de 0 a 29 reservadas para el S.O., de 30 a 60 para los procesos, la parte variable se calcula según los tiempos de procesamiento utilizados con anterioridad), asignándole a cada proceso 10 mseg. de quantum.
- El sistema posee 1 unidad de disco, 1 unidad de cinta y 2 canales selectores, y uno multiplexor para administrar la entrada / salida. El esquema de conexión es Canal A, Disco, Canal A Cinta, Canal B Disco.
- Una operación de Entrada / Salida sobre Cinta tarda 50 mseg. y sobre Disco 30 mseg.

Los procesos realizan los siguientes trabajos:

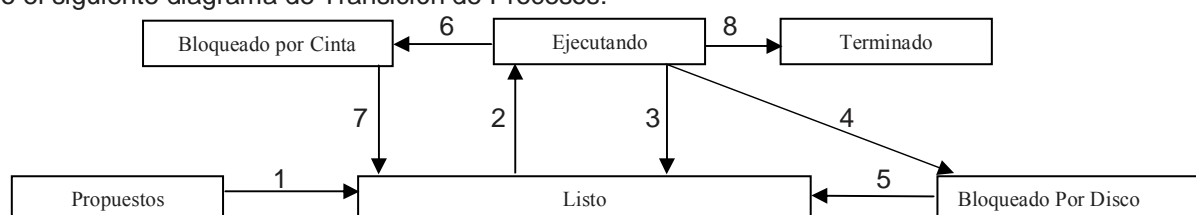
- Proceso A (prioridad 31):** Ejecuta 20 mseg., realiza una operación de I/O sobre disco, ejecuta 30 mseg. y termina.
- Proceso B (prioridad 33):** Ejecuta 10 mseg., realiza una I/O sobre disco, ejecuta 40 mseg., realiza I/O sobre disco, ejecuta 20 mseg. y termina.
- Proceso C (prioridad 30):** Ejecuta 5 mseg., realiza una I/O sobre tape, ejecuta 10 mseg., realiza I/O sobre tape, ejecuta 20 mseg. y termina.

Se pide:

Realizar un gráfico con lo que realiza cada proceso. Indicando sobre el diagrama temporal en que instante se producen las interrupciones (y su clasificación completa), que rutina del S.O. se activa en cada caso (1 a 8), y en caso de haber conflicto de interrupciones explique como se evalúo cual atender primero. Analice el gráfico resultante e indique como se podría reducir el overhead.

Ejercicio M3: 36

Dado el siguiente diagrama de Transición de Procesos:



Considerar: que la estrategia de ejecución es RR y llega primero el proceso A.

Además se supone:

- Las rutinas que producen la Transición 1 y 4, demoran 10 mseg..
- El resto de las rutinas demoran 5 mseg..
- El método de selección de la Cola de Listos es RR, asignándole a cada proceso 10 mseg. de quantum.
- El sistema posee 1 unidad de disco, 1 unidad de cinta y 2 canales selectores, y uno multiplexor para administrar la entrada / salida. El esquema de conexión es Canal A, Disco, Canal A Cinta, Canal B Disco.
- Una operación de Entrada / Salida sobre Cinta tarda 50 mseg. y sobre Disco 30 mseg.

Los procesos realizan los siguientes trabajos:

- Proceso A:** Ejecuta 20 mseg., realiza una operación de I/O sobre disco, ejecuta 30 mseg. y termina.

- **Proceso B:** Ejecuta 10 mseg., realiza una I/O sobre disco, ejecuta 40 mseg., realiza I/O sobre disco, ejecuta 20 mseg. y termina.

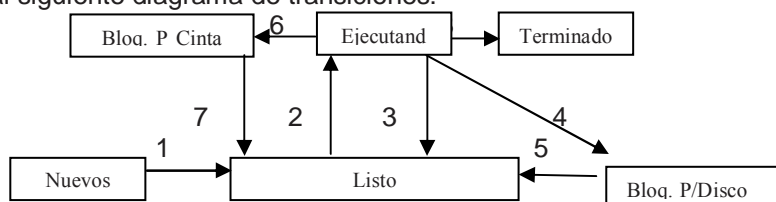
Se pide:

Realizar un gráfico con lo que realiza cada proceso. Indicando sobre el diagrama temporal en que instante se producen las interrupciones (y su clasificación completa), que rutina del S.O. se activa en cada caso (1 a 8), y en caso de haber conflicto de interrupciones explique como se evalúo cual atender primero. Analice el gráfico resultante e indique como se podría reducir el overhead.

Ejercicio M3: 37

Un sistema tiene dos canales de I/O uno selector y el otro multiplexor al que se encuentran conectados los dispositivos. Las operaciones de I/O demoran: salida de video 20, y en disco 15. El algoritmo de planificación utilizado es prioridades variables (fija + tiempo de procesamiento, que se reinicia con cada operación de I/O), con desalojo del procesador. El Quantum (Q=15). Tenga en cuenta que el algoritmo primero asigna y luego ordena la cola de listos incluyendo al proceso que llega. El planificador a largo plazo trabaja por ráfagas y planificado a través de FCFS.

En base al siguiente diagrama de transiciones:



Todas las rutinas del sistema demoran 5, excepto las rutinas 1 y 8 que demoran 10.

En un momento determinado llegan al sistema dos procesos (A y B en ese orden), con prioridad A=15, y B=3 (considere que la prioridad del sistema operativo es 0), transcurridos 35 unidades de tiempo de la llegada de los proceso A, y B, llega el proceso C, con prioridad 1, que tienen el siguiente diagrama de ejecución:

- **Proceso A:** Eje.10, I/O disco, eje.5 y termina
- **Proceso B:** Eje.15, I/O disco, eje.5, I/O monitor, eje 5, I/O disco, eje.5, y termina
- **Proceso C:** Eje.15, I/O monitor, eje.10, I/O disco, eje.10, I/O monitor, eje.5, y termina

Se pide: Realizar el diagrama temporal de procesamiento, indicando claramente los tiempos de ejecución de los procesos, de las rutinas del sistema operativo, las interrupciones que se producen para posibilitar cada cambio (con su clasificación), los conflictos (en caso de existir), y cual fue el criterio usado para su resolución, y en caso de existir el tiempo de ejecución del proceso nulo, como así los tiempos de espera.

En cada asignación del procesador debe indicar la prioridad de los procesos. Además indique:

- Tiempo de retorno de cada proceso.
- Tiempo de espera para los dispositivos de I/O.
- Tiempo de servicio para cada proceso
- Analice la performance del SO

Ejercicio M3: 38

Módulo 4: Concurrencia y Sincronización de Procesos, I.P.C. (comunicación entre procesos) y Deadlocks

SINCRONIZACION

En cuanto a la sincronización, básicamente el objetivo que se persigue es poder asegurar la exclusión mutua; es decir, que solo un proceso pueda entrar en su sección crítica (la cual hace uso de algún recurso crítico). Entiéndase por recurso crítico a todo aquel que puede ser accedido por un proceso a la vez, ya sea una variable, una estructura de datos, un dispositivo, etc.

PARTE TEORICA SEMAFOROS

Recordemos que un semáforo es un mecanismo proveído por un lenguaje de programación; estos semáforos no son ni mas ni menos que variables por las cuales los procesos se envían señales.

Los semáforos se manejan mediante tres operaciones básicas:

- Una para inicializarlo
- Una para decrementarlo
- Una para incrementarlo

Estas tres operaciones son primitivas, atómicas; es decir, no pueden ser interrumpidas. Para decrementar un semáforo podemos encontrar la primitiva: **P()** o **Down()** o **Wait()** y para incrementarlo, la primitiva **V()** o **Up()** o **Signal()**. Si bien realizan lo mismo, su construcción no es la misma, por lo tanto se deben utilizar en esa combinación; si utilizo **P()** debo utilizar **V()** (y viceversa) y si utiliza **Down()** sí o sí se debe utilizar **Up()** (y viceversa), lo mismo para **wait()** y **signals()**.

Mediante estas variables los procesos utilizan los recursos incrementando y decrementando los semáforos. Cuando un proceso decrementa el valor del semáforo y está queda en un valor negativo (<0), significa que los recursos controlados por dicho semáforos están siendo utilizados y por lo tanto el proceso debe esperar a que alguno los desocupe, para lo cual se bloquea encolándose en la cola del semáforo y queda a la espera de tener disponibilidad. Cuando un proceso incrementa el valor del semáforo. Cuando éste queda en un valor positivo o nulo (≥ 0), desencola (despierta) a alguno de los procesos que estaban a la espera del recurso, por lo cual el proceso deja de estar bloqueado y procede a continuar su ejecución.

Vale aclarar dos cosas: por un lado, cuando un proceso sale de la cola de espera y hace uso del recurso no vuelve a decrementar el semáforo sino que pasa directamente a la utilización del mismo. La disminución del valor del semáforo ya la realizó cuando intentó acceder al recurso y quedó bloqueado; en la cola sólo estaba a la espera de que se libere dicho recurso. Por otro lado, podemos ver que cuando el semáforo tiene un valor negativo, el módulo del mismo nos indica la cantidad de procesos encolados (a la espera de la utilización del recurso). Teniendo en mente esto, podemos completar la idea anterior: Si hay dos procesos encolados el valor del semáforo será igual a -2, cuando el recurso se libera el valor del semáforo pasará a ser = -1, lo cual es correcto ya que el proceso a la cabeza de la cola hará uso del recurso y ahora quedará un proceso esperando.

De esta forma, podemos ver en este caso, como las dos ideas se complementan: cuando el recurso se libera el valor del semáforo pasa a ser = -1 y el proceso que estaba esperando a la cabeza de la cola puede utilizarlo pero no vuelve a decrementar el semáforo; si lo hiciera, el semáforo pasaría a valer nuevamente = -2 lo cual no es coherente ya que hay un solo proceso esperando en la cola.

EJEMPLOS DE EJERCICIOS RESUELTOS DE SINCRONIZACIÓN:

Ejercicio RM4: 1

Una cochera subterránea inteligente cuenta con capacidad para 10 autos, y un montacargas que sube y baja los autos. Dicho montacargas solo puede ser utilizado por un auto a la vez, ya sea para subir o para bajar. Determine los semáforos necesarios con sus primitivas para sincronizar dicha cochera.

Utilice las funciones: salir_de_la_cochera() y entrar_a_la_cochera()

Solución:

Semáforos	Valor inicial
Cochera	10 (suponiendo que la cochera está vacía)
Coches	0 (suponiendo que la cochera está vacía)
Montacargas	1 (mutex)

P(cochera)	P(coches)
P(montacargas)	P(montacargas)
entrar_a_la_cochera()	salir_de_la_cochera()
V(montacargas)	V(montacargas)
V (coches)	V(cochera)

_____ ○ ○ ○ _____

Ejercicio RM4: 2

Una cochera subterránea inteligente ubicada en el segundo subsuelo cuenta con capacidad para 10 autos. Posee 2 montacargas, uno que sube y baja los autos entre la Pb y el 1° subsuelo y otro que sube y baja los autos entre el 1° subsuelo y el 2° subsuelo. Dichos montacargas solo puede ser utilizado por un auto a la vez, ya sea para subir o para bajar. En el 1° subsuelo solo hay capacidad para 2 autos.

Determine los semáforos necesarios con sus primitivas para sincronizar dicha cochera.

Utilice las funciones: bajar_al_primer_subsuelo()
 bajar_al_segundo_subsuelo()
 subir_al_primer_subsuelo()
 subir_a_la_Planta_baja()

Solución:

P(cochera)	P(coches)
P(primer_subsuelo)	P(primer_subsuelo)
P(montacargas_1)	P(montacargas_2)
bajar_al_primer_subsuelo()	subir_al_primer_subsuelo()
V(montacargas_1)	V(montacargas_2)
P(montacargas_2)	P(montacargas_1)
bajar_al_segundo_subsuelo()	subir_a_planta_baja()
V(primer_subsuelo)	V(primer_subsuelo)
V(montacargas_2)	V(montacargas_1)
V(coches)	V(cochera)

Semáforo	Valor inicial
Cochera	10 (suponiendo que la cochera está vacía)
Coches	0 (suponiendo que la cochera está vacía)
Montacargas_1	1 (mutex)
Montacargas_2	1 (mutex)
Primer_subsuelo	2

_____ ○ ○ ○ _____

Ejercicio RM4: 3

Dados los siguientes semáforos con sus respectivos valores Y=2, Z=2) y el siguiente esquema de procesos con sus respectivas primitivas.

Indicar cual será la secuencia de ejecución, ¿puede haber mas una?, ¿Cuántas veces se ejecuta la secuencia?

Solución: No llega a ejecutarse ninguna secuencia

A	B	C	D	
P(X)	P(Z)	P(Y)	P(X)	(X=2,
P(Y)	P(X)	P(Z)	P(Y)	
P(Z)	P(Y)	P(X)	P(Z)	
-	-	-	-	de
-	-	-	-	
V(X)	V(Z)	V(X)	V(Y)	
(Y)	V(X)	V(Z)	V(X)	

_____ ○ ○ ○ _____

Ejercicio RM4: 4

1) Dados los siguientes semáforos con sus respectivos valores (W=2, X=1, Y=1, Z=1) Determine el uso de las primitivas P() y V(), que deberán hacer los procesos A, B y C, para que la secuencia de ejecución sea: A B C A B C (dos veces cada proceso).

Solución:

A	B	C
P(W)	P(Y)	P(Z)
P(X)	P(Y)	P(Z)
-	-	-
-	-	-
-	-	-
V(Y)	V(Y)	V(Z)
V(Z)	V(X)	

Ejercicio RM4: 5

Dados los siguientes semáforos con sus respectivos valores (R=1, S=1, T=0). Determine el uso de las primitivas P() y V(), que deberán hacer los procesos A, B y C, para que la secuencia de ejecución sea: B A B C (solo una vez).

Solución:

A	B	C
P(S)	P(R)	P(T)
P(S)	-	P(T)
-	-	-
-	-	-
V(R)	V(S)	
	V(T)	

Ejercicio RM4: 6

Dados los siguientes semáforos (W, X, Y, Z) Determine su valor inicial y el uso de las primitivas P() y V(), que deberán hacer los procesos A, B y C, para que la secuencia de ejecución sea: A B C B A
Valores iniciales: W: 3 X: 2 Y: 0 Z: 0

Solución:

A	B	C
P(X)	P(W)	P(Z)
P(X)	P(Y)	P(Z)
-	-	P(W)
-	-	-
-	-	-
V(Y)	V(X)	V(Y)
V(Z)	V(Z)	

Ejercicio RM4: 7 Don Mateo

Don Mateo tiene en su peluquería lugar para atender a tres clientes al mismo tiempo, y un salón de espera en el que se pueden acomodar 5 clientes en un sofá. En caso de que todas las ubicaciones estén llenas, se cuenta con un lugar donde los clientes pueden esperar parados. En ningún caso la cantidad de clientes puede superar 25 en total dentro de la peluquería, ya que no se lo habilita la municipalidad. Si el cliente puede entrar tomará un lugar en el sofá o se quedará parado (si el sofá está completamente ocupado). Cuando uno de los peluqueros está libre, atenderá a un cliente que se encuentre esperando en el sofá, y como se liberó una posición en el sofá, si hay alguien esperando de pie se ubicará en la posición liberada. Cuando se termina de cortar el pelo al cliente, éste debe pasar por la caja a pagar (que es administrada por Don Mateo).

Se pide que resuelva la sincronización utilizando semáforos.

Dentro de la solución presentada deberán estar los valores iniciales de los semáforos, el protocolo de sincronización que debe usar cada proceso para acceder a las regiones críticas, cuáles son los recursos y cuáles los procesos (todo esto a nivel teórico).

NOTA: Si ud. realiza una codificación en algún lenguaje o pseudocódigo será considerada incorrecta sin evaluar si la solución funciona o no.

SoluciónCliente

P(e)
entrar();
P(a)
sentarse();
P(co)
V(a)
cortarse();
V(p)
V(co)

Don Mateo

P(p)
cobrar();
V(m)

Valores iniciales de semáforos:
e=25 (semáforo para ingreso al local)
a = 5 (semáforo para lugares en el sofá)
co = 3 (semáforo para lugares de anteción)
p = 0 (semáforo para habilitar a Mateo a cobrar)
m = 1 (mutex)

Ejercicio RM4: 8

Dados los siguientes semáforos con sus respectivos valores (R=1, S=1, T=0). Determine el uso de las primitivas P() y V(), que deberán hacer los procesos A, B y C, para que la secuencia de ejecución sea: B A B C (solo una vez).

SOLUCIÓN:

A	B	C	Pasos	A	B	C
---	---	---	-------	---	---	---

vendedor le informe el precio del mismo. Informado el precio del producto por parte del vendedor, el alumno buscará el dinero para pagar. Mientras tanto el vendedor esperará a que el alumno abone el precio del producto. Una vez con el producto en sus manos, el alumno deberá esperar lugar para sentarse en el buffet en caso de no haber, para poder consumir su pedido. Luego podrá marcharse del buffet.

SOLUCIÓN:

```

atencionVendedor = 1;
lugarSentarse = 20;
pagar = 0;
precio = 0;
producto = 0;

void vendedor() {
    while(1) {
        wait(producto);
        buscarProducto();
        buscarPrecio();
        signal(precio);
        wait(pagar);
    }
}

void alumno() {
    wait(atencionVendedor);
    elegirProducto();
    signal(producto);
    wait(precio);
    buscarDinero();
    signal(pagar);
    signal(atencionVendedor);
    tomarProducto();
    wait(lugarSentarse);
    sentarse();
    consumirProducto();
    signal(lugarSentarse);
    dejarBuffet();
}

void main() {
    parbegin(vendedor(), alumno(), ..., alumno());
}

```

Ejercicio RM4: 12

Problema de una peluquería (Método de solución: Semáforos).

Una peluquería tiene tres sillas, tres peluqueros, y un área de espera que puede recibir a cuatro clientes en un sofá y que tiene una sala para la gente parada. El código de incendios limita el total de clientes en un negocio a 20. En este ejemplo, nosotros asumimos que la peluquería atenderá eventualmente 50 clientes.

Un cliente no entrará al negocio si la capacidad de clientes está llena. Una vez dentro, el cliente toma asiento en el sofá o se queda parado, si el sofá está lleno. Cuando un peluquero está libre, el cliente que ha estado más tiempo en el sofá es atendido y, si hay algunos clientes parado, el que más tiempo lo ha estado se sienta en sofá. Cuando el corte del cliente es terminado, cualquier peluquero puede aceptar el pago, pero como sólo hay una máquina registradora, los pagos son aceptados uno a la vez. Los peluqueros dividen su tiempo entre cortar el pelo, cobrar el pago, y dormir en su silla esperando por un cliente.

Solución de una peluquería injusta.

Se asume que todas las colas de los semáforos son manejadas bajo la política first-in-first-out.

El cuerpo principal del programa activa 50 clientes, 3 peluqueros, y un proceso de cajero. Ahora consideraremos el propósito y rol de los diferentes operadores de sincronización:

- **Capacidad del negocio y del sofá:** la capacidad del negocio y la capacidad del sofá están manipuladas por los semáforos *max_capacity* y *sofa*, respectivamente. Siempre que un cliente intenta entrar al negocio, el semáforo *max_capacity* es decrementado en 1; siempre que un cliente se va, el semáforo es incrementado en 1. Si un cliente encuentra el negocio lleno, entonces ese proceso de cliente es suspendido en *max_capacity* por la función *wait*. De forma similar, las operaciones *wait* y *signal* implementan las acciones de sentarse y pararse del sofá.
- **Capacidad de la silla del peluquero:** hay tres sillas de peluquero, y hay que tener cuidado para que sean usadas eficientemente. El semáforo *barber_chair* asegura que no más de tres clientes intenten obtener servicio al mismo tiempo, tratando de evitar la indignante situación de un cliente sentado sobre otro. Un cliente no se parará del sofá hasta que al menos una silla esté libre [*wait (barber_chair)*], y cada señal de peluquero cuando un cliente se ha dejado su silla [*signal(barber_chair)*]. El acceso justo a las sillas del peluquero está garantizado por organización de las colas de semáforos: el primer cliente que fue bloqueado es el primero en permitirle sentarse en la silla disponible.

Ejercicio RM4: 13

Existe un aeropuerto que se utiliza como base de operaciones de una flota de aviones. En este aeropuerto tenemos por un lado los aviones, las 10 pistas de aterrizaje / despegue y dos controladores aéreos encargados de gestionar todas los pedidos de los aviones. Uno de ellos se encarga de los pedidos de pistas para aterrizaje o despegue y otro de la liberación de las pistas cuando los aviones han finalizado dichas maniobras. Las pistas se pueden utilizar para despegar o aterrizar según se desee. Para poder utilizar una pista, los aviones deben de solicitarla previamente al controlador de entrada y, una vez que hayan aterrizado o despegado, avisar al controlador de salida devolviéndole la pista al conjunto de pistas libres. Cada avión, en un principio, se encuentra en el hangar realizando tareas de mantenimiento, para más tarde pasar a solicitar una pista a la torre de control. El avión entonces usa la pista para despegar y avisa al controlador que deje la pista libre. Cuando un avión decide aterrizar, realizará idénticas acciones a las que se acaban de describir.

Teniendo en cuenta el siguiente código, se pide que lo sincronice conveniente utilizando semáforos para que no se produzca ni deadlock, ni starvation, indicando el tipo y los valores iniciales de los mismos.

Avión	Controlador Entrada	Controlador Salida
while (true)	while (true)	while (true)
{	{	{
mantenimiento_en_hangar();	pistas_libres--;	pistas_libres++;
despega();	}	}
vuela();		
Aterriza();		
}		

Resolución:

Pistas_libres = 10

Disponible = 0

Entrada = 0

Liberar = 0

Avión	Controlador Entrada	Controlador Liberacion
<pre>while (true) { mantenimiento_en_hangar(); signal (entrada) down (disponible) despega(); signal(liberar) vuela(); signal (entrada) down (disponible) aterriba() signal(liberar) }</pre>	<pre>while (true) { down (entrada) down (pistas_libres); signal(disponible) }</pre>	<pre>while (true) { down (liberar) signal (pistas_libres); }</pre>

Ejercicio RM4: 14

Dados los siguientes procesos que se ejecutarán en el orden dado, realizar la traza de ejecución informando qué ocurre con dichos procesos.

p1 ()	p2 ()	p3 ()
{	{	{
wait (S1);	wait (S2);	wait (S3);
:	wait (S1);	signal (S1);
wait (S2);	:	}
:	signal (S2);	
signal (S1);	signal (S1);	
signal (S2);	}	
}		

Respuesta:

P1	P2	P3	S1=0	S2=1	S3=1
Wait(S1)			-1		
encolado	Wait(S2)			0	
	Wait(S1)		-2		

	encolado	Wait(S3)			0
		Signal(S1)	-1		
Wait(S2)		finaliza		-1	
encolado					

Los procesos p1 y p2 no terminan de ejecutarse dado que sufren deadlock al competir por los recursos s1 y s2.

_____ ○ ○ _____

Ejercicio RM4: 15

Dada la siguiente secuencia lógica y los valores iniciales:

X	Y	Z	Q
P(B)		P(S)	P(C)
P(T)	
...
V(C)		V(T)	V(D)
			V(B)

Valores iniciales S =1, T=0, B=1, C=0, D=0. Explique cuál es la secuencia normal de ejecución.

RESPUESTA:

Para encontrar la secuencia normal de ejecución debemos seguir los valores de los semáforos habilitados. Como S=1 y B=1 podríamos comenzar por la ejecución con X, pasa el semáforo B y se bloquea en el semáforo T, por lo tanto solo podrá ejecutar Y primero, cuando este finaliza habilita el sem. T, entonces ejecuta X. Este habilita el semáforo C, entonces puede ejecutar Z, que habilita el sem D y puede ejecutar Q que deja B=1.

La secuencia normal es YXZQ no repitiéndose la secuencia después de haberlo hecho una sola vez.

Para que se repita, el proceso Q debería ejecutar una función V(s) al final.

_____ ○ ○ _____

Ejercicio RM4: 16

En cierta industria se dispone de 2 balanzas electrónicas conectadas a una computadora. Cada balanza entrega la información (peso) 1 segundo después de que se le coloque el objeto a pesar. (La función **esperar_medicion()** es bloqueante). Cada balanza trabaja en forma independiente.

El siguiente código tiene por objetivo almacenar la información recibida de cada balanza e imprimirla en un dispositivo especial de salida.

Tenga en cuenta que la función **imprimir()** lee la información de la memoria compartida.

```
#include <stdio.h>
#include <sys/types.h>
#include <unistd.h>
#include <signal.h>
#include "semaforo.h"
int b1, b2;
int main()
{
    float m;
    inicializar_variables();

    if (fork()==0)
        while (true)
        {
            m=esperar_medicion(b1);
            actualizar_mem_compartida(m);
            imprimir();
        }
    else
        if (fork()==0)
            while (true)
            {
                m=esperar_medicion(b2);
                actualizar_mem_compartida (m);
                imprimir();
            }
}
```

}

return 0;

Se pide:

- a). El proceso tiene un problema de sincronización, indique qué problema y cómo lo solucionaría (Indique claramente qué haría y en qué línea/s de código).
- b). ¿Es posible que quede algún proceso como zombie?
- c). Indique claramente cual/es son los métodos de IPC que se utilizan en el ejercicio.

Resolución:

a) La actualización de la memoria compartida no está sincronizada. Usaría un semáforo mutex entre las sentencias actualizar_mem_compartida() e imprimir().

b) No, el padre muere y los hijos quedan corriendo. Pasan a ser demonios.

c) Share Memory y Semáforos (se deberían usar).

Ejercicio RM4: 17 Final 29/12/01

Resuelva la siguiente tabla sabiendo que el semáforo C esta inicializado en 1 y el resto se encuentran en 0.

P1	P2	P3	P4	P5
D(M)	D(E)	D(R)	D(B)	D(C)
D(I)	D(B)	D(I)	D(c)	U(A)
U(R)	D(A)	D(J)	D(O)	U(I)
U(J)	U(E)	U(E)	U(E)	U(C)
U(I)	D(L)	U(I)	D(P)	U(M)
U(P)	U(A)	U(B)	U(L)	U(O)

Resolución:

INSTANTE	READY	RUNNING	SEMAFOROS											COLA de SEMAFORO	ESTADOS
			A	B	C	E	J	I	L	M	O	P	R		
T inicial	P1,P2,P3,P4,P5	-----	0	0	1	0	0	0	0	0	0	0	0	-----	-----
t1	P2,P3,P4,P5	P1 D(M)								-1				M{P1}	P1=BLOCKED
t2	P3,P4,P5	P2 D(E)				-1								E{P1}	P2=BLOCKED
t3	P4,P5	P3 D(R)											-1	R{P3}	P3=BLOCKED
t4	P5	P4 D(B)		-1										B{P4}	P4=BLOCKED
t5	VACIA	P5 D(C)			0									C{}	
t6	VACIA	P5 U(A)	+1											A{}	
t7	VACIA	P5 U(I)						+1						I{}	
t8	VACIA	P5 U(C)			1									C{}	
t9	VACIA	P5 U(M)								0				M{}	P1=READY
t10	P1	P5 U(O)								+1				O{}	P5 TERMINA
t11	VACIA	P1 D(I)						0						I{}	
t12	VACIA	P1 U(R)											0	R{}	P3=READY
t13	P3	P1 U(J)					+1							J{}	
t14	P3	P1 U(I)						+1						I{}	
t15	P3	P1 U(P)										+1		P{}	P1 TERMINA
t16	VACIA	P3 D(I)						0						I{}	
t17	VACIA	P3 D(J)					0							J{}	
t18	VACIA	P3 U(E)				0								E{}	P2=READY
t19	P2	P3 U(I)						+1						I{}	
t20	P2	P3 U(B)		0										B{}	P4=READY y P3 TERMINA
t21	P4	P2 D(B)		-1										B{P2}	P2=BLOCKED
t22	VACIA	P4 D(c)			0									C{}	
t23	VACIA	P4 D(O)									+2			O{}	
t24	VACIA	P4 U(E)				+1								E{}	
t25	VACIA	P4 D(P)										0		P{}	
t26	VACIA	P4 U(L)							+1						P4 TERMINA

Bloqueados: P2 EN EL SEM B,

Deadlock: NO

Starvation: NO

Termina: P1, P3, P4 y p5

Ejercicio RM4: 18 Final 17/02/2001

De existir, muestre una secuencia de eventos para los cuales P1 , y P2 entran en DEADLOCK. Los semáforos son: s1, s2 (Semáforos Mutex).

```

Void P1() {
    While(1) {
        Down (s1);
        Down (s2);
        Printf("Bambu");
        Up (s1);
        Up (s2);
    }
}

```

```

Void P2() {
    While(1) {
        Down (s2);
        Printf("Guetzel");
        Down (s1);
        Up (s2);
        Up (s1);
    }
}

```

Respuesta:

El enunciado nos dice que ambos semáforos (s1 y s2) son Mutex. Esto quiere decir que se encuentran inicializados en 1.

Ambos procesos (p1 y p2) se ejecutan al mismo tiempo. Cuando P1 ejecuta la sentencia: **Down (s1)**; S1 que estaba en 1 pasa a estar en 0 y supongamos que en ese instante se le acaba el quantum de tiempo y ejecuta P2 haciendo un **Down (s2)**; S2 estaba en 1 pasa a estar en 0 y cuando P2 ejecute S1 queda bloqueado. Cuando le vuelve a tocar el procesador a P1 ejecuta S2 y queda bloqueado

Entonces ambos procesos quedan en **DEADLOCK**

Ejercicio RM4: 19 **FINAL DEL 18/02/2006**

Indicando la traza de ejecución, muestre en forma CLARA de que manera se ejecutarán los siguientes procesos, considerando que lo hacen concurrentemente en un sistema multiprogramado. Detalle que procesos finalizaron y cuáles no y por que razón. Para que se considere aprobado el punto deberá justificar su conclusión con un grafo de asignación de recursos, caso contrario el ejercicio se evaluará como incorrecto en su TOTALIDAD. Adicionalmente, considere que el sistema operativo no libera los recursos que tienen asignados los procesos cuando finalizan.

Inicialización de los semáforos:

I, B, A, C, P = 0

$$M, K, G = 1$$

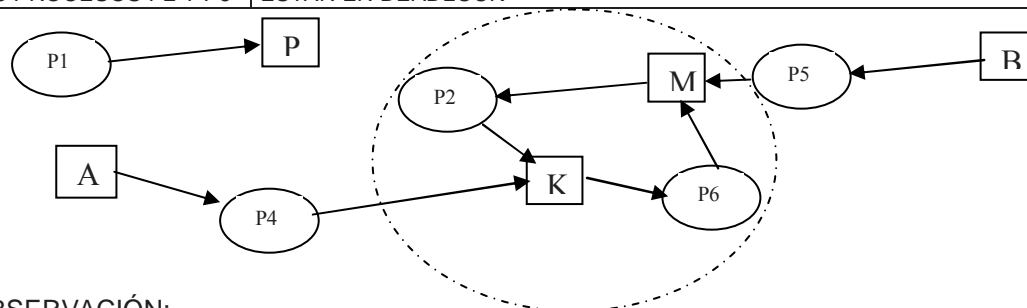
P1	P2	P3	P4	P5	P6
D (I)	D (M)	U (B)	D (A)	U (C)	D (K)
U (I)	U (A)	D (C)	D (A)	D (B)	U (A)
D (P)	D (I)	D (G)	U (I)	D (M)	D (I)
U (P)	U (I)	U (G)	D (K)	D (G)	U (I)
D (K)	D (K)		U (K)	U (M)	D (M)
	U (M)		D (P)	D (K)	U (K)
			U (P)	U (G)	U (P)

Instante	READY QUEUE	RUNNING	SEMAFOROS								COLA DE SEM.	Estado
			I	B	A	C	P	M	K	G		
Inicial	P1,P2,P3,P4,P5,P6		0	0	0	0	0	1	1	1		
t ₁	P2,P3,P4,P5,P6	P1(DI)	-1								I={P1}	P1 = Blocked
T ₂	P3,P4,P5,P6	P2(DM)						0			M={}	
T ₃	P3,P4,P5,P6	P2(UA)			1						A={}	
T ₄	P3,P4,P5,P6	P2(DI)	-2								I={P1,P2}	P2 = Blocked
T ₅	P4,P5,P6	P3(UB)		1							B={}	
T ₆	P4,P5,P6	P3(DC)				-1					C={P3}	P3 = Blocked
T ₇	P5,P6	P4(DA)			0						A={}	
T ₈	P5,P6	P4(DA)			-1						A={P4}	P4 = Blocked
T ₉	P6	P5(UC)				0					C={}	P3 = Ready
t ₁₀	P6, P3	P5(DB)		0							B={}	
t ₁₁	P6, P3	P5(DM)						-1			M={P5}	P5 = Blocked
t ₁₂	P3	P6(DK)							0		K={}	
t ₁₃	P3	P6(UA)			0						A={}	P4 = Ready
t ₁₄	P3,P4	P6(DI)	-3								I={P1,P2,P6}	P5 = Blocked
t ₁₅	P4	P3(DG)								0	G={}	

t ₁₆	P4	P3(UG)								1	G={}	P3 = TERMINA
t ₁₇	VACIA	P4(UI)	-2								I={P2,P6}	P1 = Ready
t ₁₈	P1	P4(DK)							-1		K={P4}	P4 = Blocked
t ₁₉	VACIA	P1(UI)	-1								I={P6}	P2 = Ready
T ₂₀	P2	P1(DP)					-1				P={P1}	P1 = Blocked
T ₂₁	VACIA	P2(UI)	0								I={}	P6 = Ready
T ₂₂	P6	P2(DK)							-2		K={P4,P2}	P2 = Blocked
T ₂₃	VACIA	P6(UI)	1								I={}	
T ₂₄	VACIA	P6(DM)							-2		M={P6}	P6 = Blocked

Resumen

PROCESO P3	TERMINA EN EL INSTANTE 16
PROCESO P1	ESTA BLOQUEADO ESPERANDO QUE SE LIBRE EL RECURSO P
PROCESO P5	ESTA BLOQUEADO ESPERANDO QUE SE LIBRE EL RECURSO M
PROCESO P3	ESTA BLOQUEADO ESPERANDO QUE SE LIBRE EL RECURSO K
LOS PROCESOS P2 Y P6	ESTAN EN DEADLOCK



OBSERVACIÓN:

Los procesos P1, P4 Y P5 ESTAN BLOQUEADOS aunque la liberación de recursos (P, B y A) depende del orden de ejecución externa o de los procesos involucrados en el conflicto, técnicamente no están involucrados en el DEADLOCK como sí lo están M y K

Ejercicio RM 4: 20 FINAL 17/02/2001

De existir, muestre una secuencia de eventos para los cuales P1 , y P2 entran en DEADLOCK. Los semáforos son: s1, s2 (Semáforos Mutex).

```

Void P1() {
    While(1) {
        Down (s1);
        Down (s2);
        Printf("Bambu");
        Up (s1);
        Up (s2);
    }
}

Void P2() {
    While(1) {
        Down (s2);
        Printf("Guetzel");
        Down (s1);
        Up (s2);
        Up (s1);
    }
}

```

Respuesta:

El enunciado nos dice que ambos semáforos (s1 y s2) son Mutex. Esto quiere decir que se encuentran inicializados en 1.

Ambos procesos (p1 y p2) se ejecutan al mismo tiempo. Cuando P1 ejecuta la sentencia: **Down (s1);** S1 que estaba en 1 pasa a estar en 0 y supongamos que en ese instante se le acaba el quantum de tiempo y ejecuta P2 haciendo un **Down (s2);** S2 estaba en 1 pasa a estar en 0 y cuando P2 ejecute S1 queda bloqueado. Cuando le vuelve a tocar el procesador a P1 ejecuta S2 y queda bloqueado. Entonces ambos procesos quedan en **DEADLOCK**

Ejercicio RM 4: 21 FINAL del 07/12/2002

Dado el clásico problema de productores y consumidores, se plantea una solución utilizando semáforos con el siguiente código:

```

typedef int semaforo;
typedef char* msg;
int N=100; /*Longitud del buffer */
semaforo mutex = 1; /*Da la exclusión mutua */
semaforo lleno = 0; /*Cuenta lugares llenos */

```

semaforo vacio = N; /*Cuenta lugares vacíos */

```

Productor()
{
    msg mensaje;
    while(TRUE)
    {
        producir(mensaje);
        down(&mutex);
        down(&vacio);
        entrar_msg(mensaje);
        up(&mutex);
        up(&lleno);
    }
}

Consumidor()
{
    msg mensaje;
    while(TRUE)
    {
        down(&lleno);
        down(&mutex);
        remover_msg(mensaje);
        up(&mutex);
        up(&vacio);
        consumir_msg(mensaje);
    }
}

```

¿La solución planteada es válida?. En caso de que no lo sea, explique por qué.

SOLUCIÓN

Se queda en deadlock.

si el productor da dos vueltas seguidas se bloquea en down(&vacio) después de hacer el down(&mutex) y el consumidor se bloquea en el down(&mutex) después de hacer el down(&lleno)... entonces productor está esperando el VACIO y consumidor está esperando el MUTEX.

Para el ejemplo supongo que primero se inicia el productor y corre hasta bloquearse y después se inicia el consumidor.

Productor	Consumidor	Mutex	Vacio	Lleno
Se inicia el productor		1	1	0
down mutex		0	1	0
down vacio		0	0	0
up mutex		1	0	0
up lleno		1	0	1
down mutex		0	0	1
down vacio (se bloquea)		0	0	1
	se inicia el consumidor	0	0	1
	down lleno	0	0	0
	down mutex (se bloquea)	0	0	0
LOS DOS ESTAN BLOQUEADOS	LOS DOS ESTAN BLOQUEADOS			

Ejercicio RM 4: 21 FINAL del 02/10/2003

Un pequeño centro de ski cuenta con 1 aerosilla marca Lazzeri con capacidad para 1 sola persona. Si se tiene los pseudo códigos de los procesos aerosilla y esquiador, se pide que sincronice **convenientemente usando semáforos**, para que no produzca Deadlock, ni Starvation.

```

void pasajero()
{
    while (1)
    {

        llegar_a_la_aerosilla()
    }
}

```


t ₉	P6	P5 U(C)					-1			I{P2}	P1=READY
t ₁₀	P6, P1	P5 D(P)							-1	P{P5}	P5=BLOCKED
t ₁₁	P1	P6 D(K)					0			K{}	
t ₁₂	P1	P6 U(A)	0							A{}	P4=READY
t ₁₃	P1,P4	P6 D(I)					-2			I{P2,P6}	P6=BLOCKED
t ₁₄	P4	P1 U(I)					-1			I{P6}	P2=READY
t ₁₅	P4,P2	P1 D(P)							-2	P{P5,P1}	P1=BLOCKED
t ₁₆	P2	P4 U(I)					0			I{}	P6=READY
t ₁₇	P2,P6	P4 D(K)							-1	K{P4}	P4=BLOCKED
t ₁₈	P6	P2 U(I)					+1			I{}	
t ₁₉	P6	P2 D(P)							-3	P{P5,P1,P2}	P2=BLOCKED
t ₂₀	VACIA	P6 U(I)					+1			I{}	
t ₂₁	VACIA	P6 D(M)							-1	M{P6}	P6=BLOCKED

Bloqueados: P1,P2 Y P5 EN EL SEM. P; P3 EN EL SEM. C; P4 EN EL SEM. K; y P6 EN EL SEM. M

Deadlock: SI

Starvation: NO

Termina NINGUNO



Ejercicio RM4: 22 Final del 25/02/2006

En la Provincia de Buenos Aires existe un local denominado “El Bar”, donde se sirven bebidas alcohólicas sin contar con la debida habilitación. La barra del bar se encuentra atendida por tres barmans que realizan, a pedido de los clientes, tragos con mezclas de seis bebidas (Whisky, Kalua, Cognac, Blue Curacao, KoKa-Kola Lait y Espid). Los clientes piden su consumición a un barman libre, que será una combinación cualquiera de dos de las seis bebidas disponibles. En caso que todos los barmans se encuentren ocupados, los clientes esperan a que se libere uno de los tres. El barman, una vez que recibe el pedido, toma las botellas correspondientes y realiza el trago. Mientras tanto, si otro barman recibe un nuevo pedido que requiere el uso de una de las botellas que se está utilizando, debe esperar pacientemente a que el otro termine de hacer su trago. Se supone que el máximo número de clientes en el local está limitado a 100 y cada cliente puede pedir tantas combinaciones –gratis- como quiera. La policía, que sospecha que en el bar se distribuye alcohol sin permiso, puede entrar en cualquier momento en el bar. Cuando ello ocurre, los tres barmans sirven a todo el mundo Koca-Kola Lait, independientemente de lo que hayan pedido. Se tienen los siguientes pseudo códigos del barman y de la persona que simulan dicha situación:

Valores iniciales de los semáforos:

Cant-cli = 100 (semáforo general);

Botellas [6] = {1,1,1,1,1,1} (semáforos Mutex)

Cant-Barman = 3 (semáforo general)

Preparado = 1 (semáforo general)

<pre> Proceso_Persona (numero_cliente) while(1) { pide_consumicion (numero_cliente, bebida1, bebida2); bebe(); } </pre>	<pre> Proceso_Barman () while(1) { recibe_pedido (numero_cliente, bebida1, bebida2); if (policia) { real1= "KOKA LAIT"; real2= " "; } else { real1= bebida1; real2= bebida2; } mezcla (mezcla (real1,real2)); } </pre>
---	---

Teniendo en cuenta los datos mencionados, se pide que incorpore los semáforos que crea conveniente, declarando el valor y el tipo de cada uno de los mismos, para que las acciones se lleven a cabo normalmente.

SOLUCIÓN

<pre> Proceso_Persona (numero_cliente) while(1) { down (cant-cli); down (cant-Barman); pide_consumicion (numero_cliente, bebida1, bebida2); Up (Pedido) } </pre>	<pre> Proceso_Barman () while(1) { down (pedido); recibe_pedido (numero_cliente, bebida1, bebida2); if (policia) { </pre>
---	---


```

        printf("\n soy el proceso P \n");

        operacion.sem_num=S2;
        operacion.sem_op = 1;
        operacion.sem_flg=0;
        semop(IdentificadorDeSemaforos,&operacion,1);

    exit(0);
    break ;
}
exit(0);
break;
default: // Proceso Q
    operacion.sem_num=S3;
    operacion.sem_op = -1;
    operacion.sem_flg=0;
    semop(IdentificadorDeSemaforos,&operacion,1);

    printf("\n soy el proceso Q \n");

    operacion.sem_num=S1;
    operacion.sem_op = 1;
    operacion.sem_flg=0;
    semop(IdentificadorDeSemaforos,&operacion,1);

    semctl(IdentificadorDeSemaforos,1,IPC_RMID,0);
    semctl(IdentificadorDeSemaforos,2,IPC_RMID,0);
    semctl(IdentificadorDeSemaforos,0,IPC_RMID,0);
    exit(0);
    break ;
}
}

```

Indicar la opción correcta .

El código:

- Ejecuta el hijo y entra en deadlock el padre con el nieto
- No ejecutan ninguno de los 3 procesos porque no hay ningún semáforo inicializado en 1.
- Ejecuta la secuencia P Q R.
- Ejecuta la secuencia P R Q.
- Ejecuta una secuencia no indicada anteriormente. Indicar cual sería.

RESOLUCIÓN: pto d) Dado el siguiente escenario :

Padre	Hijo	nieto
Q	P	R

P(s3)	P(s1)	P(s2)
.	.	.
.	.	.
V(s1)	V(s2)	V(s3)

vi: s1=1 ; s2=0 ;s3=0

La secuencia correcta es P R Q. Porque solo esta habilitado P inicialmente, luego habilita a s2 , ejecuta R y luego R habilita a s3 pudiendo comenzar Q.

Ejercicio RM4: 24

Un edificio inteligente posee un tanque de agua en la terraza y uno en el subsuelo que se utiliza como tanque de bombeo.

El modo de funcionamiento es el siguiente: una bomba eléctrica toma agua del tanque de bombeo y la conduce hacia el tanque de la terraza.

Una vez que el tanque superior se llena, la bomba deja de funcionar.

Por otro lado, cuando el tanque del subsuelo queda con un nivel de agua inferior a un límite establecido, ingresa agua de la red hasta llenarse.

Tanto la bomba como los mecanismos de ingreso/corte de cada tanque están manejados por una computadora. La misma posee procesos con las siguientes funciones:

```

permitir_ingreso_de_agua( ) ;           // Representa la acción de permitir el ingreso de agua al tanque del
subsuelo.
bombear_al_tanque_superior( ) ;         // Representa la acción de bombeo al tanque superior.
cosumir( ) ;                           // Representa el consumo de agua del edificio.
    
```

Se pide que coloque los semáforos necesarios para mantener el sistema estable y en continuo funcionamiento. Indique, además, los valores iniciales de los semáforos y de qué tipo son.

Resolución

P(tanque_subsuelo_vacio) ;	P(tanque_superior_lleno) ;	P(tanque_subsuelo_lleno) ;
permitir_ingreso_de_agua() ;	cosumir() ;	P(tanque_superior_vacio);
V(tanque_subsuelo_lleno) ;	V(tanque_superior_vacio) ;	bombear_al_tanque_superior() ;
		V(tanque_superior_lleno);
		V(tanque_subsuelo_vacio);

Ejercicio RM4: 25



EJERCICIOS SIN RESOLVER DE SINCRONIZACIÓN

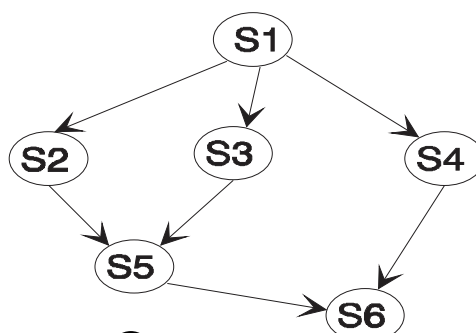
Ejercicio M4: 1

Dado el siguiente conjunto de instrucciones, determinar por medio de las condiciones de Bernstein cuáles pueden ser ejecutadas en forma concurrente. Construya el grafo de precedencia.

S1. $a = \sin(c)$;
 S2. $c = \tan(b)$;
 S3. $d = e + f$;
 S4. $f = g * h$;

Ejercicio M4: 2

Transformar el siguiente grafo de precedencia usando las sentencias parbegin - parend, y las sentencias fork - join.



Ejercicio M4: 3

Explique qué pasa con las siguientes rutinas si los valores de los semáforos son $T = 1$, $S = 0$, $R = 0$.

A	B	C
P(R)		P(S)
P(S)		P(T)
P(T)	---	---
---	---	---
V(T)	V(S)	V(S)
V(S)		V(R)
V(R)		

Ejercicio M4: 4

Dados los valores iniciales de cuatro semáforos ($R=0$, $S=0$, $T=1$, $U=1$), se pide que arme los procesos, y asigne los semáforos de forma tal que la secuencia normal de ejecución sea CABD (Una vez que los procesos ingresan a su RC, terminan sin interrupciones).

Ejercicio M4: 5

Un restaurante de comida rápida tiene cuatro tipos de empleados : (1) Los tomadores de orden, que toman las ordenes de los clientes ; (2) Cocineros, quienes preparan la comida ; (3) Empaquetadores, quienes guardan la comida en cajas ; (4) Cajeros, quienes dan las cajas a los clientes y toman su dinero. Cada empleado puede considerarse como un proceso secuencial que se comunica con los demás. ¿Cuál es la forma de comunicación utilizada entre los procesos ?

Ejercicio M4: 6

Explique que tipo de comunicación se utiliza entre un servidor de correo y el cliente (tenga en cuenta que la comunicación es bidireccional, y se deben especificar ambas).

Ejercicio M4: 7

¿Funciona la solución de espera ocupada por medio de la variable TURN cuando los dos procesos se ejecutan en dos CPU, con memoria compartida? Justifique la respuesta.

```

#include <types.h>
#define FALSE 0
#define TRUE 1
#define N 2
int turn;
int interested[N];
void enter_region(int process)
{
    int other;
    other = 1 - process;
    interested[process] = TRUE;
    turn = process;
    while (turn == process && interested[other] == TRUE);
}
void leave_region(int process)
{
    interested[process] = FALSE;
}

```

Ejercicio M4: 8

Considere el siguiente programa:

```

const n=50
var cuenta: entero;
procedure total:
    var cont: entero;
    begin
        for cont := 1 to n do cuenta := cuenta + 1;
    end;
begin (*programa principal*)
    cuenta := 0;
    cobegin total; total
    coend;
    writeln(cuenta)
end.

```

Se pide:

- Determinar los límites inferior y superior adecuados para el valor final de la variable compartida *cuenta* escrita en la salida por este programa concurrente. Supóngase que los procesos pueden ejecutar a cualquier velocidad relativa y que un valor sólo puede incrementarse después de que haya sido cargado en un registro por una instrucción separada de la máquina.
- Supóngase que se permite ejecutar en paralelo a un número arbitrario de estos procesos bajo las suposiciones del apartado (a).

Ejercicio M4: 9

Dados los valores iniciales de cuatro semáforos (R=0, S=0, T=1, U=1), Determine el uso de las primitivas P() y V(), de forma tal que la secuencia normal de ejecución sea CABD (Una vez que los procesos ingresan a su RC, terminan sin interrupciones).

A	B	C	D
P(U)	P(R)	P(T)	P(S)
P(U)	-	-	-
-	-	-	-
-	-	-	-
-	-	-	-
V(R)	V(S)	V(U)	V(U)
		V(T)	

Ejercicio M4: 10

Dados 2 procesos productores P1 y P2 y un proceso consumidor C1. Los procesos productores pueden depositar mensajes en el bufer en cualquier orden, mientras haya lugar, y el proceso consumidor puede consumir en cualquier momento siempre y cuando haya mensajes en el bufer. El bufer tiene una capacidad máxima de 3 mensajes. Solo puede acceder al bufer un proceso a la vez.

Determine los semáforos, sus valores iniciales y las primitivas necesarias para que el sistema funcione correctamente.

P1	P2	C1
P(E)	P(E)	P(D)
P(M)	P(M)	P(M)
-	-	-
-	-	-
-	-	-
V(M)	V(M)	V(M)
V(D)	V(D)	V(E)

Valores iniciales

M=1, E=3, D=0

Ejercicio M4: 11

En un sistema con 1 proceso productor P1 y 2 procesos consumidores C1 y C2. El proceso productor puede depositar mensajes en el bufer en cualquier momento, mientras haya lugar, y los procesos consumidor deben consumir manteniendo una alternancia estricta (siempre y cuando haya mensajes en el bufer). El bufer tiene una capacidad máxima de 3 mensajes. Solo puede acceder al bufer un proceso a la vez.

Determine los semáforos, sus valores iniciales y las primitivas necesarias para que el sistema funcione correctamente.

P1	C1	C2
P(E)	P(A1)	P(A2)
P(M)	P(D)	P(D)
-	P(M)	P(M)
-	-	-
-	-	-
-	-	-
V(M)	V(M)	V(M)
V(D)	V(D)	V(E)
	V(A2)	V(A1)

Valores iniciales

M=1, E=3, D=0, A1=1, A2=0

Ejercicio M4: 12

Determine los semáforos, sus valores iniciales y las primitivas necesarias para que los procesos A, B y C repitan en un ciclo infinito la secuencia ABABC

A	B	C
P(X)	P(Y)	P(Z)
P(X)		P(Z)
-	-	-
-	-	-
-	-	-
V(Y)	V(X)	V(X)
	V(Z)	V(X)

Valores iniciales:

X: 3, Y: 0, Z: 0

Ejercicio M4: 13

Dado un sistema con 2 procesos productores P1 y P2 y 2 procesos consumidores C1 y C2. Los procesos productores pueden depositar mensajes en el bufer en cualquier orden, mientras haya lugar, y los procesos consumidores puede consumir en cualquier momento siempre y cuando haya mensajes en el bufer. El bufer tiene una capacidad máxima de 3 mensajes. Solo puede acceder al bufer un proceso a la vez.

Determine los semáforos, sus valores iniciales y las primitivas necesarias para que el sistema funcione correctamente.

P1	P2	C1	C2
P(E)	P(E)	P(D)	P(D)
P(M)	P(M)	P(M)	P(M)
-	-	-	-
-	-	-	-
-	-	-	-
V(M)	V(M)	V(M)	V(M)
V(D)	V(D)	V(E)	V(E)

Valores iniciales

M=1, E=3, D=0

Ejercicio M4: 14

Se tienen 3 procesos P1, CP y C2, y 2 Bufers (A y B). EL proceso P1 produce y deposita mensajes en el bufer A en cualquier momento, mientras haya lugar. El proceso CP consume los mensajes del bufer A (puede

consumir en cualquier momento siempre y cuando haya mensajes en el bufer A) y por cada mensaje que consume del bufer A, genera y deposita un mensaje en el bufer B (debe verificar que haya espacio en el bufer B). El proceso C2 consume los mensajes del bufer B, puede consumir en cualquier momento siempre y cuando haya mensajes en el bufer B. Ambos buffers tienen una capacidad máxima de 3 mensajes (cada uno). Solo puede acceder a cada bufer un proceso a la vez.

Determine los semáforos, sus valores iniciales y las primitivas necesarias para que el sistema funcione correctamente.

P1	CP	C2
P(EA)	P(DA)	P(DB)
P(MA)	P(EB)	P(MB)
-	P(MA)	-
-	P(MB)	-
-	-	-
V(MA)	V(MA)	V(MB)
V(DA)	V(MB)	V(EB)
	V(EA)	
	V(DB)	

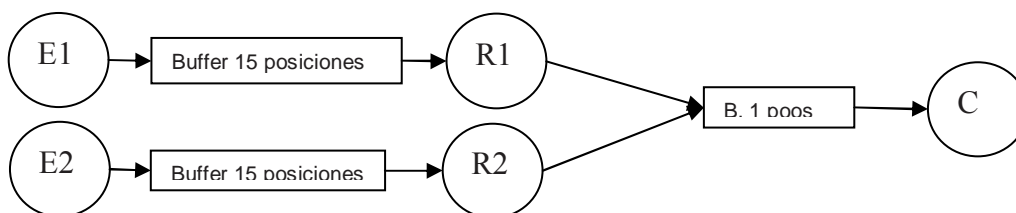
Valores iniciales

MA, MB=1 ; EA, EB=3; DA, DB=0

_____ ○ ○ _____

Ejercicio M4: 15

En un sistema de comunicaciones se posee dos procesos que envían mensajes a dos procesos receptores (que sólo trabajan de routers de los datos hacia el proceso consumidor). Cada proceso receptor comparten un único buffer de entrada de 15 posiciones. El trabajo de los receptores es el de formar un único mensaje y enviarlo al proceso consumidor. Un mensaje está formado por un mensaje recibido por el proceso receptor uno y otro mensaje recibido por el proceso receptor dos (se descarta del ejercicio la complejidad de que las posiciones de los buffers deberían ser la misma). El proceso consumidor sólo puede leer la información y consumirla si los dos receptores recibieron un mensaje, y entre estos dos tipos de procesos hay un buffer de una sola posición. Debajo se encuentra el gráfico de los procesos y los elementos que los comunican.



Se pide que realice el esquema de secuencias de los semáforos por proceso, indicando los valores iniciales de los semáforos y el tipo de cada uno.

_____ ○ ○ _____

Ejercicio M4: 16

Final:03/03/2001

Ejercicio 2

Muestre en forma clara de que manera se ejecutarán los siguientes procesos, considerando que se ejecutan en paralelo, y detalle que procesos terminaron y cuales no y por que razón.

Inicialización de los semáforos:

 $I, B, A, C, P = 0$
$$M, K, G = 1$$

P1	P2	P3	P4	P5	P6
D(I)	D(M)	U(B)	D(A)	U(C)	D(K)
U(I)	U(A)	D(C)	D(A)	D(P)	U(A)
D(P)	D(I)	D(G)	U(I)	D(M)	D(I)
U(P)	U(I)	U(G)	D(K)	D(G)	U(I)
D(K)	D(K)		U(K)	U(M)	D(M)
	U(M)		D(P)	D(K)	U(K)

Resolución:[illegible]

Dados los siguientes procesos con su respectivos códigos y los valores iniciales de los semáforos
 $A = 0$, $B = 1$, $C = 1$, $D = 0$.

X	Y	Z	W
P(C)		P(D)	P(C)
P(D)		P(A)	P(A)
.....	
V(D)	V(B)	V(A)	V(C)

a) Indique los procesos que terminaron (y en que orden) y los que no.

Secuencia: Y - Z - X - W.

b) Cambie los semáforos del ejercicio anterior para que el sistema sea SEGURO y no quede ningún proceso bloqueado.

_____ ○ ○ ○ _____

Ejercicio M4: 20

Los siguientes procesos se ejecutan en paralelo. Utilizando semáforos, especifique cuales terminaron y cuales no.

Inicialización de los semáforos: I, B, A, C, P = 0 M, K, G = 1

P1	P2	P3	P4	P5	P6
D(I)	D(M)	U(B)	D(A)	U(C)	D(K)
U(I)	U(A)	D(C)	D(A)	D(P)	U(A)
D(P)	D(I)	D(G)	U(I)	D(M)	D(I)
U(P)	U(I)	U(G)	D(K)	D(G)	U(I)
D(K)	D(K)		U(K)	U(M)	D(M)
	U(M)		D(P)	D(K)	U(K)
			U(P)	U(G)	U(P)

_____ ○ ○ ○ _____

Ejercicio M4: 21

-Realice la sincronización de 4 procesos

A ---> | 10 | ---> BDBD (una vez cada uno, en ese orden) -----> | colalimitada -----> C

-Genera mensajes	- Retira mensajes	-Retira mensajes
-Deposita mensajes	- Procesa el mens	Consume el mens.
	-Deposita mens.	

_____ ○ ○ ○ _____

Ejercicio M4: 22

Realice la sincronización de 4 procesos

A ---> | ilimitada | -----> B ---> | 5 | -----> C y D (en forma alternada estrictamente)

-Genera mensajes	- Retira mensajes	-Retira mensajes
-Deposita mensajes	- Procesa el mens	-Consume el mens.
	-Deposita mens.	

_____ ○ ○ ○ _____

Ejercicio M4: 23

Tengo un proceso productor que deposita los mensajes en un Buffer de 30 posiciones. Luego los retira y los procesa otro proceso intermediario, que va depositando los mensajes procesados en un Buffer de 5 posiciones otros dos procesos C1 y C2 los retiran e Imprimen alternadamente, C1,C2, C1,C2, C1,C2 etc.

Realizar los algoritmos para los 4 procesos.

_____ ○ ○ ○ _____

Ejercicio M4: 24

Se tienen 2 procesos paralelos:

int x = 0;

Proceso A	Proceso B
{ while (TRUE) x = x + 1; }	{ while (TRUE) print (x); }

- a) Sincronizar usando semáforos para que se escriba la secuencia: 0, 1, 2, 3, 4, 5 ...
b) Sincronizar usando semáforos para que se escriba la secuencia: 1, 4, 7, 10, 13 ...

Ejercicio M4: 25

Dados los valores iniciales de cuatro semáforos (R=0, S=0, T=1, U=1), se pide que arme los procesos, y asigne los semáforos de forma tal que la secuencia normal de ejecución sea CABD (Una vez que los procesos ingresan a su RC, terminan sin interrupciones).

Ejercicio M4: 26

Dada la siguiente solución para el problema del productor consumidor, indicar si la misma funciona correctamente o no. La respuesta deberá estar debidamente justificada para que sea tomada en cuenta.

<pre> program ProductorConsumidor ; var n :integer ; s : semaphore(:=1) ; delay : semaphore(:=0) ; </pre>	
<pre> procedure Productor ; begin repeat producir ; wait(s) ; colocar ; n := n + 1 ; if n=1 then signal(delay) ; signal(s) ; forever ; end ; </pre>	<pre> procedure Consumidor ; begin wait (delay) ; repeat wait(s) ; traer ; n := n - 1 ; signal(s) ; consumir ; if n=0 then wait(delay) ; forever ; end ; begin n := 0 ; parbegin Productor ; Consumidor ; parend ; end </pre>

Ejercicio M4: 27

Sea un sistema Operativo que utiliza un planificador con el algoritmo RR de $q=4$ y que dispone de las funciones: leer, escribir y ejecutar, así como también las funciones atómicas: up y down. Conociendo que las colas de los semáforos y la única cola de entrada/salida son independientes y como están inicializados los semáforos, indique la traza de ejecución y el orden de terminación de los siguientes procesos.

Semáforos : A=B=0; C =1;

Funciones	CPU (al inicio)	I/O (al final)
Up / Down	2	0
Leer / Escribir	1	3
Ejecutar	3	0

P1	P2	P3
Down <A>	Down 	Down <C>
Up <C>	Ejecutar <fun2>	Leer<Arch A,X>
Leer <Arch A,Y>	Up<A>	Up
Down<A>	Escribir <Arch A,Y>	Ejecutar<fun 2>
Ejecutar <fun 2>	Ejecutar<fun 1>	Escribir<ArchA,X>

Up 	Down	Ejecutar <fun1>
Down <C>	Up<C>	Up <A>

Ejercicio M4: 28

Hallar todas las posibles secuencias de ejecución mediante un diagrama explicativo.

S1=1.

Resto =0.

A	B	C	D	E	F	G	H	I
P(S11)	P(S6)	P(S4)	P(S2)	P(S3)	P(S1)	P(S7)	P(S5)	P(S8)
P(S9)				P(S10)		P(S16)		
P(S2)								
P(S15)								
P(S13)								
P(S14)								
V(S1)	V(S13)	V(S8)	V(S5)	V(S6)	V(S10)	V(S14)	V(S12)	V(S15)
			V(S11)	V(S7)	V(S4)			
					V(S16)			
					V(S2)			
					V(S3)			

Ejercicio M4: 29

Hallar las posibles secuencias de ejecución y expresarla mediante un diagrama. Los semáforos están inicializados de la siguiente manera:

S1=S2=s4=1.

Resto =0.

A	B	C	D	E	F	G
P(S5)	P(S3)	P(S7)	P(S6)	P(S1)	P(S9)	P(S2)
P(S8)	P(S10)	P(S12)			P(S4)	
P(S11)						
V(S7)	V(S5)	V(S6)	V(S12)	V(S8)	V(S11)	V(S9)
		V(S2)		V(S3)	V(S6)	V(S10)

Ejercicio M4: 30

Hallar las posibles secuencias de ejecución y expresarla mediante un diagrama. Los semáforos están inicializados de la siguiente manera:

S1=S2=S3=S4=1. Resto=0.

A	B	C	D	E	F	G	H	I
P(S11)	P(S8)	P(S12)	P(S1)	P(S3)	P(S9)	P(S10)	P(S2)	P(S4)
P(S6)	P(S14)	P(S7)				P(S5)		
	P(S13)							
V(S13)	V(S9)	V(S14)	V(S5)	V(S11)	V(S8)	V(S8)	V(S10)	V(S12)
				V(S7)	V(S7)		V(S6)	

Ejercicio M4: 31

Hallar las posibles secuencias de ejecución y expresarla mediante un diagrama. Los semáforos están inicializados de la siguiente manera:

S1=S2=1. Resto=0.

A	B	C	D	E	F
P(S6)	P(S3)	P(S2)	P(S4)	P(S1)	P(S5)
P(S7)	P(S6)				
V(S5)	V(S4)	V(S6)	V(S6)	V(S3)	V(S7)

	V(S5)				
--	-------	--	--	--	--

Ejercicio M4: 32

Se desea saber que sucede con la siguiente secuencia de semáforos, se pide la traza de ejecución. S3=1; Resto=0.

A	B	C	D	E	F
P(S3)	P(S5)	P(S1)	P(S2)	P(S6)	P(S4)
				P(S6)	
V(S6)	V(S1)	V(S2)	V(S4)	V(S1)	V(S5)
V(S5)	V(S6)	V(S5)	V(S3)		

Ejercicio M4: 33

Se desea saber que sucede con la siguiente secuencia de semáforos, se pide la traza de ejecución. S1=1; Resto=0.

A	B	C	D	E	F	G	H	I
P(S6)	P(S8)	P(S9)	P(S1)	P(S2)	P(S3)	P(S7)	P(S5)	P(S4)
		P(S9)				P(S7)		
V(S7)	V(S9)	V(S1)	V(S3)	V(S4)	V(S6)	V(S9)	V(S7)	V(S8)
			V(S2)	V(S5)				

Ejercicio M4: 34

Resuelva la siguiente traza de Sincronización considerando los siguientes valores iniciales de los semáforos: A, B, C, D, E = 0 F, J, H, I = 1

P1	P2	P3	P4	P5
DOWN (C)	DOWN (J)	DOWN (Dm)	DOWN (D)	DOWN (E)
DOWN (H)	DOWN (F)	DOWN (I)	UP (D)	UP (C)
UP (D)	UP (E)	UP (D)	DOWN (F)	DOWN (A)
DOWN (B)	DOWN (A)	UP (A)	DOWN (A)	UP (A)
UP (A)	UP (B)	DOWN (H)	DOWN (B)	
DOWN (J)	DOWN (I)	UP (I)	DOWN (I)	
UP (J)	UP (I)	DOWN (D)	UP (A)	
UP (H)	DOWN (B)	UP (D)	UP (B)	
UP (B)	UP (F)	UP (D)	UP (I)	
DOWN (F)	UP (J)	UP (H)		

Indicar el estado final de los procesos, señalando cuales finalizan y en qué orden, y cuales no, junto con la causa por la cual no lo hacen. Desarrolle la secuencia de ejecución de los procesos.

Ejercicio M4: 35

Suponga que los siguientes 2 procesos, one y two, son ejecutados concurrentemente en un Sistema Operativo preemptive y comparten los contadores generales de los semáforos A y B (ambos inicializados en 1) y la variable entera cont (inicializada en 0).

- ¿Puede ocurrir deadlock en la ejecución concurrente de los dos procesos? En caso afirmativo indicar la secuencia de ejecución que lleve a deadlock y en caso negativo explicar porque.
- ¿Puede ocurrir Starvation en alguno de los dos procesos? Explicar porque.
- ¿Considera que el acceso a los recursos criticos se encuentran correctamente sincronizado?

Proceso one repeat wait(A) writeln ("hago") if(cont > 0) then writeln("hago esto") wait(B)	Proceso two repeat while (random(0;1)<0,5 && cont!=0) { wait(B) writeln ("algo") wait(A) cont:=cont-1
--	---

cont:= cont + 1 signal(A) signal(B) forever	writeln(cont) signal(A) signal(B) } forever
--	--

Ejercicio M4: 36

Resuelva la siguiente traza de Sincronización considerando los siguientes valores iniciales de los semáforos:

J, T, S, R U = 0 H, E, C, O = 1

P1	P2	P3	P4	P5
DOWN (S)	DOWN (E)	DOWN (R)	DOWN (R)	DOWN (U)
DOWN (C)	DOWN (H)	DOWN (O)	UP (R)	UP (S)
UP (R)	UP (U)	UP (R)	DOWN (H)	DOWN (J)
DOWN (T)	DOWN (J)	UP (J)	DOWN (J)	
UP (J)	UP (T)	DOWN (C)	DOWN (T)	
DOWN (E)	DOWN (O)	UP (O)	DOWN (O)	
UP (E)	UP (O)	DOWN (R)	UP (J)	
UP (C)	DOWN (T)	UP (R)	UP (T)	
UP (T)	UP (H)	UP (R)	UP (O)	
DOWN (H)	UP (E)	UP (C)		

Indicar el estado final de los procesos, señalando cuales finalizan y en qué orden, y cuales no, junto con la causa por la cual no lo hacen. Es indispensable este análisis final, como así también indicar la secuencia de ejecución de los procesos para poder llevar a cabo la corrección.

Ejercicio M4: 37

Se plantea a los alumnos de la Universidad Tecnológica Nacional, en un parcialito de un curso de Sistemas Operativos, resolver el problema del productor/consumidor con buffer ilimitado. Se pide a los alumnos que implementen la función productor y la función consumidor utilizando semáforos y evitando que se produzcan problemas de concurrencia. Uno de los alumnos entrega la siguiente solución:

int n;

semaphore s=1;

semaphore esperar=0;

```
void productor(void)
{
  while (1)
  {
    producir();
    wait(mutex);
    añadir(buffer);
    n++;
    if (n==1) signal(esperar);
    signal(mutex);
  }
}
```

```
void consumidor(void)
{
  while (1)
  {
    wait(mutex);
    coger(buffer);
    n--;
    if (n==0) wait(esperar);
    signal(mutex);
    consumir();
  }
}
```

Dicha solución se ha corregido como incorrecta en el examen. Uno de los profesores se ha confundido y ha vuelto a tomar el mismo examen. Usted ya había visto el ejercicio porque había pedido el examen y se pone contento porque ya lo ha practicado con sus compañeros, por lo que se le pide que:

2.a) Encontrando un contraejemplo, demuestre que esta solución no es válida.

2.b) Corregir el código de manera que el problema encontrado en el punto anterior sea solucionado.

Ejercicio M4: 38

Verifique la Sincronización de los siguientes procesos. En caso de encontrar un error descríbalos mediante una traza de ejecución, sabiendo que los valores iniciales de los semáforos son: S1=1, S2 = 2

P1

```
While(1) {
  Int x;
  Down(S1);
  x=leer();
  If(x) Up(S1);
  Down(S2);
  Escribir();
  Up(S2);
}
```

P2

```
While(1) {
  Int y;
  Down(S1);
  y =escribir();
  If(y) Down(S2);
  Leer();
  Up(S2);
  Up(S1)
}
```

Ejercicio M4: 39

Dada la siguiente secuencia lógica y los valores iniciales, Se pide que explique cuál es la secuencia normal de ejecución (debe estar justificada):

X	Y	Z	Q		Val. Iniciales
P(B)	P(S)	P(C)	P(D)		S=1
P(T)	--	--	--		T=0
--	--	--	--		B=1
V(C)	V(T)	V(D)	V(B)		C=0
--	--	V(S)	--		D=0

Ejercicio M4: 40

Explique qué pasa con las siguientes rutinas si los valores de los semáforos son T = 0, S = 0, R=1.

A	B	C
P(R)	P(S)	P(T)
P(S)	P(T)	---
P(T)	---	---
---	---	---
V(T)	V(S)	V(S)
V(S)	V(R)	
V(R)		

Ejercicio M4: 41

¿Funciona la solución de espera ocupada por medio de la variable TURN cuando los dos procesos se ejecutan en dos CPU, con memoria compartida? Justifique la respuesta.

```
#include <types.h>
#define FALSE 0
#define TRUE 1
#define N 2
int turn;
int interested[N];
void enter_region(int process)
{
  int other ;
  other=1 - process ;
  interested[process] = TRUE ;
  turn = process ;
  while (turn == process && interested[other] == TRUE) ;
}
void leave_region (int process)
{
  interested[process]=FALSE ;
}
```

Ejercicio M4: 42

Supongamos que tenemos un sistema de transferencia de mensajes por medio de buzones. Un proceso no se bloquea al enviar un mensaje a un buzón completamente ocupado, o al tratar de leer de uno vacío, en lugar de ello, regresa un código de error. El proceso responde a dicho código volviendo a intentar la comunicación hasta que lo logra. ¿ Conduce este esquema a condiciones de competencia ?

Ejercicio M4: 43

Dada la siguiente solución para el problema del productor consumidor, indicar si la misma funciona o no. La respuesta deberá estar debidamente justificada para que sea tomada en cuenta.

<pre> program ProductorConsumidor ; var n :integer ; s : semaphore(:=1) ; delay : semaphore(:=0) ; procedure Productor ; begin repeat producir ; wait(s) ; colocar ; n := n + 1 ; if n=1 then signal(delay) ; signal(s) ; forever ; end ; </pre>	<pre> procedure Consumidor ; begin repeat wait (delay) ; wait(s) ; traer ; n := n - 1 ; signal(s) ; consumir ; if n=0 then wait(delay) ; forever ; end ; begin n := 0 ; parbegin Productor ; Consumidor ; parend ; end. </pre>
---	---

Ejercicio M4: 44

En un esquema automatización de noticias, en el que existen dos fuentes de información (generadores), en el que cualquiera de los dos puede colocar en un área de memoria compartida un mensaje de hasta 128 bytes. Sin importar la cantidad de veces que cada uno de los procesos generen mensajes, ni el orden de los mismos, pero nunca se puede almacenar más de un mensaje. Por otro lado existen dos procesos lectores de la información, que obligatoriamente tienen que leer la información los dos (no importa si hay orden, solo importa que lo lean los dos). Se pide que efectúe el diagrama de sincronización utilizando semáforos con espera activa para resolver el esquema propuesto. **NOTA:** Solo se tomarán en cuenta las reapiestas que incluyan el diagrama solicitado, al menos una secuencia normal que represente la solución del problema, los valores iniciales de los semáforos, y el tipo de cada uno. No se aceptan como respuestas codificaciones en pseudocódigo, o programación en "c".

Ejercicio M4: 45

En una estación de peaje hay cuatro (4) formas de pago distintas (manual con cambio, manual sin cambio, semi-automático – tarjetas, automático). Dependiendo del horario, el concesionario habilita más o menos cajas para cada una de las formas de pagos, siendo los límites inferiores y superiores los siguientes (tipo 1: 2/4, tipo 2: 2/8, tipo 3: 1/3, tipo 4: 2/4).

Se pide que represente el esquema de semáforos utilizados por los procesos, valores iniciales, la forma de que el administrador puede incrementar o decrementar la cantidad de cajas por tipo, cuales son los procesos, y cuales los recursos del sistema, indique que tipos de semáforos se usan en cada tipo. Tenga en cuenta que el administrador además quiere saber la cantidad de veces que se pasó por cada tipo de forma de pago.

Nota: No hay que codificar en pseudocódigo, o "c", solo colocar los procesos, y los semáforos, cualquier otra cosa no será tomada en cuenta.

Ejercicio M4: 46

En la famosa ruta 40, que va de este a oeste, hay que reparar un puente, lo que genera que el mismo tenga una mano inhabilitada para el tránsito. Los obreros proponen un esquema de que se coloque una bandera en el extremo este del puente, que el primer auto tome dicha bandera, y la lleve hasta el extremo oeste, donde la deja depositada en un portabandera, y que deberá ser tomada por el primer auto que desea ir en dirección oeste – este, y por supuesto que el circuito sigue así sucesivamente. Se pide que efectúe el diagrama de semáforos, procesos, y la secuencia normal que existe, y representa a la perfección el modelo. Además conteste:

- 1) ¿Cumple con el protocolo de sincronización de regiones críticas?. En caso de que no cumpla, indique como lo mejoraría para que cumpla.
- 2) ¿Explique que modificación (en forma teórica), realizaría para permitir que el paso sea de hasta como máximo 5 autos en cada sentido?

Ejercicio M4: 47

El Banco “SEGURO” tiene una línea de cajas formada por cuatro cajeros, a las que acceden los clientes a través de una única cola común para todas las cajas. Las cajas llaman al próximo cliente en cuanto se liberan, y no hay un orden específico de llamadas. Se pide que realice la muestra de los procesos (cajeros, y clientes), indicando los semáforos que usaría, los valores iniciales, al menos una secuencia normal, y un ejemplo de la corrida del mismo.

_____ ○ ○ ○ _____

Ejercicio M4: 48

Realizar la sincronización de cuatro procesos con las siguientes características. Dentro del resultado se debe indicar cuales son los valores iniciales de los semáforos, de que tipo son, la secuencia normal que utilizada, y el código de uso de los semáforos:

- 1) Dos procesos productores (desde ahora denominados P1, y P2), que genera números consecutivos infinitamente, y los deja en un buffer con capacidad para almacenar seis números. No importa quien de los dos produce, lo importante es producir.
- 2) Tres procesos consumidores (denominados C1, C2, y C3 respectivamente). Los procesos quitan del buffer el primer número. Cada uno de los procesos consumidores pueden actuar independientemente (no importa quien lo consuma, el tema es consumirlo). Los procesos consumidores deben contar la cantidad de números que consumió cada uno.
- 3) La velocidad relativa de los procesos es que los productores son 2 veces más rápido que los procesos consumidores, por lo que es necesario que sean sincronizados en el uso del recurso crítico.
- 4) En el recurso Buffer pueden estar al mismo tiempo un proceso productor (solo uno), y un proceso consumidor (solo uno).

_____ ○ ○ ○ _____

Ejercicio M4: 49

Describe claramente cuál es la forma de comunicación utilizada entre el cliente de un restaurante, el mozo que toma el pedido, y el cocinero que lo elabora. En la respuesta se debe indicar el tipo de comunicación, el tipo (sincrónico, asincrónico, o condicional), de cada uno de los tipos de comunicación existente, tanto para el canal de los pedidos, como para el de las respuestas.

_____ ○ ○ ○ _____

Ejercicio M4: 50

En un determinado centro de cómputos se tiene un proceso servidor que es accedido a través de una red dedicada usando de intermediario un proceso concentrador de comunicaciones.

El esquema de comunicaciones utilizado entre el concentrador de comunicaciones es que el concentrador envía un mensaje, el servidor lo procesa y le retorna el resultado del procesamiento a través de la misma línea. Es importante destacar que todos los mensajes tienen que tener un reconocimiento por parte del receptor. También dentro del procesamiento se debe tener en cuenta que el concentrador de comunicaciones no hace nada desde el momento de enviar un mensaje hasta recibir la respuesta.

El concentrador de comunicaciones recibe el mensaje a enviar a través de un buffer (con 10 posiciones para mensajes), que es llenado por cinco procesos productores.

Cuando el concentrador de comunicaciones recibe el mensaje conteniendo el resultado del procesamiento, y lo almacena en un buffer de salida (de 5 posiciones), que es consumido por dos procesos consumidores.

Se pide: Realizar la sincronización y comunicación indicando cuál sería el código (mensajes / P / V) de cada uno de los procesos existentes, cuántos y cuáles semáforos son necesarios y sus valores iniciales, y los tipos de mensajes implementados (bloqueante, etc.).

Instrucciones habilitadas para resolver el problema:

- P / V (no hace falta más explicación)
- Send / Receive / Accept (no hace falta más explicación)
- While (Ciclo con evaluación primero o al final como se prefiera)
- If (no hace falta más explicación)

_____ ○ ○ ○ _____

Ejercicio M4: 51

Suponga la siguiente variación al cuento de Caperucita Roja. Caperucita parte, como es sabido, a la casa de su abuelita llevando una cesta con comida. Cuando llega, se encuentra con que el Lobo Feroz (que se ha vuelto vegetariano y adicto a la New Age), ha tomado a la pobre anciana como rehén. El Lobo requiere para la liberación de la Abuelita, la cesta de comida (compuesta sólo de frutas y verduras) que lleva Caperucita.

Caperucita no le cree, y teme que la reciente adicción a la New Age sea otro engaño del ladino animal, por lo tanto, decide que no entregará la cesta hasta que el Lobo deje partir a su Abuelita.

Se pregunta:

- a) ¿Es posible el abrazo mortal? Justifique mencionando procesos, recursos y grafo de asignación de recursos.

- b) ¿Es posible la inanición? Justifique su respuesta.
 c) ¿Cómo variaría la respuesta b) si sabemos que los guardaparques están por llegar a la casa en cualquier momento, y mediante dardos tranquilizantes reducir al Lobo Feroz?

Ejercicio M4: 52

Dado los siguientes Procesos, **establecer Todas las posibles secuencias de Ejecución**, siendo los valores iniciales de los semáforos: $S7 = S8 = S10 = S11 = S12 = 0$: y $S9 = 1$.

U	V	W	X	Y	Z
P(s12)	P(S9)	P(S11) P(S11)	P(S8)	P(S10)	P(S7) P(S7)
....
V(S7)	V(S12) V(S8) V(S10)	V(S9)	V(S7)	V(S11)	V(S11)

Ejercicio M4: 53

En un laboratorio de física un investigador ha desarrollado dos funciones matemáticas muy complejas, con el objetivo de comprender aún más el comportamiento de sistemas dinámicos no lineales. Este investigador, desarrolló un programa en C para obtener por pantalla los resultados de éstos cálculos.

El sistema donde corre el programa solo tiene éste único proceso corriendo, el cuál únicamente realiza cálculos matemáticos y al finalizar imprime por pantalla. Este código es:

```
int main(int argc, char *argv[])
{
    int x, y;
    x=funcion1(argv[1]);
    y=funcion2(argv[2]);
    printf("x = %d - y = %d\n", x, y);
}
```

Un amigo de éste investigador (estudiante de Sistemas de Computación II), al ver el código, le asegura a su amigo que es capaz de disminuir el tiempo de procesamiento de sus rutinas, y entonces le ofrece ejecutar en el mismo equipo y bajo las mismas condiciones el siguiente código:

```
int main(int argc, char *argv[])
{
    int x, y;
    if (fork()==0)
    {
        y=funcion2(argv[2]);
        printf("y = %d\n", y);
    }
    else
    {
        x=funcion1(argv[1]);
        printf("x = %d\n", x);
    }
}
```

Para su sorpresa, éste nuevo código, es mas lento que el primero, aunque él asegura que esto es imposible, dado que con el último código debería aprovechar mejor el procesador, ya que la ejecución es concurrente.

Dado que usted es mejor estudiante que éste programador, explíqueme qué está pasando en éste sistema, y qué código es más apropiado para el investigador.

Ejercicio M4: 54

Existen N instancias de un recurso a compartir entre diferentes procesos concurrentes. Dada la siguiente posible solución para sincronizar su uso:

```
var S1,S2: semaphore;
    C:integer;
```

Con valores iniciales $S1=1$, $S2=0$ y $C=N$

Tomar-recurso:

```
P(S1);
C:= C-1;
```

```

    if C<0
    then begin
        P(S2);
        V(S1);
    end
V(S1);

```

Liberar-recurso:

```

P(S1);
C:=C+1;
If C<= 0
then V(S2)
else V(S1)

```

Responder:

- ¿Es correcta la solución planteada? Justifique.
- ¿Cómo podría proveerse una solución mucho más simple también a través de semáforos? Plantearla.

Ejercicio M4: 55

Dados los siguientes semáforos y procesos, que los buffers fuesen de múltiples instancias en lugar de una?

A	B	C	D	E
P(s1)	P(s1)	P(s3)	P(s2) P(s4) P(s5)	P(s6)
V(s7) V(s2)	V(s8) V(s2)	V(s9) V(s4)	V(s1) V(s3) V(s6)	V(s10) V(s5)

Valores iniciales: S1, S3, S5 = 1, el resto 0

Se pide que conteste las siguientes preguntas:

- Efectúe el diagrama de procesos, semáforos y recursos.
- Indique la forma normal de ejecución.
- Indique cuál es el objetivo que le encuentra a cada uno de los semáforos, y de que tipo son.
- Proponga al menos una modificación a la secuencia de los semáforos que garantice que se siga cumpliendo el objetivo, y que posibilite mejorar la performance de los procesos.
- ¿Qué modificaciones efectuaría para permitir que los buffers fuesen de múltiples instancias en lugar de una?

Ejercicio M4: 56

En la redacción de un diario existen seis fuentes de información (generadores). Cualquiera de los procesos puede colocar un mensaje en un área de memoria compartida de 1024 bytes. El tamaño del mensaje es fijo de 128 bytes. Sin importar la cantidad de veces que cada uno de los procesos generen mensajes, ni el orden de los mismos, nunca se puede almacenar más de ocho mensajes en el área de memoria compartida. Además existen tres procesos lectores que se dedican a difundir las noticias a tres destinos distintos, el área de gráfica que determina si se pone o no la noticia en el diario, el área de reportes que se dedica a decidir que noticias debe investigar más en detalle y el área de redacción web que publica las noticias en la página del diario.

Se pide que efectúe el diagrama de sincronización utilizando semáforos con espera activa o pasiva que resolver el esquema que se detalla.

NOTA: Solo se tomarán en cuenta las respuestas que incluyan el diagrama solicitado, al menos una secuencia normal que represente la solución del problema, los valores iniciales de los semáforos, y el tipo de cada uno. No se aceptan como respuestas codificaciones en pseudocódigo, o programación en "c".

Ejercicio M4: 57

Una embotelladora dispone de una máquina que se encarga de reciclar botellas usadas. La máquina trabaja de la siguiente manera:

Las botellas usadas ingresan por una cinta transportadora e al sector de lavado. Puede haber hasta 10 botellas limpiándose al mismo tiempo. Si llegaran más botellas antes que termine el ciclo de lavado, la cinta se detiene.

Luego, una vez lavadas, circulan por otra cinta hasta el sector de secado, donde pueden secarse simultáneamente hasta 5 botellas.

Igual que en el caso anterior, si el sector de secado está completo y hay más botellas limpias para secar, se detiene la segunda cinta, y por ende, de persistir la situación, terminaría por detenerse también la primera.

Se pide que:

Identifique claramente los procesos y los recursos.

Realice el protocolo de sincronización utilizando las primitivas P y V.

Indique cuántos semáforos utilizaría y de qué tipo.

Ejercicio M4: 58

Considere que estos 4 procesos se están ejecutando concurrentemente, y que en el. Sistema existen 4 semáforos inicializados en 1. Dada la siguiente secuencia de ejecución se pide determinar los posibles estados del sistema, justificando su respuesta en cada caso. Si existiera la posibilidad de que se produzca deadlock indique entre que procesos y con que recursos.

P(S)	P(X)	P(U)	P(T)
P(T)	P(U)	P(S)	P(X)
P(S)	P(X)
.....	V(S)	V(X)
V(T)	V(U)	V(U)	V(T)
V(S)	V(X)		


Ejercicio M4: 59

Dado el siguiente código conteste las preguntas, considerando que las funciones inicializar-semaforo, P() y V() ya están codificadas y sirven para inicializar, solicitar y liberar semáforos respectivamente. Por otro lado, la función actualizar-registro guarda en un archivo el parámetro recibido y la función leer-registro consulta el valor que actualmente tenga ese archivo.

El archivo mencionado solo almacena un único valor y cada vez que la función actualiza-registro es invocada, se reemplaza con el nuevo valor recibido.

- ¿Cuántos procesos se generan al ejecutar éste programa?
- ¿Qué ocurre si el planificador pone en la cola de listos a un proceso que no puede capturar un semáforo?
- ¿Qué proceso logra actualizar el archivo primero? Esto puede variar o siempre será igual en sucesivas ejecuciones?
- ¿Es posible predecir observando el código cuál será el orden de ejecución de cada proceso? De ser posible indique cuál es ese orden, y en caso contrario explique el por qué.
- En ciertas ocasiones se ha observado que las salidas de pantalla no son las esperadas. Por ejemplo, a veces aparecen mensajes idénticos en forma contigua, lo cual no parece lógico en primera instancia. ¿Puede determinar cuál puede ser el motivo de éstas variaciones esporádicas

<pre>int main() { pid_t x, y, =0 ; inicializar semaforo(a, 0) ; inicializar semaforo(b, 1) ; inicializar semaforo(c, 0) ; x=fork () ; if (x) y=fork () ; if ((x) && (y)) { for (; ;) { P(a); Actualizar_registro (getpid ()); V (b); printf("E1 valor registrado es: %d\n", leer_registro ()); fflush (stdout); } } }</pre>	<pre>else if (!x) { for (; ;) { P (b); Actualizar_registro (getpid ()); V(c); printf ("E1 valor registrado es: %d\n", leer_registro ()); fflush (stdout); } } else { for (; ;) { P(c) ; actualizar_registro (getpid ()) ; V(a) ; printf ("E1 valor registrado es: %d\n", leer_registro ()); fflush(stdout); } } }</pre>
---	---



SIGUE EN LA OTRA COLUMNA

OBSERVACIÓN: Las primitivas P() y V() son equivalentes a Down() y Up() o Wait() y Signal()

Ejercicio M4: 60

(Deadlock).

ALGORITMO DEL BANQUERO

Este algoritmo fue realizado por Dijkstra, este consiste en advertir si luego de una operación, pedido de recursos, se estará en un estado seguro, *analizando las pedidos que van a realizar los procesos; por lo que se requiere conocer de antemano las solicitudes de recursos de cada uno de ellos*, si lo cumple se le otorgara la solicitud pero de lo contrario, si esta en estado inseguro, se pospone para mas adelante.

Se dice que esta en estado seguro si el sistema se encuentra en una taza de ejecución donde todos los procesos finalizan cuando haya recursos suficientes para satisfacer todas las necesidades, *es decir, se tiene los recursos disponibles para poder satisfacer las solicitudes. Aunque suene más que lógico, un estado que no es seguro, entonces es inseguro; lo que significa que si no se puede encontrar una traza en el que todos los procesos finalicen, el estado del sistema es inseguro.*

Otra cosa a tener en cuenta es que un estado inseguro no asegura que los procesos están en deadlock. El estado inseguro alerta que si la secuencia se ejecuta de una manera determinada, los procesos pueden llegar a quedar en un estado de inanición (starvation), pero no significa que sí o sí va a suceder. Poniéndolo en claro: un estado inseguro no afirma que los procesos van a desembocar en un deadlock; puede suceder que en el momento de ejecución un proceso sea abortado y libere sus recursos, por lo que quedarán disponibles y tal vez se pueda satisfacer las peticiones del resto de los procesos.

En resumen, este algoritmo se utiliza para la prevención de posibles inconsistencias en el sistema.

Pasos del algoritmo:

Datos:

Un vector: recursos disponibles en el inicio o en ese instante

Tres Matrices: máximo pedidos, Asignado, y necesidad, con tantas columnas como tipos de recursos existan, y tantas filas como procesos existan.

El contenido de cada matriz es el siguiente:

Máximo: Cantidad máxima de recursos de cada tipo que necesita el proceso.

Asignado: lo que ya le entregado.

Necesidad: Lo que le falta pedir ($\text{Necesidad} = \text{max} - \text{asig}$)

Disponible: Vector con los tipos de recursos que existen, que indica que recursos le queda al sistema operativo para entregar.

Algoritmo: Ante un pedido $P_i()$ de un proceso, se debe analizar como quedaría el estado del sistema

1) $P_i() \leq \text{NEC}_i()$ --> Si Ir a 2 --> No Kill (Mintió en el máximo)

2) $P_i() \leq D()$ --> Si Ejecutar seguridad --> No Block (no hay recursos)

Algoritmo de seguridad: Simula que se entregan los recursos al proceso solicitante, y analiza en que estado quedaría el sistema.

1) Actualizaciones:

a) $\text{AS}_i() = \text{AS}_i() + P_i()$

b) $\text{NEC}_i() = \text{NEC}_i() - P_i()$

c) $D() = D() - P_i()$

d) Definir un vector Finish (F) con tantas posiciones como procesos, inicializado todo en Falso

2) Encontrar un i Tal que $F_i = \text{Falso}$ AND $\text{NEC}_i() \leq D()$ Si encuentra ir a 3, sino a 4

3) Como ese proceso podría terminar, lo termino, lo que para la simulación significa realizar lo siguiente:

a) $F_i() = \text{Verdadero}$ --> Lo marco como que ya finalizó

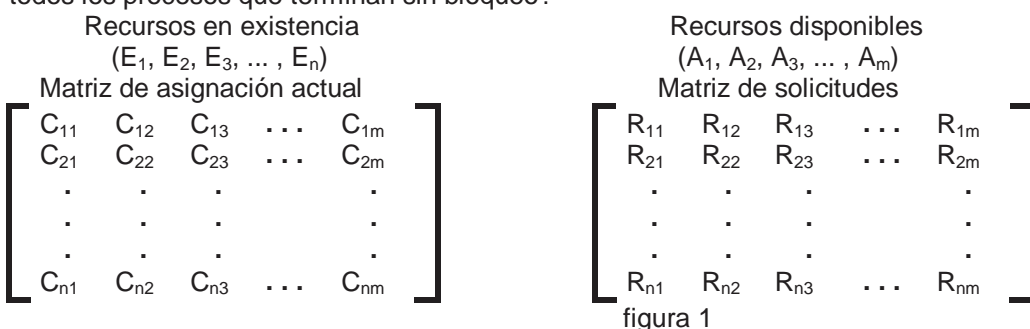
b) $D() = D() + \text{AS}_i()$ --> Como finalizó libera los recursos

4) Si $F_i() = \text{Verdadero}$ para todos los i , entonces se puede asignar, ya que el estado del sistema es seguro, porque por lo menos hay una secuencia segura de ejecución de los procesos. En caso contrario no se puede asignar, porque el estado quedaría inseguro, hay que deshacer los cambios del punto 1, y bloquear al proceso hasta que haya más recursos.

EJEMPLOS DE EJERCICIOS RESUELTOS DE DEADLOCK:

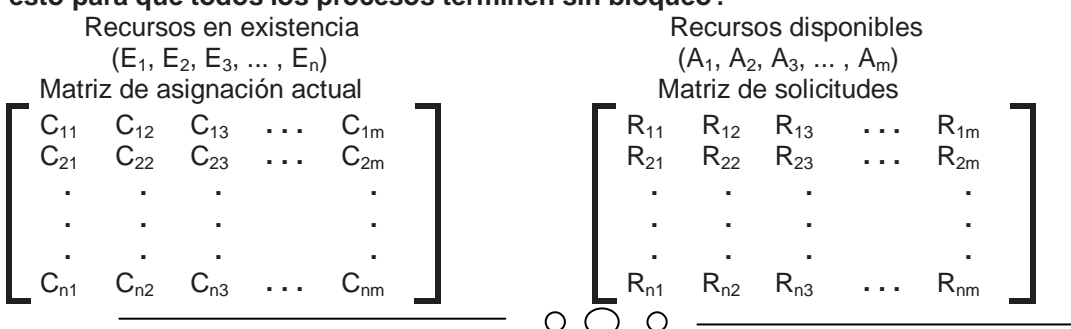
Ejercicio RM4D: 1

Caso 1: Supongamos que en la figura 1, $C_{ij} + R_{ij} > E_j$ para alguna i . ¿Cuáles son las implicaciones de esto para todos los procesos que terminan sin bloqueo?



Si se cumpliera que para algún proceso, $C_{ij} + R_{ij} > E_j$, ese proceso no podría concluir su ejecución nunca, debido a que esperaría por un tiempo infinito a un recurso que no existe. La solución a este problema es eliminar el proceso y liberar los recursos que posee para que otros puedan utilizarlos.

Caso 2). Supongamos que en la figura 1, $C_{ij} + R_{ij} \leq E_j$ para alguna i . ¿Cuáles son las implicaciones de esto para que todos los procesos terminen sin bloqueo?



Ejercicio RM4D: 2

Considere la siguiente fotografía de un sistema. No existen actualmente requerimientos en cola de espera.

- Complete la columna Still Needs
- ¿Está este sistema actualmente en estado seguro ó inseguro?. ¿Por qué?
- ¿Está este sistema actualmente en deadlock? ¿Porqué sí ó porqué no?
- ¿Cuál proceso, si es que hay alguno, que puede provocar deadlock?
- Si el requerimiento de p3 es (0,1,0,0) puede ser servido inmediatamente en forma segura? ¿En que estado queda el sistema luego de servir dicho requerimiento?

Respuestas:

PROCESO	r 1	r 2	r 3	r 4	r 1	r 2	r 3	r 4	r 1	r 2	r 3	r 4
p 1	0	0	1	2	0	0	1	2	2	1	1	2
p 2	2	0	0	0	2	7	5	0	6	7	9	8
p 3	0	0	3	4	6	6	5	6	6	7	1 2	1 2
p 4	2	3	5	4	4	3	5	6	4	4	6	6
p 5	0	3	3	2	0	6	5	2	4	7	9	8

Corre p 1 Disponible (2.1.1.2)

Corre p 4 Disponible (4.4.6.6)

Corre p 5 Disponible (4.7.9.8)

Corre p 2 Disponible (6.7.9.8)

Corre p 3 Disponible (6.7.12.12)

b. El sistema se encuentra en estado seguro, ya que existe una secuencia que puede satisfacer los requerimientos de recursos.

c. No se encuentra en *deadlock*. Esta estrategia lo que realiza es una evaluación de un posible estado de *deadlock*.

d. Si p4 solicitara una unidad más de r1, este se bloquearía y ninguno de los procesos restantes podrían ser servidos.

e. Si puede ser servido inmediatamente y el estado del sistema sigue siendo seguro.

Ejercicio RM4D: 3

○ ○ ○

Determinar el estado del siguiente sistema y mostrar los resultados.

DISPONIBLE													
					r1	r2	r3	r4					
					2	1	0	0					
		ASIGNACIÓN ACTUAL				MÁXIMA DEMANDA				STILL NEEDS			
PROCESO		r1	r2	r3	r4	r1	r2	r3	r4	r1	r2	r3	r4
p1		0	0	1	0	2	0	0	1	4	2	3	1
p2		2	0	0	1	1	0	1	0	4	2	2	1
p3		0	1	2	0	2	1	0	0	2	2	2	0

Corre p3 Disponible (2.2.2.0)

Corre p2 Disponible (4.2.2.1)

Corre p1 Disponible (4.2.3.1)

Ejercicio RM4D: 4

Determinar el estado del siguiente sistema y mostrar los resultados.

DISPONIBLE										
						r1	r2	r3	r4	r5
						0	0	0	0	1
PROCESO	ASIGNACIÓN ACTUAL					MÁXIMA DEMANDA				
	r1	r2	r3	r4	r5	r1	r2	r3	r4	r5
	p1	1	0	1	1	0	1	0	0	1
	p2	1	1	0	0	0	0	1	0	1
	p3	0	0	0	1	0	0	0	0	1
	p4	0	0	0	0	0	1	0	1	1

Corre p3 Disponible (0.0.0.1.1.)

Ningún proceso puede ser servido posterior a p3

Ejercicio RM4D: 5

Considere el siguiente sistema y determine si se encuentra en estado seguro; luego considere que el proceso 2 efectúa un requerimiento (0.0.1).

ASIGNADO				MÁXIMO			
	A	B	C		A	B	C
P0	0	1	0	P0	0	0	0
P1	2	0	0	P1	2	0	2
P2	3	0	3	P2	0	0	0
P3	2	1	1	P3	1	0	0
P4	0	0	2	P4	0	0	2

DISPONIBLE			
A	B	C	
0	0	0	

Corre p0 Disponible (0.1.0)

Corre p1	Disponibile (3.1.3)
Corre p2	Disponibile (3.1.3)

Corre p3	Disponibile (5.2.4)
----------	---------------------

Corre p1	Disponibile (7.2.4)
----------	---------------------

Corre p4	Disponibile (7.2.6)
----------	---------------------

P2 solicita (0.01) Con esta solicitud el sistema se encuentra en bloqueo mutuo.

Ejercicio RM4D: 6

¿Cuál es el valor de la matriz necesidad?. ¿Está el sistema en estado seguro?. ¿Se puede satisfacer el pedido (0, 4, 2, 0) para P1 dejando al sistema en estado seguro.

ASIGNADO					MÁXIMO				
A	B	C	D			A	B	C	D
0	0	1	2		P0	0	0	1	2
1	0	0	0		P1	1	7	5	0
1	3	5	4		P2	2	3	5	6
0	6	3	2		P3	0	6	5	2
0	0	1	4		P4	0	6	5	6

DISPONIBLE				
A	B	C	D	
1	5	2	0	

Corre p0 Disponible (1.5.3.2)
 Corre p2 Disponible (2.8.8.6)
 Corre p1 Disponible (3.8.8.6)
 Corre p3 Disponible (3.14.11.8)
 Corre p4 Disponible (3.14.12.12)

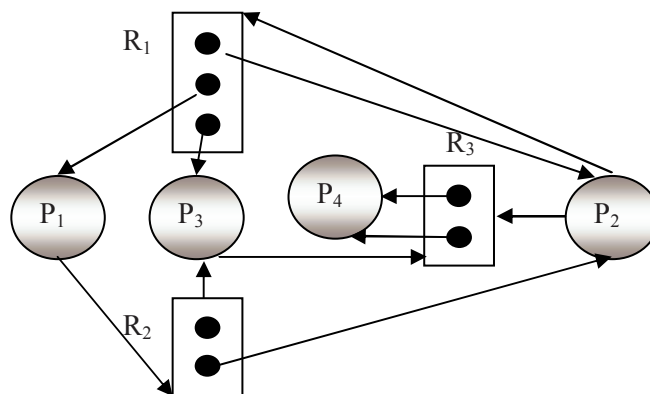
El sistema se encuentra en estado seguro ya que existe una secuencia a través de la cuál se pueden satisfacer los requerimientos del sistema.

Se puede satisfacer el requerimiento para p1 dejando al sistema en estado seguro.

Ejercicio RM4D: 7

FINAL UTN del 06/12/2003

Dado el estado de un sistema, mediante el siguiente grafo de asignación de recursos:



Suponiendo que los procesos ya no piden más recursos durante el resto de su ejecución y no liberarán sus recursos hasta su finalización, indique si puede llegar a existir Deadlock. En caso negativo, de una secuencia de finalización de los procesos; caso contrario diga cuáles procesos se encuentran en interbloqueo y proponga una solución.

SOLUCIÓN

No se produce Deadlock. Posibles Trazas de ejecución: P4, P3, P1, P2 y P4, P3, P2, P1.

Ejercicio RM4D: 8 FINAL UTN del 06/03/2003

De existir, muestre una secuencia de eventos para los cuales P1, y P2 entran en deadlock. Los semáforos son: S1, S2 (Semáforos Mutex).

```

void P1( ){
    While (1) {
        Down (s1);
        Down (s2);
        printf("Bambu");
        Up (s1);
        Up (s2); }
}
  
```

```

void P2( ){
    While (1) {
        Down (s2);
        Printf("Guetzel");
        Down(s1);
        Up (s2);
        Up (s1);}
}
  
```

SOLUCIÓN

Puede haber mas de una., EJEMPLO:

1. Proceso 1 ejecuta y hace un Down de s1.
2. Justo después de hacer el Down de s1 el SO lo schedulea.
3. Arranca Proceso 2. Hace un Down de s2, imprime "Guetzel", hace un Down de s1 y se bloquea por ser semáforos mutex.
4. Sigue el Proceso 1, hace el Down de s2 y se bloquea por ser semáforos mutex.
5. DEADLOCK.

_____ ○ ○ ○ _____

Ejercicio RM4D: 9

Contamos con 4 procesos (Arbitro, Pantalla, Teclado y PersonajeAutomatico) cada uno de los cuales necesita para poder llevar a cabo sus tareas obtener un máximo de dos (2) unidades de un recurso compartido por todos ellos. Este recurso cuenta con cinco (5) unidades que se pueden reservar e ir liberando sólo una cada vez. ¿Puede Ud. asegurar que nunca se producirá deadlock?

Rta: Si podemos asegurar que no se producirá deadlock por lo siguiente: Cada proceso no puede tomar más de una unidad del recurso a la vez y las unidades que existen son iguales a la cantidad de procesos más una. O sea que en el peor de los casos cada proceso tomará una unidad del recurso, dejando siempre una libre que será la necesaria para que finalice alguno de los procesos dejando luego 2 unidades libres al finalizar; suficiente para que las tomen otros procesos y finalicen ellos también.

_____ ○ ○ ○ _____

Ejercicio RM4D: 10

Suponga que contamos con "n" procesos que necesitan tomar 3 unidades de un determinado recurso (que comparten) para su finalización; y el recurso cuenta con 10 unidades en total. ¿Cuál será el número máximo de "n" para asegurar que no se producirá deadlock?

Rta: Si empezamos a verlo desde el principio veremos que para $n=0$, $n=1$, $n=2$ y $n=3$ nunca se producirá deadlock dado que hasta sobra una unidad. Ahora bien, si $n=4$ puede darse el caso que:

PROCESO	UNIDADES
P1	2
P2	2
P3	2
P4	2

Por ende no tenemos problema alguno dado que el próximo proceso que pida una unidad la tendrá y va a finalizar liberando 3 unidades.

Si $n=5$ en el peor de los casos tendremos:

PROCESO	UNIDADES
P1	2
P2	2
P3	2
P4	2
P5	2

Entonces las unidades libres del recurso son 0 (cero), y ninguno de los procesos alcanzó la cantidad de recursos necesaria para finalizar, entonces vemos que sólo será seguro tener hasta 4 procesos corriendo $n=4$.

_____ ○ ○ ○ _____

Ejercicio RM4D: 11

Un profesor califica a sus alumnos con letras de la "A" a la "H", y sólo aprueban los que obtienen "A", "B" o "C". Tiene como costumbre establecer las normas de calificación el primer día de clase y una de ellas es que todos los alumnos que se presentan a rendir el último examen, para poder firmar la materia, ya tienen que haber aprobado todos los exámenes anteriores. Los que desapruében este último examen tienen 3 fechas para recuperarlo en diciembre, pero para motivar a sus alumnos a que obtengan notas altas, fija prioridades en el momento de tomar recuperatorios, con esta última cláusula quiere decir que siempre que exista la posibilidad

de que se presente a recuperar un alumno que haya sido reprobado con "D" o "E" (pertenece al primer nivel dentro de los desaprobados) no le tomará recuperatorio a los que pertenecen al grupo de los desaprobados con "F" o "G", ni a los que pertenecen al grupo de las "H". Con el transcurso del tiempo el profesor ha realizado una estadística obteniendo los siguientes resultados: El alumno que llega a rendir el último examen en diciembre no desaprovecha ninguna de las tres fechas, y los resultados tanto del último examen como de los tres recuperatorios son: el 66,66% aprueba, el 6,6% obtiene "D" y "E", el 6,6% "F" y "G", y el 20% restante obtiene "H". ¿Es posible que al vencer las tres fechas, algún alumno sufra de "estado de inanición"? Justifique.

Sí, es posible siempre que el grupo que se presente a rendir el último examen sea superior a 15 personas, porque si son exactamente 15, 10 aprobarán, 1 pertenecerá al grupo "D-E" que ocupará la primera fecha con una alta probabilidad de aprobar, el grupo "F-G" estará compuesto también por 1 que se presentará en la segunda fecha con una alta probabilidad de aprobar, y en la tercera fecha los 3 restantes que pertenezcan al grupo de las "H" podrán rendir. También existe la posibilidad de que los dos grupos que tiene prioridad más alta estén vacíos. Cómo así también es seguro que el grupo de las "H" termina el año en estado de inanición si es que el grupo original que se presenta a rendir el último examen es de 225.

Ejercicio RM4D: 12

En un sistema que utiliza el algoritmo del banquero existen tres tipos de recursos con las siguientes posibilidades: A=9, B=3 y C=6 (Recursos iniciales). En un instante dado existen cuatro procesos en ejecución y el sistema se encuentra en el siguiente estado:

Proceso	Necesidades máximas			Asignado		
	A	B	C	A	B	C
P1	3	2	2	1	0	0
P2	6	1	3	5	1	1
P3	3	1	4	2	1	1
P4	4	2	1	0	0	1

- Verifique que el sistema se encuentra en un estado seguro.
- Si a partir de este estado, el proceso P1 realiza una petición (A=1,B=0,C=1) además de los que poseía ¿cómo reaccionará el sistema?
- Y luego Suponga que se lo asigna este pedido a P1 ¿Está en Deadlock?
Para ello, debemos buscar una secuencia segura.

A) Los recursos disponibles son A=1; B=1; C=3. Con estas existencias, podemos atender cualquier petición que realicen P2 y P3. Si cualquiera de ellos termina, se puede también atender a P1 en el peor caso. Y con todos estos recursos liberados, puede terminar P4. Por tanto, existen múltiples secuencias seguras para este estado, así que es un estado seguro.

B).Reacción si P1 solicita (A=1,B=0,C=1)

Como se trabaja con el algoritmo del banquero, el sistema puede reaccionar concediendo la petición o denegándola en caso de que se pasara a un estado inseguro. El nuevo estado, si concediéramos la petición, sería:

Disponible: A=0; B=1; C=2.

Proceso	Necesidades máximas			Asignado		
	A	B	C	A	B	C
P1	3	2	2	2	0	1
P2	6	1	3	5	1	1
P3	3	1	4	2	1	1
P4	4	2	1	0	0	1

Busquemos una secuencia segura. En esta situación, ningún proceso puede finalizar en el peor caso, porque todos y cada uno de ellos pueden pedir un recurso de tipo A, del cual no quedan existencias. Por tanto no existe secuencia segura y el estado es inseguro.

Así pues, el sistema tendrá que denegar la petición y dejar al proceso bloqueado.

Ejercicio RM4D: 13

Aplicando el algoritmo del banquero conteste si el siguiente sistema está o no en un estado seguro.

	Asignados				Máximo			
	A	B	C	D	A	B	C	D
P0	1	0	0	3	2	0	1	3
P1	1	0	0	0	1	7	5	0
P2	1	3	5	4	2	3	5	6
P3	0	6	3	2	0	6	5	2
P4	0	0	1	4	0	6	5	6

Disponible			
A	B	C	D
3	1	1	0

Respuesta

Un estado es seguro si el sistema puede asignar recursos a cada proceso (hasta su máximo) en algún orden y aún así evitar los bloqueos mutuos.

Más formalmente, un sistema está en un estado seguro solo si existe una secuencia segura. Una secuencia de procesos $[P_0, P_1, \dots, P_n]$ es una secuencia segura para el estado de asignación actual si, para cada P_i , los recursos que P_i todavía puede solicitar se pueden satisfacer con los recursos que actualmente están disponibles más los recursos que tienen todos los P_j , donde $j < i$. En esta situación, si los recursos que P_i necesita todavía no están disponibles, P_i podrá esperar hasta que todos los P_j hayan terminado. En ese momento, P_i podrá obtener todos los recursos que necesita, llevar a cabo su tarea designada, liberar los recursos que adquirió y terminar.

Vamos a ver si el sistema está en estado seguro. Aplicando el Algoritmo del banquero.

trabajo0 = (3 1 1 0)

fin0 = (False False False False False)

Elijo P0 (Necesidad(0) ≤ trabajo0 : (1 0 1 0) ≤ (3 1 1 0)):

trabajo1 = trabajo0 + Asignados(0) =

(3 1 1 0) + (1 0 0 3) = (4 1 1 3)

fin1 = (True False False False False)

Elijo P2 (Necesidad(2) ≤ trabajo1 : (1 0 0 2) ≤ (4 1 1 3)):

trabajo2 = trabajo1 + Asignados(2) =

(4 1 1 3) + (1 3 5 4) = (5 4 6 7)

fin2 = (True False True False False)

Elijo P3 (Necesidad(3) ≤ trabajo2 : (0 0 2 0) ≤ (5 4 6 7)):

trabajo3 = trabajo2 + Asignados(3) =

(5 4 6 7) + (0 6 3 2) = (5 10 9 9)

fin3 = (True False True True False)

Elijo P4: (Necesidad(4) ≤ trabajo3 : (0 6 4 2) ≤ (5 10 9 9)):

trabajo4 = trabajo3 + Asignados(4) =

(5 10 9 9) + (0 0 1 4) = (5 10 10 13)

fin4 = (True False True True True)

Elijo P1 (Necesidad(1) ≤ trabajo4 : (0 7 5 0) ≤ (5 10 10 13)):

trabajo5 = trabajo4 + Asignados(1) =

(5 10 10 13) + (1 0 0 0) = (6 10 10 13)

fin5 = (True True True True True)

	Necesidades (Máximo – Asignados)			
P0	1	0	1	0
P1	0	7	5	0
P2	1	0	0	2
P3	0	0	2	0
P4	0	6	4	2

Verificación:

Asignadosini + Disponiblesini = Trabajofin

3 9 9 13) + (3 1 1 0) = (6 10 10 13)

Entonces existe una secuencia $[P_0, P_3, P_1, P_4, P_2]$ que es segura.

Entonces es estado seguro y por lo tanto no tiene que haber deadlock.

Ejercicio RM4D: 14

Considere la siguiente situación del sistema

	Asignado	Máximo	Disponible
--	----------	--------	------------

P0	0	0	1	2	0	0	1	2
P1	1	0	0	0	1	7	5	0
P2	1	3	5	4	2	3	5	6
P3	0	6	3	2	0	6	5	2
P4	0	0	1	4	0	6	5	6

1	5	2	0
---	---	---	---

- a) Escriba la tabla de necesidad.

Respuesta

a)	Necesidades
P0	0 0 0 0
P1	0 7 5 0
P2	1 0 0 2
P3	0 0 2 0
P4	0 6 4 2

Ejercicio RM4D: 15

Considérese un sistema con 10 recursos del mismo tipo en el que se está ejecutando el algoritmo del banquero. Su estado, en un instante dado, es el representado por la siguiente tabla:

Proceso	Recursos asignados	Recursos máximos
A	1	2
B	1	5
C	4	7
D	2	6

- a) Analizar el estado actual, indicando si es seguro o inseguro.
 b) Si el proceso B solicita un recurso, indicar si se le debe conceder o no la demanda. Justificar.

Respuesta

a) El estado es seguro. Basta con aplicar el algoritmo del banquero para la planificación A-C-B-D o A-C-D-B para comprobar que todos los procesos pueden terminar su ejecución.

b) Para comprobar si se le debe conceder o no la demanda, tenemos que suponer que se le concede y evaluar si el estado resultante es seguro o inseguro. En este caso, el estado es inseguro puesto que sólo podríamos terminar el proceso A, quedando los 3 restantes con una necesidad de recursos superior a los disponibles. Conclusión: No se le debe conceder la demanda de un recurso al proceso B.

Ejercicio RM4D: 16

Teniendo a los procesos P1, P2, P3, y P4 ejecutados en el siguiente orden:

P2, P1, P4, P3

, los recursos R1, R2 y R3 y dadas las siguientes matrices, demostrar que el sistema se encuentra en un estado seguro:

Asignados			
	R1	R2	R3
P1	0	2	0
P2	4	0	4
P3	3	2	2
P4	0	0	3

Solicitudes		
R1	R2	R3
0	0	0
0	0	0
2	0	0
0	0	3

siguientes matrices, demostrar que el

Disponibles		
0	0	0

Solución

La cantidad de recursos totales del sistema son R1=7; R2=4; R3=9

Ejecuta P2 y termina:

Asignados			
	R1	R2	R3
P1	0	2	0
P2	0	0	0
P3	3	2	2
P4	0	0	3

Solicitudes		
R1	R2	R3
0	0	0
0	0	0
2	0	0
0	0	3

Disponibles		
0	0	0
4	0	4

Ejecuta P1 y termina:

Disponibles		
-------------	--	--

0	0	0
4	0	4

Asignados			
	R1	R2	R3
P1	0	0	0
P2	0	0	0
P3	3	2	2
P4	0	0	3

Solicitudes		
R1	R2	R3
0	0	0
0	0	0
2	0	0
0	0	3

4	2	4
---	---	---

Solicita recursos P4, se le asignan, ejecuta y termina:

Asignados			
	R1	R2	R3
P1	0	0	0
P2	0	0	0
P3	3	2	2
P4	0	0	0

Solicitudes		
R1	R2	R3
0	0	0
0	0	0
2	0	0
0	0	0

Disponibles		
0	0	0
4	0	4
4	2	4
4	2	1
4	2	7

Solicita recursos P3, se le asignan, ejecuta y termina:

Asignados			
	R1	R2	R3
P1	0	0	0
P2	0	0	0
P3	0	0	0
P4	0	0	0

Solicitudes		
R1	R2	R3
0	0	0
0	0	0
0	0	0
0	0	0

Disponibles		
0	0	0
4	0	4
4	2	4
4	2	1
4	2	7
2	2	7
7	4	9

Como vemos los recursos disponibles al final de la ejecución de los 4 procesos (se ejecutaron sin ningún problema, pudiéndose servir todas sus demandas de recursos) es igual a la cantidad de recursos totales, el estado del sistema es *seguro*.

Ejercicio RM4D: 17

Teniendo a los procesos P0, P1, P2, P3, y P4 ejecutados en el siguiente orden:

P0, P3, P1, P4, los recursos R1 y R2 y dadas las siguientes matrices, ¿el sistema se encuentra en un estado seguro?:

Asignados		
	R1	R2
P0	2	0
P1	1	1
P2	0	4
P3	1	3
P4	1	0

Solicitudes	
R1	R2
0	0
4	0
1	1
0	1
0	2

Disponibles	
0	0

Los recursos totales del sistema son: R1=5; R2=8

Ejecuta P0 y termina

Asignados		
	R1	R2
P0	0	0
P1	1	1
P2	0	4
P3	1	3
P4	1	0

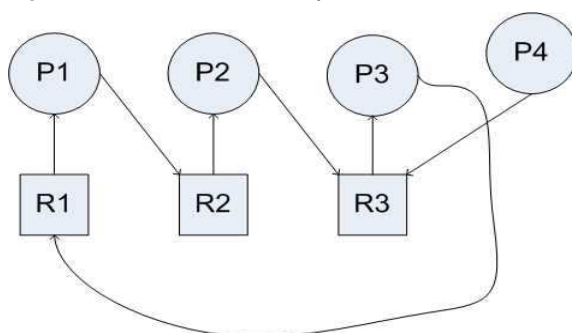
Solicitudes	
R1	R2
0	0
4	0
1	1
0	1
0	2

Disponibles	
0	0
2	0

Solicita recursos P3, no se le pueden asignar, todos los procesos (excepto P0, claro) quedan en deadlock, el estado del sistema es *inseguro*.

Ejercicio RM4D: 18

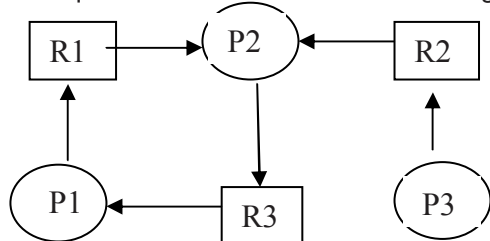
En el siguiente grafo de asignación de recursos la ejecución es: P1, P2, P3, P4, y por tanto, se encuentra una situación de deadlock. ¿Qué se debe modificar para salvar esta situación?

**Solución**

Una alternativa es quitarle R3 a P3 y asignárselo a P4 luego cambiar el orden de ejecución: P4, P2, P1, P3, quedando entonces un estado seguro.

Ejercicio RM4D: 19

Justifique si existe deadlock o no en el siguiente grafo de asignación de recursos.



Existe deadlock porque P2 está bloqueado esperando a R2 que está asignado a P1, que está bloqueado esperando que se le asigne R3, que está asignado a P3 que está esperando a R1 que está asignado a P1. Existe una espera circular entre P1 y P3, haciendo que cualquier proceso que requiera los recursos R1 y/o R3 también pasen a formar parte del deadlock.

Ejercicio RM4D: 20

Dada la siguiente secuencia ejecución: ZZYXY, los valores iniciales de los semáforos contadores: S = 0; R = 1; B = 1 y la secuencia lógica:

X	Y	Z
P(B)	P(S)	P(R)
.....	P(B)
.....
V(S)	V(B)	V(S)
	V(R)	

Se piden los valores finales de los semáforos en cuestión, y cómo se obtienen dichos valores.

Proceso	Secuencia	S	R	B	Estado
		0	1	1	
Z	P(R)		0		Activo
	P(B)			0	Activo
	V(S)	1			Fin
Z	P(R)		-1		En Cola
Y	P(S)	0			Activo
	V(B)			1	Fin
	V(R)		0		Fin
Z	P(B)			0	Activo
	V(S)	1			Fin
X	P(B)			-1	En Cola
Y	P(S)	0			Activo
	V(B)			0	Fin
	V(R)		1		Fin
X	V(S)	1			Fin
		1	1	0	

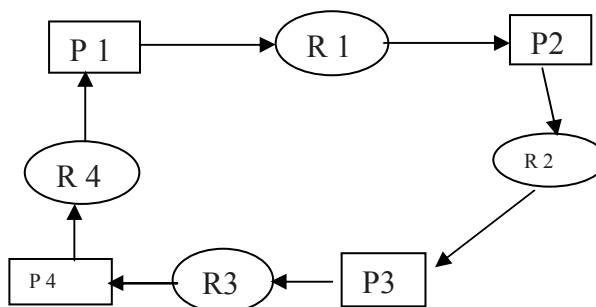
Estados Finales de los semáforos: S = 1, R = 1 y B = 0

Ejercicio RM4D: 21

Punto 2a- Si tengo un sistema de paginación por demanda y 4 procesos que solicitaran recursos según de la siguiente forma:

R1 Archivo1 para escritura
 R2 Impresora
 R3 Frame de memoria
 R4 Unidad de Cinta

Determinar si hay Deadlock y si hubiera explicar como recuperar este Deadlock

**Resolución**

aquí no hay deadlock dado que el recurso R3 no cumple con la condición de no expropiación. Dado que su estado puede ser salvado en disco si fue modificado (paginación por demanda)

Ejercicio RM4D: 22

Considere un sistema con 2 tipos de recursos y 2 posibilidades para cada tipo. En este sistema podrán existir como máximo 3 procesos, los cuales piden y liberan recursos mediante las funciones Pido_Recurso(int tipo) y Libero_Recurso(int tipo), respectivamente. (Observe que mediante estas funciones sólo se puede solicitar y liberar una posibilidad del tipo especificado y que se cumple el principio de independencia de dispositivo).

Implemente en lenguaje C una función de detección de interbloqueo (DEADLOCK) para el sistema descrito. En su implementación emplee las siguientes declaraciones:

```

/* DEFINICIÓN DE LITERALES */
#define TIP_RECUR 2 /* Tipos de recursos */
#define NUM_PROC 3 /* número máximo de procesos */
#define FALSO 0
#define VERDADERO 1
#define HAY_INTERBLOQ 0
#define NO_HAY_INTERBLOQ 0
/* DECLARACIÓN DE VARIABLES GLOBALES */
/* Disponibilidades para cada tipo */
int Disponible[TIP_RECUR] = {2, 2};
/* Lo que cada proceso tiene asignado */
int Asignación[NUM_PROC][NUM_REC] = {0, 0, 0, 0, 0, 0};
/* Recurso por el que espera cada proceso */
/* -1 significa que no espera */
int Solicita[NUM_PROC] = {-1,-1,-1};
/* Prototipo de la función de detección */
/* Que deberá retornar HAY_INTERBLOQ o NO_HAY_INTERBLOQ */
int deteccion();
  
```

Una posible solución es: Función de detección de interbloqueo

```

/* Vector auxiliar utilizado en la función de detección */
int Trabajo[NUM_RECU];

/* Vector auxiliar utilizado en la función de detección */
int Fin[NUM_PROC];

int deteccion( )
/* Retona HAY_INTERBLOQ si existe un estado de interbloqueo */
/* Retorna NO_HAY_INTERBLOQ si no existe interbloqueo */
{
  int i, j;
  
```



```

/* Inicializamos vectores auxiliares */
for (i=0; i<NUM_REC; i++) {
    Trabajo[i] = Disponible[i];

for (i=0; i<NUM_PROC; i++) {
    Fin[i] = VERDADERO;
    for (j=0; j<NUM_REC; j++)
        if ( Asignado[i][j] ) {
            FIN[i] = FALSO;
            break;
        }
    }
}
/* Búsqueda de ciclos que impliquen interbloqueo */
nuevo = FALSO;
do {
    for (i=0; i<NUM_PROC; i++) {
        if ( Fin[i] == FALSO ) {
            if ( Solicita[i] != -1 )
                if ( !Trabajo[Solicita[i]] ) continue;
            FIN[i] = VERDADERO;
            for (j=0; j<NUM_REC; j++)
                Trabajo[j] += Asignado[i][j];
            nuevo=VERDADERO;
            break;
        }
    }
} While ( nuevo == VERDADERO );

/* Preguntamos si todos los procesos están marcados; si no entonces interbloqueo */
for (i=0; i<NUM_PROC; i++)
    if ( Fin[i] == FALSO ) return( HAY_INTERBLOQUEO);
return(NO_HAY_INTERBLOQUEO);

} /* Final de la función */

```

_____ ○ ○ ○ _____

Ejercicio RM4D: 23

Un hotel está compuesto por la siguiente infraestructura: 5 habitaciones en suite (S), 30 habitaciones matrimoniales con vista panorámica (MVP), 50 habitaciones matrimoniales comunes (MC), 20 habitaciones base doble con vista panorámica (BDVP), 45 habitaciones base doble comunes (BDC) y 20 habitaciones base triple (BT). La agencia de turismo “A” le confirma al administrador del hotel que a partir del domingo 13/07, todos los domingos hasta el 3/08 se renovará un tour que requerirá 10 MVP, 20 MC, 10 BDVP y 30 BDC. La agencia “B” le confirma un tour que arribará al hotel el martes 15/07, que se renovará por única vez el 29/07 y cuyos requerimientos son 3 S, 15 MVP, 20 MC, 10 BDVP, 10 BDC y 15 BT. Y finalmente la agencia “C” le confirma un tour que se hará todos los viernes a partir del 11/07 y que tendrá una duración de 4 días (hasta el lunes inclusive) y cuyos requerimientos son: 2 S, 10 MVP y 8 BDVP. El administrador le ha confirmado a las tres agencias su servicio. ¿ Se le presentará una situación de deadlock? ¿Por qué?

RESOLUCIÓN

Se produce Starvation, porque para el segundo tour de la agencia “C” no tendrá disponible 10 MVP, ni 6 BDVP, ya que estarán ocupadas por los pasajeros de la agencia “B”.

_____ ○ ○ ○ _____

EJERCICIOS SIN RESOLVER DE DEADLOCKS

Ejercicio M4D: 1

Dado un sistema con tres impresoras, tres CD-Rom y dos cintas disponibles, y las siguientes matrices:

	Máximo		
	Impresoras	CD	Cintas
P0	8	5	3
P1	3	4	2
P2	9	3	2
P3	2	2	1
P4	4	4	2

	Asignación		
	Impresoras	CD	Cintas
P0	1	1	0
P1	2	2	0
P2	3	3	2
P3	2	1	0
P4	0	1	1

Se pide:

- Indique cuál es el estado actual del sistema y justifique.
- Si P4 solicita tres impresoras y 3 CD's ¿ puede ser asignado ?
- Si P0 solicita 2 CD's ¿ pueden ser asignados ?

○ ○ ○ _____

Ejercicio M4D: 2

En un sistema de cómputos que cuenta con el sistema operativo bark, se encuentran ejecutando cinco procesos que utilizan recursos críticos en distintas etapas de su vida. El sistema operativo utiliza una estrategia múltiple para evitar los abrazos mortales (deadlock). La primera de las estrategias es la de realizar Checkpoint. Entre Checkpoint y Checkpoint utiliza el algoritmo del Avestruz.

En un determinado momento el sistema se encuentra con la siguiente asignación de recursos:

- El proceso A tiene capturada la cinta, y está esperando por el W-CD.
- El proceso B tiene capturado el semáforo contador de lectura del registro 1 de la base de datos, y está esperando por la impresora.
- El proceso C tiene capturado el W-CD, y está esperando por la impresora.
- El proceso D tiene capturada la impresora y espera la cinta.
- El proceso E tiene capturado el semáforo contador de lectura del registro 1 de la base de datos, y está esperando por la cinta.

En último Checkpoint realizado los procesos se encontraban de la siguiente manera:

- El proceso A tiene capturada la cinta, y está esperando por el W-CD.
- El proceso B termina de capturar el semáforo contador de lectura del registro 1 de la base de datos
- El proceso C tiene capturado el W-CD
- El proceso D tiene capturada la impresora y espera la cinta.
- El proceso E tiene capturado el semáforo contador de lectura del registro 1 de la base de datos, y está esperando por la cinta.

Se pide (todas las respuestas deben estar justificadas para que sean tenidas en cuenta, si la justificación es incorrecta no importa el valor asignado a la misma):

- Efectuar el gráfico de asignación de procesos y recursos (indique claramente que significa cada flecha – posee / necesita).
- Justifique cual es el mejor camino a tomar a partir de este momento para resolver el problema que se presenta en el sistema, y por el cual los procesos no terminan. Las posibles soluciones son realizar un Rollback, o eliminar uno o más procesos (se debe indicar cual/es, y justificar). Para tomar una decisión debe tener en cuenta el costo de reprocesamiento. Dentro de la justificación deberá indicar los parámetros que tomó en cuenta para la decisión final.

_____ ○ ○ ○ _____

Ejercicio M4D: 3

Dado un sistema que se encuentra ejecutando el algoritmo del banquero, y que cuenta con tres impresoras, cinco CD-Rom's y dos cintas disponibles y las siguientes matrices:

Máximo				Asignado			
Pr.	Impresoras	CD-Roms	Cintas	Pr.	Impresoras	CD-Roms	Cintas
A	0	0	1	A	0	0	1
B	1	7	5	B	1	0	0
C	2	3	5	C	1	3	5
D	0	6	5	D	0	6	3
E	0	6	5	E	0	0	1

Se pide (En todos los puntos se debe justificar la respuesta a través del algoritmo de seguridad.) :

- Indique cuál es el estado actual del sistema y justifique.
- Si B solicita (0, 4, 2) ¿pueden ser asignado?
- Si C solicita (0, 1, 0) ¿pueden ser asignados?
- Si E solicita (0, 6, 0) ¿pueden ser asignados?

Ejercicio M4D: 4

En un sistema de computación real, ni los recursos disponibles ni los requerimientos de los procesos por recursos se mantienen por largo tiempo. Los recursos se rompen o son reemplazados, los procesos vienen y van, se agregan nuevos recursos al sistema, etc.

Si en tal sistema, los bloqueos se controlan con el algoritmo del banquero, ¿cuáles de los siguientes cambios se puede hacer con seguridad (sin introducir posibilidad de bloqueos) y bajo qué circunstancias?:

- Incrementar AVAILABLE (se agregan nuevos recursos).
- Decrementar AVAILABLE (se eliminan recursos del sistema).
- Incrementar MAX para un proceso (el proceso necesita más recursos que los permitidos).
- Decrementar MAX para un proceso (el proceso decide que no va a necesitar algunos recursos).
- Incrementar el número de procesos.
- Decrementar el número de procesos.

Hallar la matriz de necesidad. Tener en cuenta que el sistema usa el “Algoritmo del Banquero” para bloquear a los procesos que piden más recursos que los disponibles. Solo se asignara a un proceso los recursos que necesite, si estos son todos los que va a necesitar para completar la ejecución.

El número de los procesos corresponde con el orden de llegada. Se deberá utilizar el algoritmo de planificación FCFS.

Además indicar si el sistema es seguro o no (o sea verificar si ningún proceso queda sin los recursos necesarios).

	Peticiónes Máximas			
	R1	R2	R3	R4
P1	2	3	2	5
P2	1	2	6	3
P3	0	2	4	5
P4	3	0	5	2
P5	3	4	5	4

Disponibles			
R1	R2	R3	R4
1	0	9	4

	Recursos Asignados			
	R1	R2	R3	R4
P1	0	3	1	3
P2	1	1	3	2
P3	0	2	1	0
P4	2	0	2	0
P5	1	3	5	2

Ejercicio M4D: 5

Hallar la matriz de necesidad. ¿Se puede satisfacer P4(0,6,0,1)?

Tener en cuenta que el sistema usa el “**algoritmo del banquero**” para bloquear a los procesos que piden más recursos que los disponibles.

Solo se asignara a un proceso los recursos que necesite, si estos son todos los que va a necesitar para completar la ejecución.

El número de los procesos corresponde con el orden de llegada.

Peticiones Máximas				
	R1	R2	R3	R4
P1	5	0	2	4
P2	2	3	5	0
P3	4	3	5	6
P4	0	8	0	3
P5	1	3	2	4
P6	4	5	5	7
P7	1	2	0	6

Recursos Asignados				
	R1	R2	R3	R4
P1	2	0	0	2
P2	1	3	5	0
P3	3	2	2	2
P4	0	0	0	2
P5	0	0	2	4
P6	2	1	3	0
P7	1	1	0	5

Disponibles			
R1	R2	R3	R4
0	7	1	2

Ejercicio M4D: 6

Hallar la matriz de necesidad e indique si el sistema esta o no seguro. Tener en cuenta que el sistema usa el “algoritmo del banquero” para bloquear a los procesos que piden más recursos que los disponibles. Solo se asignara a un proceso los recursos que necesite, si estos son todos los que va a necesitar para completar la ejecución. El algoritmo de planificación de la cola de listos y la de bloqueados es FCFS (First Come First Served).

Tener en cuenta que el número de los procesos corresponde con el orden de llegada.

Indicar el estado del sistema al realizar la petición.

No olvidar que la CPU es apropiativa.

Matriz de asignación					Matriz de recursos máximo				
Proceso					Proceso				
A	6	2	7	8	A	6	3	8	9
B	8	5	2	3	B	8	6	3	3
C	2	1	7	8	C	2	2	7	9
D	0	1	0	0	D	0	2	0	0
E	5	4	5	4	E	5	4	6	5
F	2	8	0	1	F	2	8	1	1
G	5	0	1	3	G	5	0	1	4

Disponible (0 0 0 1)

NOTA : Todos los recursos son del mismo tipo.

b)

Matriz de asignación					Matriz de recursos máximo				
Proceso					Proceso				
A	0	1	1	2	A	0	1	1	3
B	2	0	1	0	B	2	2	2	4
C	1	0	0	1	C	4	0	0	2
D	1	2	3	0	D	1	9	3	9
E	2	4	1	1	E	3	4	1	1
F	2	0	0	2	F	2	0	0	2

Disponible (0 1 0 0)

c)

Matriz de asignación					Matriz de recursos máximo				
Proceso					Proceso				
P0	0	0	0	1	P0	0	0	1	2
P1	0	0	1	1	P1	2	1	2	2
P2	0	1	0	1	P2	3	1	0	1
P3	0	1	1	1	P3	0	1	1	2
P4	1	0	0	1	P4	1	3	3	1

Disponible (0 0 0 1)

d)

a)

Matriz de asignación						Matriz de recursos máximo					
Proceso						Proceso					
P0	0	1	3	0	2	P0	1	1	3	6	3
P1	0	2	2	0	0	P1	0	4	5	3	3
P2	1	2	3	0	0	P2	2	2	3	0	0
P3	1	2	3	0	0	P3	1	2	5	1	1
P4	3	0	0	3	1	P4	4	2	2	3	1

Disponible (1 0 0 0 0)

NOTA : Todos los recursos son del mismo tipo.

	Necesidades (Maximo – Asignados)
P0	1 0 1 0
P1	0 7 5 0
P2	1 0 0 2
P3	0 0 2 0
P4	0 6 4 2

Ejercicio M4D: 7

Dado un sistema con tres impresoras, tres CD-Rom y dos cintas disponibles, y las siguientes matrices:

	Máximo		
	Impresoras	CD	Cintas
P0	8	5	3
P1	3	4	2
P2	9	3	2
P3	2	2	1
P4	4	4	2

	Asignación		
	Impresoras	CD	Cintas
P0	1	1	0
P1	2	2	0
P2	3	3	2
P3	2	1	0
P4	0	1	1

Se pide:

- h. Indique cuál es el estado actual del sistema y justifique.
- i. Si P4 solicita tres impresoras y 3 CD's ¿ puede ser asignado ?
- j. Si P0 solicita 2 CD's ¿ pueden ser asignados ?

Ejercicio M4D: 8

En un sistema de cómputos que cuenta con el sistema operativo bark, se encuentran ejecutando cinco procesos que utilizan recursos críticos en distintas etapas de su vida. El sistema operativo utiliza una estrategia múltiple para evitar los abrazos mortales (deadlock). La primera de las estrategias es la de realizar Checkpoint. Entre Checkpoint y Checkpoint utiliza el algoritmo del Avestruz.

En un determinado momento el sistema se encuentra con la siguiente asignación de recursos:

- El proceso A tiene capturada la cinta, y está esperando por el W-CD.
- El proceso B tiene capturado el semáforo contador de lectura del registro 1 de la base de datos, y está esperando por la impresora.
- El proceso C tiene capturado el W-CD, y está esperando por la impresora.
- El proceso D tiene capturada la impresora y espera la cinta.
- El proceso E tiene capturado el semáforo contador de lectura del registro 1 de la base de datos, y está esperando por la cinta.

En último Checkpoint realizado los procesos se encontraban de la siguiente manera:

- El proceso A tiene capturada la cinta, y está esperando por el W-CD.
- El proceso B termina de capturar el semáforo contador de lectura del registro 1 de la base de datos
- El proceso C tiene capturado el W-CD
- El proceso D tiene capturada la impresora y espera la cinta.
- El proceso E tiene capturado el semáforo contador de lectura del registro 1 de la base de datos, y está esperando por la cinta.

Se pide (todas las respuestas deben estar justificadas para que sean tenidas en cuenta, si la justificación es incorrecta no importa el valor asignado a la misma):

- k. Efectuar el gráfico de asignación de procesos y recursos (indique claramente que significa cada flecha – posee / necesita).
- l. Justifique cual es el mejor camino a tomar a partir de este momento para resolver el problema que se presenta en el sistema, y por el cual los procesos no terminan. Las posibles soluciones son realizar un Rollback, o eliminar uno o más procesos (se debe indicar cual/es, y justificar). Para tomar una decisión debe tener en cuenta el costo de reprocesamiento. Dentro de la justificación deberá indicar los parámetros que tomó en cuenta para la decisión final.

Ejercicio M4D: 9

Dado un sistema que se encuentra ejecutando el algoritmo del banquero, y que cuenta con tres impresoras, cinco CD-Rom's y dos cintas disponibles y las siguientes matrices:

Máximo				Asignado			
Pr.	Impresoras	CD-Roms	Cintas	Pr.	Impresoras	CD-Roms	Cintas
A	0	0	1	A	0	0	1
B	1	7	5	B	1	0	0
C	2	3	5	C	1	3	5
D	0	6	5	D	0	6	3
E	0	6	5	E	0	0	1

Se pide (En todos los puntos se debe justificar la respuesta a través del algoritmo de seguridad.):

- m. Indique cuál es el estado actual del sistema y justifique.
- n. Si B solicita (0, 4, 2) ¿pueden ser asignado?
- o. Si C solicita (0, 1, 0) ¿pueden ser asignados?
- p. Si E solicita (0, 6, 0) ¿pueden ser asignados?

Ejercicio M4D: 10

Dadas las siguientes matrices de procesos y recursos, contestar si es un estado seguro

Recursos disponibles: A(0), B(2), C(1), D(0)

	Max. Pedido de Recursos				Recursos asignados			
	A	B	C	D	A	B	C	D
P0	1	1	1	1	0	1	1	0
P1	2	3	3	2	2	1	0	0
P2	2	4	4	0	0	1	0	0
P3	0	0	0	3	0	0	3	3

Ejercicio M4D: 11

Una aerolínea brinda servicios de vuelo a un determinado destino a razón de un 1 vuelo diario. Sus aviones tienen una capacidad de 10 asientos en primera categoría (P), 20 en Business (B) y 50 en clase Turista (T). Si la agencia A pidió reservas para todos los días del mes de mayo de 5 asientos en Primera categoría, 10 en categoría Business, 10 en Turista. La agencia B pidió reservas para todos los sábados y domingos de mayo de 5 asientos en Primera, 8 en Business y 40 en Turista. Y la agencia C pidió reservas para el ultimo sábado del mes de mayo de 5 asientos clase turista. Si la aerolínea acepto todas las reservas en que estado (Seguro o Inseguro) quedó?. Por que?

Ejercicio M4D: 12

Suponga los siguientes 2 procesos que son ejecutados concurrentemente en un sistema operativo preemptive y comparten los contadores generales de los semáforos A y B y la variable CONT; inicializados de la siguiente manera: A=B=1; CONT=0;

Responder:

- ¿Puede llegar a ocurrir deadlock? Justificar la respuesta
- ¿Puede llegar a ocurrir starvation? Justificar la respuesta
- ¿Considera que los procesos se encuentran correctamente sincronizados?

PROCESO 1

Repeat

```
Wait(A);
Writeln("A");
If(CONT>0) then
writeln("jejeje");
Wait(B);
CONT:=CONT+1;
Signal(A);
Signal(B);
```

Forever

PROCESO 2

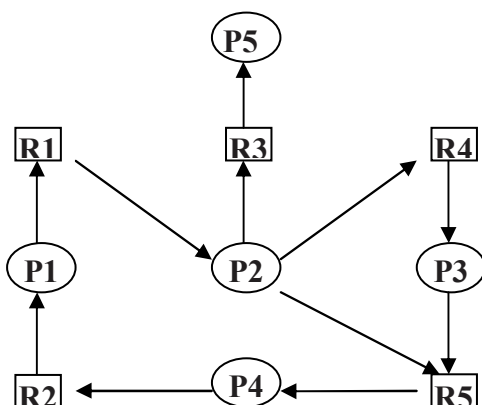
Repeat

```
While(CONT!=0)
{
Wait(B);
Writeln("B");
Wait(A);
CONT:=CONT-1;
Writeln(CONT);
Signal(A);
Signal(B);
```

}
Forever

Ejercicio M4D: 13

Deadlock: dado el siguiente grafo de asignación de recursos:



Cuáles son los procesos y recursos en deadlock?:

Con qué estrategia se puede resolver el deadlock?:

○ ○ ○ _____

Ejercicio M4D: 14: Final UTN 04/04/06

B2. Mientras 60.000 personas cantaban sin parar, Mick se cambiaba en el backstage del escenario y Keith ya comenzaba a entonar las primeras estrofas de "This Place Is Empty". El público deliraba. Los plomos se estaban preparando para la próxima canción, la que llevaría a Mick, a Keith, a Ronnie y a Charlie al centro del campo. La lluvia no paraba de caer. El sistema que controlaba el escenario móvil constaba de 8 dispositivos de desplazamiento (DD), 7 controladores de posición (CP) y 7 servomecanismos para elevar el escenario (EE). Keith seguía cantando bajo la lluvia, mientras Mick consultaba con el equipo sobre la situación. Estaba cayendo mucha lluvia y había dudas respecto a la posibilidad de mover el escenario. El Sistema estaba ejecutando los siguientes procesos:

Proceso	Demanda Máxima declarada			Recursos Alocados		
	DD	CP	EE	DD	CP	EE
Sonido	4	3	6	1	1	0
Movimiento	0	4	4	0	2	1
Luces	4	2	2	1	1	1
Frenos	1	6	3	0	0	2
Video	7	3	2	2	1	0

Mick quería que el equipo de backstage le garantizara que el escenario no se iba a trabar en el medio del camino por culpa de un deadlock, dado que dejaría al grupo en una mala posición.

B2.1) Indicarle a Mick si se podrá mover el escenario (Estado seguro) o si no se podrá (Estado inseguro). Detalle en la respuesta los pasos para determinar en que estado se encuentra el escenario.

B2.2) Describir las estrategias de recuperación de deadlock (Deadlock recovery).

○ ○ ○ _____

Ejercicio M4D: 15:

En una determinada instalación del sistema operativo "Doors", que se encuentra operativo y con carga de trabajo, en una implementación non-stop (7x24), se instala un Service Pack, que como mejora importante trae una actualización al algoritmo utilizado por el sistema operativo para trabajar con los deadlocks. En la versión sin el parche el método utilizado era el algoritmo del Avestruz, en cambio en ésta nueva actualización se puede elegir entre seguir usando dicho algoritmo, o utilizar una nueva facilidad denominada AB14.2, que se basa en la teoría del algoritmo del banquero.

El System Administrator, luego de leer la breve documentación sobre ésta nueva facilidad, decide ponerlo en marcha inmediatamente, ya que estaba cansado de tener que conectarse a cualquier hora, y desde cualquier lado para eliminar procesos que habían entrado en abrazo mortal. El Administrador tiene muy en claro que el problema se centra en la escasez de recursos críticos con los que cuenta en el centro de cómputos (solo uno de cada uno de los tipos más usados), versus la gran utilización de los mismos, y la gran cantidad de procesos concurrentes existentes en el sistema, pero para su desgracia no le aprueban los presupuestos para comprar mayor cantidad de recursos, por lo que decide sin perder mucho tiempo instalar directamente en producción la nueva facilidad.

Una vez instalado el paquete de software, se intenta iniciar manualmente el servicio AB14.2 (según los parámetros indicados en la documentación), pero el resultado es el código de error **-41875b**, que como suele suceder en la mayoría de los casos es el famoso error no documentado. Luego de buscar en toda la documentación que le fue suministrada con el paquete, y de buscar en todas las FAQ's existentes en la página de MiniSoft, decidió enviarle un mail a Ud. (jefe de soporte técnico del sistema operativo Doors en la empresa MiniSoft), indicándole el problema.

Se pide que Ud. analice el problema y de su respuesta, indicando cual es el inconveniente, y como debe actuar el Administrador en consecuencia.

Observación: Los posibles caminos a tomar pueden ser básicamente dos distintos, o declarar un bug en el sistema, o enviarle al System Administrator el procedimiento correcto. Toda la información que Ud. necesite consultar con el Administrador será por mail, y el contenido de los mismos debe ser parte de su respuesta. Cualquier decisión que Ud. tome debe estar debidamente justificada. Dentro de su respuesta debe estar explicado el funcionamiento de cada uno de los algoritmos descriptos en el enunciado.



Ejercicio M4D: 16: OLIMPIADAS:

Estudiemos la siguiente situación en una competencia deportiva entre varios equipos.

Los equipos A y B compiten en tres disciplinas: patinaje, atletismo en 100 m llanos y atletismo en 400 m con obstáculos. Antes del día de las pruebas, ambos equipos solicitan se les asignen los circuitos para el entrenamiento de sus atletas.

El equipo A tiene asignada la pista de 400 m (con obstáculos), mientras que el equipo B tiene la pista recta de 100 m y la pista de patinaje.

Aunque la pista de 400 m sirve para los corredores de 100 m llanos, no sucede así con los patinadores, así que el entrenador del equipo A solicita la asignación de la pista de patinaje. Algo similar sucede al equipo B, ya que necesita usar la pista de 400 m con obstáculos.

Ninguno de los equipos tiene la intención de liberar las pistas que ya tiene asignadas, y tampoco compartirlas, aunque no las utilicen las 24 horas del día.

Se pregunta:

- ¿Hay abrazo mortal? ¿Hay inanición? Justifique brevemente su respuesta.
 - Si alguna de las respuestas de a) fue afirmativa, ¿es posible crear algún tipo de administración eficiente de los recursos que evite el tipo de inconveniente del punto a)? ¿Cuál sería?
- Usted puede crear procesos para la solución, pero no puede duplicar recursos, porque las pistas son muy costosas y hay solo una instancia de cada una.



Ejercicio M4D: 17: ALADINO

En su lucha por el poder en el califato de Bagdad, Aladino y el Emir se encuentran en los momentos culminantes. El Emir ha secuestrado a la Princesa y la ha encerrado en la torre más alta de su castillo.

Aladino envió a su fiel Genio a rescatarla, y lo único que consiguió fue que el malvado Emir lo drogara y lo encerrara también (aunque en una torre separada).

Ahora, Aladino se encuentra vagando por las calles de Bagdad, sin dinero, como siempre, y con la lámpara vacía mientras que el Emir mantiene prisioneros a la Princesa y al Genio.

Pero nuestro héroe (Aladino), tiene una idea genial: para conseguir liberar a la Princesa antes que el malvado Emir logre casarse con ella (y así obtener el califato del reino), debe liberar primero al Genio, para conseguir un poco de ayuda extra. Para esto, roba una llave que le permite ingresar a la torre donde el Genio está prisionero, (recuerde Ud. que Aladino era un ladrón), y de paso, algo de dinero (lo suficiente) como para comprar el antídoto a la droga que le fuera administrada al Genio.

El Emir, mientras tanto, está muy preocupado porque el efecto de la droga se pase antes de que él pueda conseguir la lámpara (que Aladino mantiene muy bien oculta), y encerrarlo para toda la eternidad.

Suponga que ahora Ud. se convierte en el guionista de esta historia. Ud. debe resolver las siguientes cuestiones, para hacer que la película se convierta en un éxito de taquilla. Como Ud. no ha estudiado cine **y sí Sistemas Operativos**, se le sugiere hacer una analogía con las conductas de procesos y recursos y los esquemas clásicos de abrazo mortal e inanición y así poder contestar a lo siguiente:

- ¿Conseguirá Aladino liberar a la Princesa? ¿Y a su Genio?
- ¿Hay abrazo mortal?
- ¿Hay inanición?

Justifique TODAS SUS RESPUESTAS. En el caso de las dos últimas, dibuje el grafo de asignación de recursos, indicando cuáles son los recursos y cuáles los procesos en las dos etapas: 1) antes que Aladino obtuviera la llave y el antídoto y 2) luego que liberara y restableciera la salud al Genio (obviamente que para esto quería el jovencito la llave y el antídoto).

Ejercicio M4D: 18

Dado un sistema con tres impresoras, tres CD-Rom y dos cintas disponibles, y las siguientes matrices:

Máximo			
	Impresoras	CD	Cintas
P0	7	5	3
P1	3	2	2
P2	9	0	2
P3	2	2	2
P4	4	3	3

Asignación			
	Impresoras	CD	Cintas
P0	0	1	0
P1	2	0	0
P2	3	0	2
P3	2	1	1
P4	0	0	2

Se pide:

- Indique cuál es el estado actual del sistema y justifique.
 - Si P4 solicita tres impresoras y 3 CD's ¿ puede ser asignado ?
 - Si P0 solicita 2 CD's ¿ pueden ser asignados ?
- Los puntos b y c deben estar justificados a través del algoritmo de seguridad.

Ejercicio M4D: 19

Dado un sistema que se encuentra ejecutando el algoritmo del banquero, y que cuenta con tres impresoras, cinco CD-Rom's y dos cintas disponibles y las siguientes matrices:

Máximo			
Pr.	Impresoras	CD-Roms	Cintas
A	0	0	1
B	1	7	5
C	2	3	5
D	0	6	5
E	0	6	5

Asignado			
Pr.	Impresoras	CD-Roms	Cintas
A	0	0	1
B	1	0	0
C	1	3	5
D	0	6	3
E	0	0	1

Se pide (En todos los puntos se debe justificar la respuesta a través del algoritmo de seguridad.):

- Indique cuál es el estado actual del sistema y justifique.
- Si B solicita (0, 4, 2) ¿ pueden ser asignado ?
- Si C solicita (0, 1, 0) ¿ pueden ser asignados ?
- Si E solicita (0, 6, 0) ¿ pueden ser asignados ?

Ejercicio M4D: 20

ado un sistema con una impresora, cinco CD-Rom's y dos cintas disponibles y las siguientes matrices :

Máximo			
Proceso	Impresoras	CD-Roms	Cintas
P0	0	0	1
P1	1	7	5
P2	2	3	5
P3	0	6	5
P4	0	6	5

Asignado			
Proceso	Impresoras	CD-Roms	Cintas
P0	0	0	1
P1	1	0	0
P2	1	3	5
P3	0	6	3
P4	0	0	1

Se pide :

- Indique cuál es el estado actual del sistema y justifique.
- Si P1 solicita (0, 4, 2) ¿ puede ser asignado ?
- Si P2 solicita (0, 1, 0) ¿ pueden ser asignados ?
- Si P4 solicita (0, 6, 0) ¿ pueden ser asignados ?

Ejercicio M4D: 21

Dado un sistema con 12 cintas, 5 impresoras, 3 CD-Roms, y 4 plotters, en donde se están ejecutando tres procesos P0, P1, P2, en el siguiente orden P1, P0, P2. Los procesos demandan, y tienen asignados los siguientes recursos:

Máximo

Proceso	Cintas	Impresoras	CD-Roms	Plotters
P0	10	2	1	2
P1	4	2	2	2
P2	9	3	1	3

Asignado

Proceso	Cintas	Impresoras	CD-Roms	Plotters
P0	5	2	1	2
P1	5	2	1	2
P2	3	1	0	0

Se pide justificar en que estado se encuentra el sistema

Ejercicio M4D: 22

Dado un sistema que se encuentra ejecutando el algoritmo del banquero, se pide que analice que sucedería si el proceso **E** solicita lo siguiente (1, 0, 1, 0). Tenga en cuenta que el estado del sistema es el siguiente.

Máximo				
Pr.	R1	R2	R3	R4
A	4	1	3	2
B	2	3	0	1
C	5	2	3	0
D	1	1	4	1
E	2	2	1	0

Asignado				
Pr.	R1	R2	R3	R4
A	2	0	3	2
B	1	3	0	0
C	3	1	1	0
D	1	0	3	1
E	1	2	0	0

Disponible			
R1	R2	R3	R4
1	1	1	1

Luego de la solicitud de **E** (si fue asignado continuar con el sistema como haya quedado), Llega una solicitud de **C** (0, 0, 0, 1), indique que pasa con cada una de las asignaciones.

Ejercicio M4D: 23

Dado un sistema que se encuentra ejecutando el algoritmo del banquero, se pide que analice que sucedería si el proceso **B** solicita lo siguiente (1, 0, 1, 0). Tenga en cuenta que el estado del sistema es el siguiente.

Máximo				
Pr.	R1	R2	R3	R4
A	4	1	3	2
B	2	2	1	0
C	2	3	0	1
D	5	2	3	0
E	1	1	4	1

Asignado				
Pr.	R1	R2	R3	R4
A	2	0	3	2
B	1	2	0	0
C	1	3	0	0
D	3	1	1	0
E	1	0	3	1

Disponible			
R1	R2	R3	R4
1	1	1	1

Luego de la solicitud de **B** (si fue asignado continuar con el sistema como haya quedado), Llega una solicitud de **C** (0, 0, 0, 1), indique que pasa con cada una de las asignaciones.

Ejercicio M4D: 24

Dado un sistema con la siguiente asignación de recursos, y el siguiente disponible de recursos se pide que evalúe cada una de las solicitudes que se indican a continuación.

Máximo			
	R1	R2	R3
P0	5	3	2
P1	3	5	6
P2	8	3	4
P3	3	2	1

Pedidos Resueltos:

P3(0,1,0), P0(0,0,1), P1(2,1,0), P2(1,2,0), P1(1,0,1), P2(1,1,0), P3(1,0,1), P0(1,1,0), P3(1,0,0)

Pedidos Pendientes:

P1 (0, 2, 2), P3(1, 1, 1), P2(1, 0, 3)

Nota: Todos los puntos deben estar debidamente justificados por el algoritmo del banquero y el de seguridad. El orden de los pedidos es el de la cola de espera, y cada uno debe ser resuelto luego del anterior, y con el estado que quedó el sistema.

Ejercicio M4D: 25

Indique en que estado dejaría al sistema que trabaja con el algoritmo del banquero. Responda *Seguro* / *inseguro* / *abrazo mortal* y justifique: (todas las respuestas deben estar justificadas para que sean tenidas en cuenta, si la justificación es incorrecta no importa el valor asignado a la misma):

- Incrementar el vector de disponible.
- Incrementar la matriz de necesidad para un determinado proceso.
- Un proceso puede redefinir (en menos) la cantidad máxima que declaró al iniciar.
- Decrementar el vector de disponible.
- Eliminar un proceso.
- Pasar de propuestos a listos procesos (con asignación estática de recursos).
- Pasar de propuestos a listos procesos (sin asignación estática de recursos).

Ejercicio M4D: 26

Dadas las siguientes matrices:

PROC ESO	Asignación corriente				Demanda máxima				Todavía Necesitan			
	r1	r2	r3	r4	r1	r2	r3	r4	r1	r2	r3	r4
p1	0	0	1	2	0	0	1	2				
p2	2	0	0	0	2	7	5	0				
p3	0	0	3	4	6	6	5	6				
p4	2	3	5	4	4	3	5	6				
p5	0	3	3	2	0	6	5	2				

- Calcule lo que los procesos todavía podrían peticionar en las columnas "Todavía Necesitan"
- El Sistema se encuentra en estado "Seguro o Inseguro" ¿Porqué?.
- El Sistema se encuentra en Abrazo Mortal o nó? ¿Porqué?.
- Cuales procesos, si es que hay alguno, están o pueden estar en abrazo mortal?
- Si una petición llega de p3 para (0,1,0,0). 1)Puede la petición ser satisfecha inmediatamente en forma segura o no?
2)¿en qué estado (Abrazo mortal, seguro, inseguro) dejaría esta petición al Sistema?. 3)¿qué procesos , si es que hay alguno, están o pueden estar en abrazo mortal si esta petición entera es otorgada inmediatamente?

Ejercicio M4D: 27 Modificado

En un sistema de cómputos que cuenta con el sistema operativo bark, se encuentran ejecutando cinco procesos que utilizan recursos críticos en distintas etapas de su vida. El sistema operativo utiliza una estrategia múltiple para evitar los abrazos mortales (deadlock). La primera de las estrategias es la de realizar Checkpoint. Entre Checkpoint y Checkpoint utiliza el algoritmo del Avestruz.

En un determinado momento el sistema se encuentra con la siguiente asignación de recursos:

- El proceso **A** tiene capturada el semáforo, y está esperando por la impresora.
- El proceso **B** tiene capturado la cinta contador de lectura del registro 1 de la base de datos, y está esperando por el W-CD.
- El proceso **C** tiene capturado el W-CD, y está esperando por la cinta.
- El proceso **D** tiene capturada la impresora y espera la cinta.

- El proceso **E** tiene capturado el semáforo contador de lectura del registro 1 de la base de datos, y está esperando por la impresora.

En último Checkpoint realizado los procesos se encontraban de la siguiente manera:

- El proceso **A** tiene capturada la impresora y espera la cinta
- El proceso **B** termina de capturar el semáforo contador de lectura del registro 1 de la base de datos
- El proceso **C** tiene capturado el W-CD
- El proceso **D** tiene capturada la cinta, y está esperando por el W-CD..
- El proceso **E** tiene capturado el semáforo contador de lectura del registro 1 de la base de datos, y está esperando por la cinta.

Se pide (todas las respuestas deben estar justificadas para que sean tenidas en cuenta, si la justificación es incorrecta no importa el valor asignado a la misma):

- Efectuar el gráfico de asignación de procesos y recursos (indique claramente que significa cada flecha – posee / necesita).
- Justifique cual es el mejor camino a tomar a partir de este momento para resolver el problema que se presenta en el sistema, y por el cual los procesos no terminan. Las posibles soluciones son realizar un Rollback, eliminar uno o más procesos (se debe indicar cual/es, y justificar). Para tomar una decisión debe tener en cuenta el costo de reprocesamiento. Dentro de la justificación deberá indicar los parámetros que tomó en cuenta para la decisión final. Dentro de su respuesta debe estar explicado el funcionamiento de cada uno de los algoritmos descriptos en el enunciado

○ ○ ○

Ejercicio M4D: 28

Dado un sistema que se encuentra ejecutando el algoritmo del banquero, y que cuenta con tres impresoras, tres CD-Rom's y dos cintas **disponibles** y las siguientes matrices:

Máximo			
Pr.	Impresoras	CD-Roms	Cintas
A	7	5	3
B	3	2	2
C	9	0	2
D	2	2	2
E	4	3	3

Asignado			
Pr.	Impresoras	CD-Roms	Cintas
A	0	1	0
B	2	0	0
C	3	0	2
D	2	1	1
E	0	0	2

Se pide (en todos los puntos se debe justificar la respuesta a través del algoritmo de seguridad/banquero):

- Indique cuál es el estado actual del sistema y justifique.
- Si **D** solicita tres impresoras y tres CD's, ¿puede ser asignado?
- Si **A** solicita dos CD's ¿pueden ser asignados?

○ ○ ○

Ejercicio M4D: 29

Módulo 5: Administración de Memoria.

MEMORIA REAL

En cuanto a la memoria central o principal, básicamente es como organizamos el espacio disponible en ésta: si utilizamos particiones fijas, particiones variables, buddy-system, paginación o segmentación.

Cuando utilizamos particiones variables, al introducir y remover bloques comienza a producirse fragmentación externa. Para poder hacer un uso mas eficiente de los espacios de memoria que comienzan a quedar libres, se utilizan diferentes algoritmos, los cuales repasamos aquí rápidamente:

- **First Fit:** Comienza a buscar en la memoria un bloque libre en el que quepa la partición. Cuando lo encuentra, lo introduce ahí.
- **Best Fit:** Busca en toda la memoria, pero no se conforma con el primero que encuentra, sino que busca el que mejor le calce; es decir, el que menos espacio desperdicie.
- **Worst Fit:** Es prácticamente el contrario al anterior. Busca, por decirlo de alguna forma, el bloque libre más grande posible para colocar al nuevo bloque; es decir, el que mas espacio desperdicie. Esto lo hace para que quede un nuevo bloque libre lo mas grande posible.
- **Next Fit:** Es parecido al First Fit. La diferencia es que la búsqueda de espacios libres la realiza a partir de la ubicación del último bloque; a partir de ahí, el primer lugar que encuentra para ubicar el bloque lo ubica.

Si utilizamos el algoritmo de buddy-system (o sistema de compañeros), lo que realiza es lo siguiente: va dividiendo a la memoria en mitades, y cada mitad la sigue dividiendo en mitades, así hasta llegar a una mitad cuyo tamaño no alcanza para ubicar el bloque, en ese caso, vuelve a la división anterior (une los pedazos que partió en el paso anterior) para ubicar el bloque. Cuando un bloque se libera, si la otra mitad contigua también está libre, entonces une ambas mitades para poder formar un bloque libre mas grande; es decir, comienza a agrupar las divisiones libres.

En la técnica de paginación, el proceso se dividen en páginas y la memoria en frames; tanto los frames como las páginas tienen el mismo tamaño y suelen ser potencias de 2. A cada proceso se le adhiere una tabla de páginas donde la entrada de cada una de las páginas contiene el número de frame en el cual están ubicadas.

La técnica de segmentación es similar a la paginación, con la diferencia de que el programa se divide en segmentos que suelen tener diferentes tamaños. Los procesos tienen también una tabla de segmentos, y cada entrada de segmento dice la dirección base y el tamaño de dicho segmento. Si el desplazamiento referenciado en la instrucción es mayor al tamaño del segmento, la dirección se toma como inválida; caso contrario se suma a la dirección base del segmento y se obtiene la dirección real.

Tanto en la técnica de paginación como en la de segmentación es necesario tener en cuenta el "formato" que adquiere la instrucción. Es decir, supongamos que se utiliza paginación: en este caso, la dirección lógica estará formada por un número de página (bits más a la izquierda) y el resto representará al desplazamiento (Offset) dentro de esa página. El número de página será utilizado como entrada en la tabla de páginas del proceso para así obtener el número de frame, de esta manera la dirección física estará formada por el número de frame y el mismo desplazamiento que antes. Básicamente:

Dirección lógica: #Pag / #Offset

Dirección física: #Frame / #Offset

Teniendo en cuenta este concepto podemos deducir que el tamaño de cada página (o el tamaño del frame en memoria) estará dado por el tamaño del desplazamiento; es decir, si el desplazamiento ocupa 8 bits, entonces el tamaño de cada página será de $2^8 = 256$ bytes.

MEMORIA VIRTUAL

En memoria virtual se utiliza ya sea la técnica de paginación, de segmentación o una combinación entre ambas. En cualquier de los casos, a las tablas de segmentos o de paginación de los procesos se les suele incorporar un bit de presencia y un bit de modificación.

Cuando se hace referencia a una página (o segmento) que no tiene frame asignado, es decir, la página no está cargada en memoria, se produce un **PAGE FAULT**. Es en este momento cuando la página es traída de disco a memoria y el bit de presencia de la entrada a esa página es modificado. Lo mismo sucede con el bit de modificación.

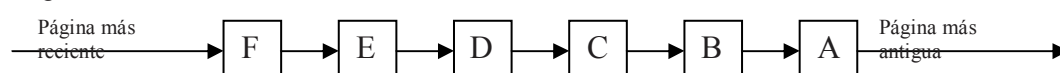
Cuando se trae de memoria virtual la página que ha sido referenciada, suele suceder que hay que escoger una víctima a ser reemplazada (en el caso de no tener frames libres), es decir, alguna de las páginas que

actualmente está ocupando un frame debe ser desalojada. Para elegir a esta “víctima” se suelen tener diferentes algoritmos:

- **Óptimo:** En sí es un algoritmo teórico. Se basa en determinar que páginas serán referencias con menor frecuencia en el futuro, las cuales son candidatas a ser desalojadas ya que no serán utilizadas brevemente y por lo tanto no producirán demasiados PAGE FAULTS. Como se debe predecir de alguna forma el futuro, es un algoritmo que no suele ser aplicado en la práctica.
- **FIFO:** Es el básico algoritmo ya conocido. Las páginas van siendo reemplazadas en el orden en que fueron llegadas.

El algoritmo más sencillo para remplazo de páginas es el FIFO (First In – First Out). Este algoritmo asocia a cada página el momento en que ésta fue traída a memoria. Cuando una página debe ser reemplazada se selecciona a la más antigua.

No es estrictamente necesario registrar el momento de entrada de la página a memoria, sino que se puede crear una cola en la que se van agregando las páginas conforme van llegando a la memoria (Ejemplo llega primero la Página “A”, luego la “B” y así sucesivamente. Cuando se debe eliminar una página, se selecciona la que está al frente de la lista (o sea, la más antigua de la lista). Cuando llega una página nueva, se inserta en la parte trasera de la cola. En la siguiente figura se representa el funcionamiento de éste algoritmo.



Reemplazo de páginas mediante el algoritmo FIFO.

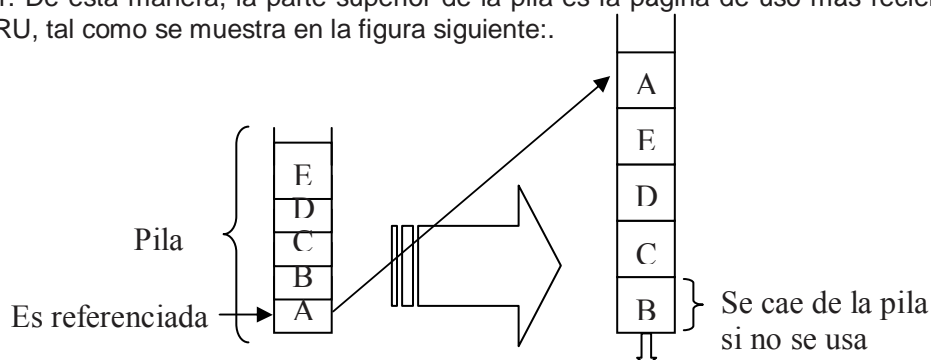
Al igual que el algoritmo aleatorio, este algoritmo es fácil de comprender y de programar. Sin embargo, su desempeño no siempre es del todo bueno. La página reemplazada puede ser un módulo de inicialización que fue usado hace mucho tiempo y ya no se tiene necesidad de él. Por otro lado, puede contener una variable de uso muy frecuente que fue inicializada de manera temprana y está en uso constante.

□ LRU - Algoritmo de reemplazo de páginas “la de menor uso reciente”

Este algoritmo es una buena aproximación al óptimo y se basa en la observación de que las páginas de uso frecuente en las últimas instrucciones se utilizan con cierta probabilidad en las siguientes. De la misma manera, es probable que las páginas que no hayan sido utilizadas durante mucho tiempo permanezcan sin uso por bastante tiempo. Implementando el algoritmo con esta base, al ocurrir un fallo de página, se elimina la página que no haya sido utilizada durante el tiempo más grande. De ahí su denominación: menor uso reciente (LRU - Least Recent Use).

A diferencia de los algoritmos anteriores, el LRU tiene un mejor rendimiento en cuanto al tiempo de aprovechamiento del CPU y del uso de la memoria. Sin embargo, el problema con este algoritmo es que su implementación es muy cara, ya que requiere de una asistencia considerable de hardware. Otro problema es el de determinar un orden para los marcos definido por el tiempo de menor uso. Para éste último hay dos posibles implementaciones:

- **Contadores:** En el caso más sencillo, se asocia cada entrada tabla-página un campo de tiempo-de-uso y se le agrega al CPU un reloj lógico o contador. Este reloj es incrementado en cada referencia de memoria. Siempre que se hace referencia a una página, el contenido del registro del reloj es copiado al campo de tiempo-de-uso en la tabla de páginas para esa página. De esta forma, siempre se dispone del “tiempo” de la última referencia a cada página. La página que se reemplaza es la del menor valor de tiempo. Este esquema requiere de una búsqueda en toda la tabla de páginas para encontrar la página LRU, y una escritura en memoria al campo de tiempo-de-uso en la tabla de páginas por cada acceso a memoria. Los tiempos también se deben de mantener cuando las tablas de páginas son alteradas (debido a organización del CPU). Se debe considerar la posibilidad de sobrecarga en el reloj.
- **Pilas:** Otra aproximación para implementar el reemplazo LRU es la de tener una pila con los números de páginas. Siempre que se hace referencia a una página, se quita de la pila y se pone en la parte superior. De esta manera, la parte superior de la pila es la página de uso más reciente y la de abajo es la LRU, tal como se muestra en la figura siguiente:.



Uso de pilas en el algoritmo LRU

- **Last Frequently Used (LFU):** Escoge las páginas que han tenido menor número de referencias.
Reloj: Suele tener un bit de uso por página. Cuando una página es referenciada, este bit suele ponerse a 1 (uno). Cuando se debe elegir una página para reemplazar se recorren los bits de todas las páginas; aquellos que se encuentran en “uno” se ponen a “cero” y aquellos que ya estaban en “cero” indican que la página puede ser extraída.

□ **Algoritmo de reemplazo de páginas según el uso no tan reciente (NUR: Not Used Recently)**

Este algoritmo hace uso de los dos bits de estado que están asociados a cada página. Estos bits son: R, el cual se activa cuando se hace referencia (lectura / escritura) a la página asociada; y M, que se activa cuando la página asociada es modificada (escritura). Estos bits deben de ser actualizado cada vez que se haga referencia a la memoria, por esto es de suma importancia que sean activados por el hardware. Una vez activado el bit, permanece en ese estado hasta que el sistema operativo, mediante software, modifica su estado.

Estos bits pueden ser utilizados para desarrollar un algoritmo de reemplazo que cuando inicie el proceso, el sistema operativo asigne un valor de 0 a ambos bits en todas las páginas. En cada interrupción de reloj, limpie el bit R para distinguir cuáles páginas tuvieron referencia y cuáles no.

Cuando ocurre un fallo de página, el sistema operativo revisa ambos bits en todas las páginas y las clasifica de la siguiente manera:

Clase 0: La página no ha sido referenciada, ni modificada.

Clase 1: La página no ha sido referenciada, pero ha sido modificada.

Clase 2: La página ha sido referenciada, pero no ha sido modificada.

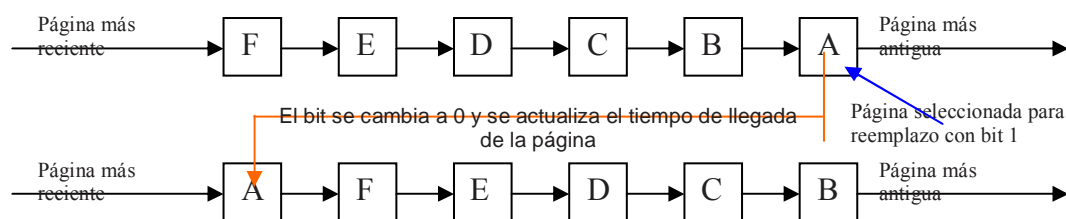
Clase 3: La página ha sido referenciada y también modificada.

Una vez obtenida la clasificación, elimina una página de manera aleatoria de la primera clase no vacía con el número más pequeño. Esto porque para el algoritmo es mejor eliminar una página modificada sin referencias en al menos un intervalo de reloj, que una página en blanco de uso frecuente.

A pesar de que este algoritmo no es el óptimo, es fácil de implementar y de comprender y con mucha frecuencia es el más adecuado.

• **Algoritmo de reemplazo de páginas de la segunda oportunidad (Second-Chance Algorithm)**

Este algoritmo es una modificación del FIFO. El algoritmo hace uso del bit de referencia de la página. Cuando una página ha sido seleccionada para reemplazo, se revisa el bit de referencia. Si tiene valor de 0, se procede a reemplazar la página. Si por el contrario, el bit de referencia es 1 se le da a la página una segunda oportunidad. En la siguiente figura se puede apreciar el funcionamiento del algoritmo:



Algoritmo de la segunda oportunidad.

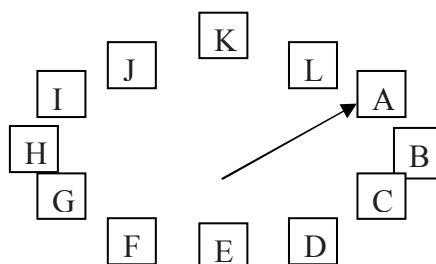
Cuando esto sucede, se le cambia el bit de referencia a 0 y se actualiza su tiempo de llegada al tiempo actual para que la página se colocada al final de la cola. De esta manera, la página espera todo un ciclo completo de páginas para ser entonces reemplazada.

Si la página tiene un uso muy frecuente, el bit de referencia se mantendría constantemente en 1 y la página no sería reemplazada..

□ **Algoritmo de reemplazo de páginas del reloj**

Modificando el algoritmo de la segunda oportunidad (que a su vez es una modificación de FIFO) obtenemos el algoritmo aumentado de la segunda oportunidad o algoritmo del reloj. Usamos la misma clasificación vista en el algoritmo de uso no tan reciente Este algoritmo organiza las páginas en una

lista circular como se muestra en la siguiente figura y se usa un apuntador (o manecilla) que señala a la página más antigua.



Algoritmo de reloj.

Cuando se presenta un fallo de página, el algoritmo revisa la página a la que está apuntando la manecilla. Si el bit de referencia es 0, la página es reemplazada con la nueva y la manecilla avanza una posición. Si el bit es 1, entonces se limpia (cambia a 0) y la manecilla avanza a la siguiente página y así sucesivamente hasta encontrar una con bit 0.

También hay que tener en cuenta que las políticas de reemplazo se realizan sobre el conjunto residente de un proceso, esto es, las páginas que actualmente se ubican en memoria. Este conjunto residente puede ser fijo (el número de páginas siempre es el mismo) o variable (puede tener más o menos páginas en memoria). Por otro lado, el alcance del reemplazo puede ser local (se reemplazan las páginas del proceso que produjo el fallo) o global (se consideran todas las páginas residentes en memoria). Teniendo en cuenta estas alternativas, veamos que posibilidades pueden ser aplicadas:

- **Asignación fija, alcance local:** Se reemplazan alguna de las páginas del proceso que produjo el fallo.
- **Asignación variable, alcance local:** Se consideran las páginas del proceso que produjo el fallo, pero se utilizan diferentes criterios (como el PFF) para determinar si le quito un frame al proceso o le sumo uno más (le agrego una página mas en memoria).
- **Asignación fija, alcance global:** Imposible de implementar. Como hemos visto, si la asignación es fija el número de páginas del proceso no puede cambiar. Si consideramos todas las páginas de memoria, puede pasar que al proceso se le sume o se le quite un frame más, lo cual contradice lo anterior.
- **Asignación variable, alcance global:** Se toman en cuenta todas las páginas residentes en memoria (no solo las del proceso que produjo el Page Fault) y se reemplaza alguna según el algoritmo utilizado. El número de frames asignados al proceso puede crecer o puede disminuir.

Vale aclarar también que cuando se hace referencia en el alcance global de “considerar todos los frames en memoria”, estamos tomando en cuenta a todos los frames disponibles (estén utilizados o no) y que no son frames bloqueados. Debe tenerse en cuenta que el sistema operativo (o al menos una parte de él) siempre reside en memoria y como tal, ocupa frames en la misma; la diferencia está en que reside en frames bloqueados, los cuales no pueden ser considerados para reemplazar o asignar páginas.

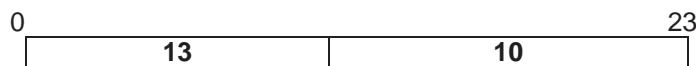
Sobre la paginación de procesos podemos hacer referencia también a un fenómeno denominado “Anomalía de Belady”. Esta anomalía se presenta cuando al aumentar la cantidad de páginas por procesos (se le asignan mas frames), también aumenta la cantidad de PAGE FAULT. Obviamente, esta es una situación inesperada ya que al parecer, si se le das mas memoria a un proceso, cada vez que haga alguna referencia hay más probabilidades de encuentre esa referencia en memoria, sin embargo, esto no sucede y los fallos de páginas se incrementan.

EJEMPLOS DE EJERCICIOS RESUELTOS DE MEMORIA:

Ejercicio RM5: 1

Sea un sistema de administración de memoria paginada por demanda. El sistema tiene una memoria real de 64k y un bus de direccionamiento de 23 bits. Se direcciona por bloques de 8k. Desprecie el espacio ocupado por el sistema operativo en memoria. Se pregunta:

- a) ¿Cuál es el tamaño de un programa a ejecutar en este sistema?



Tamaño de la página = $2^{13} = 8k$.

Cantidad de páginas = $2^{(23-13)} = 2^{10} = 1024$.

Tamaño de Página = 8k.

- b) ¿Cuál es el máximo y el mínimo número de página que puede direccionar un programa en esta instalación y por qué?

Número Mínimo de Página: Página 0.

- c) Sea el programa PROG1 que cuenta con 52 páginas. El programa bifurca en un momento a las direcciones virtuales 310cfh y 7ab08h. Indique cuáles la página y el desplazamiento al cual se bifurca.

310Cf = 0011 0001 0000 1100 1111 b

Página = 24.

Desplazamiento = 4303.

7AB08 = 0111 1010 1011 0000 1000

Página = 61.

Desplazamiento = 2824.

Esta dirección no pertenece al programa debido a que tiene 52 páginas y la dirección corresponde a la página 61.

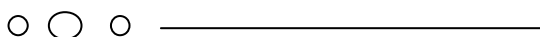
- c) ¿Qué operaciones se requerirían al módulo de administración de memoria si las páginas estuvieran cargadas en memoria real?

Estando cargadas la páginas en memoria central el administrador de memoria pone a disposición de la CPU la MPT (Memory Page Table) para que esta pueda hacer los cálculos y hallar así el Frame correspondiente a la página pedida.

- d) ¿Qué operaciones se requerirían al módulo de administración de memoria si las páginas estuvieran cargadas en memoria virtual?

Al ocurrir una interrupción por Page Fault el módulo de administración de memoria carga desde la memoria secundaria la/las página/as faltantes. Luego actualiza las tablas de control (MPT, MFT y la tabla de mapas de archivos).

Si no hay espacio suficiente en memoria para cargar las páginas solicitadas se lo hace seleccionando una víctima para llevar a la memoria secundaria mediante un algoritmo de selección.



Ejercicio RM5: 2

Suponga un sistema de administración de memoria paginada por demanda. El tamaño de página es de 1k y el bus de direcciones de 32 bits. Se pregunta:

- a) ¿Cuál es el tamaño de la memoria?

22 bits	10 bits
Page Number	Page Offset

Tamaño de la Memoria: 2^{22} páginas * 2^{10} tamaño de página = 4 gb.

- b) Dado un proceso de 3k ¿cuántas páginas tiene? ¿Cuántas conviene cargar en memoria real? ¿De qué depende su respuesta anterior?

Tamaño de la página = 1k.

Cantidad de Páginas del Proceso = 3 pag.

Conviene cargarlas todas en memoria central debido a que el S.O. busca cargar de una sola vez todas las páginas para evitar el overhead que significaría cargarlas de a una cada vez que fueran solicitadas.

- c) Numere las páginas del proceso del punto anterior comenzando en cero, suponga que dispone de 3 marcos de página en memoria real para ejecutar este proceso. En cierto momento, necesita cargar una página no presente. Enumere los pasos de hardware y de software necesarios para llevar a cabo esta tarea.

**Página 0
1K**

**Pág. 1
1K**

**Pág. 2
1K**

Proceso = 3K.

Página = 1K.

1. Un trabajo pide una página que no está presente en memoria central (Software).
2. El hardware de mapeo genera una interrupción por falta de página (Hardware).
3. El S.O. procesa esta interrupción, no como una interrupción de protección sino, cargando las páginas requeridas desde el almacenamiento secundario (Hardware).
4. El S.O. actualiza en forma adecuada las entradas de la tabla de páginas, luego reinicia la ejecución de la instrucción interrumpida (Software).
5. Si no es posible encontrar un bloque libre, un algoritmo destina a la elección de la "Víctima" se encarga de seleccionar una página. Esta selección depende del algoritmo utilizado, Ej.: FIFO: la más vieja, OPT: La que no va ser usada por el período de tiempo más largo, etc (Software).
6. Una vez selecciona la página "Víctima" ésta se copia a memoria secundaria (Swap-Out), liberando así espacio para la página pedida (Hardware).

_____ ○ ○ ○ _____

Ejercicio RM5: 3

Suponga un proceso de 7 páginas, numeradas de 0 a 6, y administración de memoria paginada por demanda. La traza de la ejecución del proceso es 0120460120345. Se le asignan tres marcos de páginas. La estrategia de remplazo es local. Se pide diagramar el esquema de remoción según:

- a) FIFO ¿Cuál es el índice de falla?

Referencia	0	1	2	0	4	6	0	1	2	0	3	4	5
Frame 1	0	1	2	0	4	6	0	1	2	0	3	4	5
Frame 2		0	1	2	0	4	6	0	1	2	0	3	4
Frame 3			0	1	2	0	4	6	0	1	2	0	3

P.F.

F

F

F

F

F

F

F

F

F

F

13 referencias = 7 fallas + 6 aciertos. Índice de fallas = $7 / 13 = 0,5384$.

- b) LRU ¿Cuál es el índice de falla?

Referencia	0	1	2	0	4	6	0	1	2	0	3	4	5
Frame 1	0	0	0	0	0	0	0	0	0	0	0	0	5
Frame 2		1	1	1	4	4	4	1	1	1	3	3	3
Frame 3			2	2	2	6	6	6	2	2	2	4	4

P.F.

F

F

F

F

F

F

F

F

F

F

13 referencias = 7 fallas + 6 aciertos.

Índice de fallas = $7 / 13 = 0,5384$.

c) Algoritmo óptimo ¿Cuál es el índice de falla?

Referencia	0	1	2	0	4	6	0	1	2	0	3	4	5
Frame 1	0	0	0	0	0	0	0	0	0	0	3	3	3
Frame 2		1	1	1	1	1	1	1	2	2	2	2	5
Frame 3			2	2	4	6	6	6	6	6	6	4	4

P.F.

F

F

F

F

F

F

13 referencias = 6 fallas + 7 aciertos.

Índice de fallas = $6 / 13 = 0,4615$.

Ejercicio RM5: 4

Suponga un esquema de administración de memoria paginada por demanda. Cierta proceso mide 4K. El tamaño de la página es de 0.5K. El tamaño de la memoria real es de 8Mb. Y el de la memoria virtual es de 24 Mb. Se pregunta:

a) Cuántas páginas tiene el proceso;

Tamaño de página = 0,5k.

Tamaño del proceso = 4K.

Cantidad de páginas = 8.

b) ¿Cuántos bits se necesitan para direccionar en este sistema?

¿Cuántos para el número de páginas y cuántos para el desplazamiento?

Se necesitan 25 bits para direccionar en este sistema.

$2^{25} = 32$ Mb de memoria

$2^9 = 512$ bytes de tamaño de página.

$2^{25} / 2^9 = 2^{16} \rightarrow 16$ bits para el número de página.

16 bits	9 bits
Número de Página	Offset

c) ¿A qué página direcciona el proceso si la dirección absoluta de la próxima instrucción es 1F9Ah?

1F9Ah = 0001 1111 1001 1010 b

Número de páginas = 15.

Offset = 410.

Ejercicio RM5: 5

Una computadora tiene cuatro marcos de página. Se lista el tiempo de carga, tiempo de último acceso y los bits de remoción (bit R) y de modificación (bit M) (los tiempos son ticks de reloj).

Página	Cargada	Última referencia	Bit R	Bit M
0	126	279	0	0
1	230	260	1	0
2	120	272	1	1
3	160	280	1	1

a) ¿Cuál página reemplazará NRU?

La página que reemplazará es la 0, ya que el algoritmo Not used Recently trabaja con un bit de referencia el cual pone un uno cada vez que se accede a esa página. Y el candidato a ser sustituido es aquel que tenga el bit de referencia en 0, en este caso la página 0 es la única que tiene su bit de Referencia en 0.

b) ¿Cuál página reemplazará FIFO?

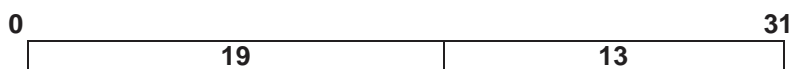
El algoritmo FIFO reemplazará la página 2. Ya que es la primer página cargada de las cuatro (Primera ingresada primera en salir).

c) ¿Cuál página reemplazará LRU?

Last recently used reemplazará la página 1, debido a que es la página que se uso en el periodo de tiempo más lejano de las cuatro. En conclusión su uso fue el menos reciente.

Ejercicio RM5: 6

Una máquina tiene un espacio de direccionamiento de 32 bits y un tamaño de página de 8k. La tabla de páginas está en hardware, con una palabra de 32 bits en cada entrada. Cuando un proceso comienza, la tabla de páginas es copiada desde el hardware a la memoria, una palabra cada 100 nseg. Si cada proceso ejecuta durante 100 nseg. (incluyendo el tiempo de carga de la tabla de páginas). ¿Qué fracción del tiempo de CPU se dedica a cargar las tablas de páginas?



Su direccionamiento es de 32 bit (del 0 al 31), el Page Number es de 19 bit (un tamaño de página de $8k = 2^{19}$) y el Page Offset es de 13 bit.

La tabla tiene 524288 entradas multiplicado por el tiempo que tarda que es de 100 nseg. Entonces la carga de la tabla tarda 52428800 nanosegundos.

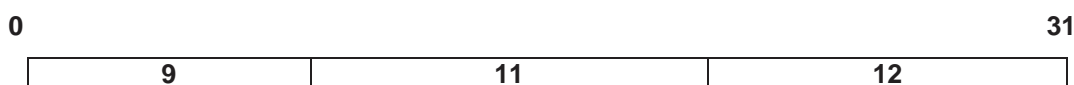
Por lo tanto el porcentaje de tiempo para cargar la tabla es de:

$$\frac{52428800 \text{ nseg.}}{10^6 \text{ nseg.}} = 52.42 \%$$



Ejercicio RM5: 7

Una computadora con un direccionamiento de 32 bits usa una tabla de página de dos niveles. Las direcciones virtuales se dividen en un campo de 9 bits par la tabla de primer nivel, y un campo de 11 bits para la tabla de segundo nivel, y un desplazamiento. ¿Qué tamaño tiene que tener las páginas y cuántas hay en un espacio de direcciones?



Tamaño de la página = $2^{\text{offset}} = 2^{12} = 4096$.

Entonces el tamaño de la página es de 4k.

La cantidad de páginas es de $2^{20} = 1024 \text{ kbyte} = 1 \text{ Megabyte}$.



Ejercicio RM5: 8

Supongamos que una dirección virtual de 32 bits se divide en cuatro campos a, b, c y d. Los primeros tres se usan para un sistema de paginación de tres niveles. El cuarto d, es el desplazamiento. ¿Depende el número del tamaño de los cuatro campos?. Si no ¿Cuáles importan y cuáles no?.

El número de paginas solo depende de los tres primeros números (el a, b y c) y no del cuarto (el d).



Ejercicio RM5: 9

Una pequeña computadora tiene cuatro marcos de páginas. En el primer tick de reloj, los bit R son 0111 (página 0 es 0, el resto 1). En ticks subsecuentes, los valores 1011, 1010, 1101, 0010, 1010, 1100 y 0001. Si el algoritmo de envejecimiento se usa con un contador de 8 bits, dar los valores de los cuatro contadores después del último tick.

	0111	1010	1101	0010	1010	1100	0001
00000000	00000000	10000000	11000000	01100000	10110000	11011000	01101100
00000000	10000000	01000000	10100000	01010000	00101000	10010100	01001010
00000000	10000000	11000000	01100000	10110000	11011000	01101100	00110110
00000000	10000000	01000000	10100000	01010000	00101000	00010100	10001010



Ejercicio RM5: 10

Una computadora provee a cada proceso con un espacio de direccionamiento de 65536 bytes dividido en páginas de 4096 bytes. Un programa particular tiene un código de 32768 bytes, datos 16386 bytes y pila de 15879 bytes. ¿Entrará este programa en el espacio de direcciones? Si el tamaño de cada página fuera de 512 bytes, ¿entraría?. Recuerde que una página no puede contener partes de dos segmentos diferentes.

Con páginas de 4096 bytes:

- Para texto $32768 / 4096 = 8$ páginas.
- Para datos $16386 / 4096 = 5$ páginas.
- Para pila $15870 / 4096 = 4$ páginas.

El total de páginas es de 17 pero solo hay 16 páginas (65536 / 4096).

Con páginas de 512 bytes:

- Para texto $32768 / 512 = 64$ páginas.
- Para datos $16386 / 512 = 33$ páginas.
- Para pila $15870 / 512 = 341$ páginas.

Por lo tanto preciso en total 128 páginas y son exactamente las que tengo.

_____ ○ ○ ○ _____

Ejercicio RM5: 11

Si ha observado que el número de instrucciones ejecutadas entre fallas de páginas es directamente proporcional al número de marcos asignados a un programa. Si la memoria disponible se duplica, el intervalo entre fallas de páginas también se duplica. Supongamos que una instrucción normal tarda 1 microseg, pero si ocurre una falla de página, toma 2001 microseg. (o sea, 2 mseg para manejar la falla). Si un programa tarda 60 segundos para ejecutar, cuánto tiempo le llevará atender 15000 fallas de páginas, y cuanto le llevará ejecutar si hay casi el doble de memoria disponible?.

El programa tarda 60 seg. En ejecutarse con 15000 fallos de página con el doble de memoria se tendrán 7500 fallos de páginas.

Por lo tanto tardaría: $60 \text{ seg.} - 7500 \cdot 2000 = 60 \text{ seg.} - 15 \text{ seg.} = 45 \text{ seg.}$

_____ ○ ○ ○ _____

Ejercicio RM5: 12 FINAL UTN del 01/03/2003

Nuestro Servidor es un AS/400 modelo 150, sub-modelo 2269, cuyo número de serie es 101492R. Es uno de los "pequeños" de IBM, su Performance es de 27 (CPW), y el fitcher interactivo es de 13.8 CPW. El sistema Operativo es el OS400 V4R4M0 y el último nivel de PTF aplicada (Service Pack) es TL00350. Se dispone de 1 solo procesador con 64 MB de memoria RAM y el nivel de seguridad adoptado es 30 (Permisos a nivel usuario).

En la plataforma AS/400, la memoria física se divide en agrupaciones lógicas llamadas POOLes de memoria. Cada POOL tiene asociados uno o más subsistemas en el cual corren trabajos y tareas del sistema.

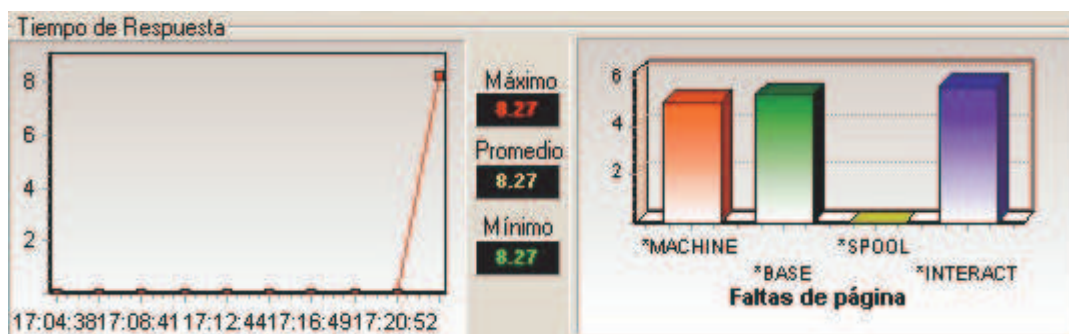
Algunos POOLes son específicos del sistema y otros son creados por los usuarios. Los POOLes del sistema son:

- *MACHINE, donde corren las tareas.
- *SPOOL, donde corren los trabajos de impresión.
- *INTERACT, donde corren los trabajos interactivos (Los que provienen de una consola interactiva y/o emulación 5250)
- *BASE, donde corren los trabajos Batch y los que no tienen otro POOL especificado.

En el AS/400, los defaults son regidos por valores del sistema. El valor del sistema QPFRADJ permite especificar si el tamaño de los POOLes es estático, Cada POOL queda con el tamaño que el usuario indica y no se modifica, si es 0, o si el tamaño del POOL va a ser administrado dinámicamente por el Sistema Operativo si fuese 3.

La tarea que se nos encarga es analizar el rendimiento del AS/400. Lo que nos interesa optimizar es el tiempo de respuesta de los usuarios, para lo cuál nos enfocaremos en el subsistema *INTERACT.

Para dicha tarea hemos puesto el valor del sistema QPFRADJ en 0, sin ajuste, y mediante la ejecución de un trabajo interactivo (QPADEV0005) que realiza un query en el archivo físico SUELDOS, hemos logrado la siguiente instantánea de Performance, obtenida con una herramienta de monitorización ON-Line:



Al cambiar el valor de QPFRADJ a 3, la nueva distribución de los POOLes es:

Pool	Tamaño
*MACHINE	10 MB
*INTERACT	20 MB
*SPOOL	4 MB
*BASE	30 MB

Se disponen de 3 frames (0,1,2) para el proceso dentro del POOL *INTERACT. Las direcciones son de 20 bits, con 12 bits para desplazamiento. La pila de llamadas del proceso interactivo (QPADEV0005) tiene las siguientes referencias a memoria: 1052 – 12864 – 5968 – 658 – 8954 – 147 (decimal)

Si la política de elección de la víctima es LRU con asignación fija y alcance local, se pregunta:

1) ¿Se puede afirmar que es conveniente el ajuste automático de memoria?. Justifique claramente su respuesta.

En el AS/400, los procesos tienen un esquema de 100 posibles prioridades. Las tareas del sistema se ejecutan con prioridad 0, mientras que los trabajos tienen prioridades entre 1 y 99, el menor valor indica la máxima prioridad.

A su vez, cada trabajo dispone de un quantum de tiempo, llamado Timeslice. Dicho valor se mide en milisegundos y puede ser cualquiera dentro del rango 1 y 9999999.

La prioridad de cada trabajo se define en un objeto del tipo *CLS, siendo para los procesos BATCH de 50 y los INTERACTIVOS de 20.

Los valores por omisión de Timeslice de cada trabajo dependen del subsistema en el cual fueron creados pero para simplificar las cosas tomaremos que:

Trabajos BATCH : 5000
INTERACTIVOS: 2000
SYSTEM: 1000

Una vez que el trabajo obtiene la CPU para ser ejecutado, se mantiene hasta que necesite alguna operación de Entrada/Salida, alcance el total de su Timeslice, finalice su ejecución, o sea desplazado por otro trabajo con prioridad superior.

2) Dada la siguiente tabla de procesos, suponiendo que cada ciclo de CPU utiliza su timeslice completo y que se cuenta con un único dispositivo de entrada / salida, determinar la traza de ejecución y el orden de finalización, mediante un diagrama de Gantt, en forma clara y detallada.

Instante de entrada (milisegundos)	Nombre Trabajo (Tipo)	CPU	E/S	CPU	E/S	CPU
0	STRBDMON (BATCH)	xxx	3000	xxx	5000	xxx
2200	P_USRSPC (INTERACT)	xxx	1000	xxx	1000	xxx
3500	DLTSPool (BATCH)	xxx	2000	xxx	800	xxx
2600	WRKACTJOB (SYSTEM)	xxx	2800	xxx	4000	xxx

SOLUCIÓN

Pedido	Página	Offset	Pto b			Pto a	
			Frame	Desplazamiento	Dirección Física	PF	Mapa de Memoria
1052	0	1052	0	1052	1052		(0, ,)
12864	3	576	1	576	4672		(0,3,)
5968	1	1872	2	1872	10064		(0,3,1)
658	0	658	0	658	658		(0,3,1)
8954	2	762	1	762	4858	SI	(0,2,1)
147	0	147	0	147	147		(0,2,1)

Pedido	Página	Offset	Frame	Desplazamiento	Dirección Física	PF	Mapa de Memoria
1052	0	10000011100	0	10000011100	10000011100		(0, ,)
12864	11	1001000000	1	1001000000	11001001000000		(0,3,)
5968	1	11101010000	10	11101010000	1011101010000		(0,3,1)
658	0	1010010010	0	1010010010	1010010010		(0,3,1)
8954	10	1011111010	1	1011111010	10001011111010	SI	(0,2,1)
147	0	10010011	0	10010011	10010011		(0,2,1)

Pedido	Página	Offset	Frame	Desplazamiento	Dirección Física	PF	Mapa de Memoria
1052	0	41C	0	41C	41C		(0, ,)
12864	3	240	1	240	1240		(0,3,)
5968	1	750	2	750	2750		(0,3,1)
658	0	292	0	292	292		(0,3,1)
8954	2	2FA	1	2FA	12FA	SI	(0,2,1)
147	0	93	0	93	93		(0,2,1)

Es mas conveniente porque este método produce 1 page fault, en cambio el otro produce un promedio de 6 page faults (ver grafico de barras)



Ejercicio RM5: 13 FINAL UTN del 06/12/2003

La gestión de memoria de un sistema operativo debe permitir que sea posible ejecutar concurrentemente 3 procesos, donde los tamaños en **bytes** de cada uno de los segmentos que forman parte de sus imágenes de memoria son los siguientes:

PROCESO	CODIGO	DATOS	PILA
A	16384	8700	8192
B	2048	1000	1024
C	4096	2272	2048

Además, se sabe que se dispone de una memoria física de 16 Kbytes y que el espacio de direcciones del sistema es de 64 Kbytes.

Se pide:

- 1.1) Determinar si son viables tamaños de página de 1024 bytes y 512 bytes suponiendo que una página no puede contener partes de dos segmentos diferentes (código, datos o pila).
- 1.2) Suponiendo de que ambos tamaños de página sean posibles, justificar qué tamaño debería utilizar el sistema de paginación si se tiene en cuenta la fragmentación interna producida.
- 1.3) Si se desea que las entradas de la tabla de páginas dispongan de 3 bits para referenciar el Frame de cada página, ¿qué tamaño de página debería utilizarse sin tener en cuenta la ejecución de los procesos A, B y C?

SOLUCIÓN

1.1. Sabiendo que una página no puede contener partes de dos segmentos diferentes (código, datos o pila), se obtienen los siguientes valores:

PROCESO A	Nº bytes	Nº páginas 1024 bytes	Fragmentación interna	Nº páginas 512 bytes	Fragmentación interna
CODIGO	16384	16	0	32	0
DATOS	8700	9	516	17	4
PILA	8192	8	0	16	0

PROCESO B	Nº bytes	Nº páginas 1024 bytes	Fragmentación interna	Nº páginas 512 bytes	Fragmentación interna
CODIGO	2048	2	0	4	0
DATOS	1000	1	24	2	24
PILA	1024	1	0	2	0

PROCESO C	Nº bytes	Nº páginas 1024 bytes	Fragmentación interna	Nº páginas 512 bytes	Fragmentación interna
CODIGO	4096	4	0	8	0
DATOS	2272	3	800	5	288
PILA	2048	2	0	4	0

- Con páginas de 1024 bytes se necesitan para poder ejecutar los 3 procesos cargar en memoria:

$$16+9+8+2+1+1+4+3+2 = 46 \text{ páginas}$$

Puesto que el sistema tiene un espacio de direcciones de 64 Kbytes (65536 bytes), el número de páginas posibles sería: $65536 / 1024 = 2^{16} / 2^{10} = 2^6 = 64$ páginas.

Para poder ejecutar los procesos se necesitan 46 páginas al menos, y se dispone de 64 por lo que es posible utilizar páginas de 1024 bytes.

- Con páginas de 512 bytes se necesitan para poder ejecutar los 3 procesos cargar en memoria:

$$32+17+16+4+2+2+8+5+4 = 90 \text{ páginas}$$

Puesto que el sistema tiene un espacio de direcciones de 64 Kbytes (65536 bytes), el número de páginas posibles sería: $65536 / 512 = 2^{16} / 2^9 = 2^7 = 128$ páginas.

Para poder ejecutar los procesos se necesitan 90 páginas al menos, y se dispone de 128 por lo que es posible utilizar páginas de 512 bytes.

Por tanto, es viable utilizar cualquiera de los dos tamaños de página.

1.2. Teniendo en cuenta la fragmentación interna producida se elegirá aquel tamaño de página en el que se desperdicie menos memoria en las últimas páginas:

Con 1024 bytes $\Rightarrow 516+24+800=1340$ bytes desperdiciados

Con 512 bytes $\Rightarrow 4+24+288=316$ bytes desperdiciados

Por tanto se elegirán páginas de 512 bytes.

1.3. El número de marcos de página posibles en el sistema en función del tamaño de página es:

Páginas de 1024 bytes $\Rightarrow 16384 / 1024 = 2^{14} / 2^{10} = 2^4 \Rightarrow$ Se necesitan 4 bits para hacer referencia al marco de página.

Páginas de 512 bytes $\Rightarrow 16384 / 512 = 2^{14} / 2^9 = 2^5 \Rightarrow$ Se necesitan 5 bits para hacer referencia al marco de página.

Por tanto, ninguno de los dos tamaños de página propuestos es adecuado para los 3 bits de referencia al marco de página. Lo ideal es poder referenciar todos los marcos de página posibles en memoria física, por lo que se diseñaría un sistema con $2^3 = 8$ marcos $\Rightarrow 16384 / 8 = 2048 \Rightarrow$ Se deberían utilizar **páginas de 2048 bytes**.

Ejercicio RM5: 14

Final UTN del 29/05/06

Se tiene un sistema cuyo algoritmo de reemplazo de páginas es el del reloj modificado y en el cual la siguiente tabla muestra la ocupación actual de los marcos con páginas (por ejemplo el marco 0 contiene actualmente la página 17, el marco 1 contiene la página 32, etc.).

Frame Number	0	1	2	3	4	5	6	7
Page Number	17	32	41	5	7	13	2	20
Bit Uso	1	0	0	0	0	1	1	0
Bit Modificación	1	0	1	1	1	1	1	1


Si el puntero del reloj apunta al marco 4 y la secuencia de pedidos de página es la siguiente:

32 (escritura), 14 (lectura), 15 (escritura), 2 (lectura), 18 (lectura), 14 (escritura)

Contestar:

- ¿Qué páginas serán reemplazadas de acuerdo a la secuencia de pedidos?. Debe indicar la secuencia de reemplazo.
- Indicar la ocupación de los marcos y el valor de los bits de uso y modificación de manera similar a la presentada en el enunciado de este ejercicio al finalizar la atención de los pedidos.

SOLUCIÓN

				PEDIDOS					
Uso	Modif	Frame	Page	Write	Read	W	R	R	W
				32	14	15	2	18	14
1 0	1	0	17		17	17		17	
0 1 0	0 1	1	32	√	32	32		32	
0 1	1 0	2	41		40	41		<u>18</u>	
 0 0	1	3	5		<u>5</u>	5		5	
0 1	1 0 1	4	7		<u>14</u>	14		14	√
1 0	1	5	13		13	13		13	
1 0 1	1	6	2		2	<u>2</u>	√	2	
0 1	1	7	20		20	<u>15</u>		15	
				PF		PF		PF	

EXPLICACIÓN:

1er PEDIDO: PAGINA 32 (ESCRITURA) PUNTERO DEL RELOJ APUNTA AL MARCO 4 (Frame #3) SE OBSERVA QUE LA PAGINA 32 ESTA EN MEMORIA, POR LO TANTO SE PUEDE ATENDER EL PEDIDO Y SE ACTUALIZA LA TABLA:

NÚMERO DE FRAME:	1
NÚMERO DE PAGE;	32
NÚMERO DE BIT DE USO;	1
NÚMERO DE BIT DE MODIFICACIÓN:	1

2do PEDIDO: PAGINA 14 (LECTURA) SE OBSERVA QUE LA PAGINA 14 NO ESTA EN MEMORIA, POR LO TANTO SE DEBE CARGARLA PRODUCIENDOSE EL PRIMER PAGE FAULT. EL PUNTERO DEL RELOJ APUNTA AL Frame #4 Y TIENE EL BIT DE USO EN CERO, ENTONCES SE PUEDE REEMPLAZARLA PREVIO GRABARLA EN DISCO PUES EL BIT DE MODIFICACIÓN ESTA EN UNO, O SEA QUE FUE MODIFICADA. SE ACTUALIZA LA TABLA:

NÚMERO DE FRAME:	4
NÚMERO DE PAGE;	14
NÚMERO DE BIT DE USO;	1
NÚMERO DE BIT DE MODIFICACIÓN:	0

Y EL PUNTERO DEL RELOJ APUNTA AL Frame #5

3er PEDIDO: PAGINA 15 (ESCRITURA) SE OBSERVA QUE LA PAGINA 15 NO ESTA EN MEMORIA, POR LO TANTO SE DEBE CARGARLA PRODUCIENDOSE EL SEGUNDO PAGE FAULT. SE PONE EL BIT DE USO DEL FRAME #5 EN CERO, EL BIT DE USO DEL FRAME #6 EN CERO Y CUANDO LLEGA AL FRAME#7 SE OBSERVA QUE SE LO PUEDE REEMPLAZAR PREVIO GRABARLA EN DISCO PUES EL BIT DE MODIFICACIÓN ESTA EN UNO, O SEA QUE FUE MODIFICADA Y SE CARGA LA PAGINA 15.

LA TABLA QUEDA:

NÚMERO DE FRAME:	0	1	2	3	4	5	6	7	EL PUNTERO QUEDA APUNTANDO AL FRAME #0
NÚMERO DE PAGE;	17	32	41	5	14	13	2	15	
NÚMERO DE BIT DE USO;	1	1	0	0	1	0	0	1	
NÚMERO DE BIT DE MODIFICACIÓN:	1	1	1	1	0	1	1	0	

Y EL PUNTERO DEL RELOJ APUNTA AL Frame #0

4to PEDIDO: PAGINA 2 (LECTURA) SE OBSERVA QUE LA PAGINA 2 ESTA EN MEMORIA, POR LO TANTO SE PUEDE ATENDER EL PEDIDO Y SE ACTUALIZA LA TABLA SOLO EL BIT DE USO

NÚMERO DE FRAME:	6
NÚMERO DE PAGE;	2
NÚMERO DE BIT DE USO;	1
NÚMERO DE BIT DE MODIFICACIÓN:	1

5to PEDIDO: PAGINA 18 (LECTURA) SE OBSERVA QUE LA PAGINA 18 NO ESTA EN MEMORIA, POR LO TANTO SE DEBE CARGARLA PRODUCIENDOSE EL TERCER PAGE FAULT. SE BUSCA CON EL PUNTERO DEL RELOJ DONDE SE LA PUEDE UBICAR LA TABLA ACTUAL ES:

NÚMERO DE FRAME:	0	1	2	3	4	5	6	7	EL PUNTERO QUEDA APUNTANDO AL FRAME #0
NÚMERO DE PAGE;	17	32	41	5	14	13	2	15	
NÚMERO DE BIT DE USO;	1	1	0	0	1	0	1	1	
NÚMERO DE BIT DE MODIFICACIÓN:	1	1	1	1	0	1	1	0	

EL #FRAME 0 TIENE EL BIT DE USO = 1 SE LO CAMBIA EL BIT DE USO = 0 Y SE AVANZA;
EL #FRAME 1 TIENE EL BIT DE USO = 1 SE LO CAMBIA EL BIT DE USO = 0 Y SE AVANZA;
EL #FRAME 2 TIENE EL BIT DE USO = 0 LO QUE INDICA QUE ESTA PAGINA SE PUEDE REEMPLAZAR POR LA PAGINA 18, ENTONCES SE GRABA LA PAGINA 41 POR ESTAR MODIFICADA Y SE CARGA LA PAGINA 18.

LA TABLA QUEDA:

NÚMERO DE FRAME:	0	1	2	3	4	5	6	7	EL PUNTERO QUEDA
NÚMERO DE PAGE;	17	32	18	5	14	13	2	15	

NÚMERO DE BIT DE USO;	1	1	1	0	1	0	1	1	APUNTANDO AL FRAME #3
NÚMERO DE BIT DE MODIFICACIÓN:	1	1	0	1	0	1	1	0	

6to PEDIDO: PAGINA 14 (ESCRITURA) LA PAGINA ESTA EN MEMORIA, POR LO TANTO SE PUEDE ATENDER EL PEDIDO Y SE ACTUALIZA LA TABLA:

NÚMERO DE FRAME:	4
NÚMERO DE PAGE;	14
NÚMERO DE BIT DE USO;	1
NÚMERO DE BIT DE MODIFICACIÓN:	1

FINALMENTE LA TABLA QUEDA:

NÚMERO DE FRAME:	0	1	2	3	4	5	6	7	EL PUNTERO QUEDA APUNTANDO AL FRAME #3
NÚMERO DE PAGE;	17	32	18	5	14	13	2	15	
NÚMERO DE BIT DE USO;	1	1	1	0	1	0	1	1	
NÚMERO DE BIT DE MODIFICACIÓN:	1	1	0	1	1	1	1	0	

Y SE REEMPLAZARON LAS PAGINAS:

NÚMERO DE PAGE:	7	20	41
NÚMERO DE FRAME:	4	7	2

Ejercicio RM5: 15

Un sistema posee 2KB de memoria física, la cual se administra mediante paginación por demanda. Se tiene una entrada de 12 bits para el direccionamiento lógico, de los cuales los 3 primeros determinan el número de página y la carga inicial no produce page fault. Determine, utilizando FIFO, la cantidad de Page Faults que produce la siguiente secuencia de referencias a memoria: **589h - 111h - 3B8h - 9FBh - ABCh - 824h - 159h**

Solución:

Direccionamiento:

# Página 3 bits	Offset 9 bits
-----------------	---------------

Cantidad de frames:

Los 9 bits del Offset determinan el tamaño de la página $2^9 = 512$ bytes

Con 2KB de memoria física (2048 bytes) poseemos $2048 / 512 = 4$ frames.

Secuencia:

Pasamos los números de referencia hexadecimales a su forma binaria. De dicho resultado, sólo nos interesan los tres primeros bits, que son los que determinan el número de página.

589 h = **0 1 0 1** 1 0 0 0 1 0 0 1 ----> Página **2**
 111h = **0 0 0 1** 0 0 0 1 0 0 0 1 ----> Página **0**
 3B8h = **0 0 1 1** 1 0 1 1 1 0 0 0 ----> Página **1**
 9FB h = **1 0 0 1** 1 1 1 1 1 0 1 1 ----> Página **4**
 ABCh = **1 0 1 0** 1 0 1 1 1 1 0 0 ----> Página **5**
 824h = **1 0 0 0** 0 0 1 0 0 1 0 0 ----> Página **4**
 159h = **0 0 0 1** 0 1 0 1 1 0 0 1 ----> Página **0**

Traza	2	0	1	4	5	4	0
Frame 1	2	2	2	2	5	5	5
Frame 2		0	0	0	0	0	0
Frame 3			1	1	1	1	1
Frame 4				4	4	4	4
Page Fault					F		

SE PRODUCE UN SÓLO FALLO DE PÁGINA si se considera que las primeras paginas no producen PF. Si se considera que las primeras paginas producen PF son 5.

Ejercicio RM5: 16

En un SO propietario se encuentra corriendo un único proceso que maneja a un determinado dispositivo.

* Supongamos que se tratase de una administración de **memoria paginada simple** donde la **pag 0** se corresponde con el **frame 0**, la **pagina 1** con el **frame 1**, y a si sucesivamente.

* Las **direcciones virtuales de 16 bits** se dividen en **9 de offset** y **7 bits** para el número de pagina.

- En el frame nro **0Ah** se encuentra el área de **código**.
- En el frame nro **1Eh** el área de **datos** (variables globales, etc).
- En el frame nro **3Ch** el área de **stack** (en nuestro caso variables locales a la función miFuncion(void *).

* **No se encuentran asignados otros frames al sistema**

Se pide responder :

a) La variable **int a**; que reside en la dirección física **3C08h** . Es una variable global o es local a la función **miFuncion(void *)** ?

b) En la dirección física **7834h** se encuentra declarado **int miVector[?]**. (un **int** = 2 bytes)

Hasta que tamaño podrá tener mi vector ? (Ayuda : Fijarse que no se asigna más de un frame a cada área, lo que en realidad no sería usar punteros lejanos).

RESOLUCIÓN:

a) Simplemente lo que pide es pasar de una dirección física a una dirección lógica y fijarse cual es el frame involucrado. Lo podemos ver en Hex o decimal.

3	C	0	8 h
0011	1100	0000	1000

Tomando 9 bits desde la derecha para el offset y 7 a la izquierda para el frame

Y reagrupando de a 4 bits :

001	1110	0	0000	1000
-----	------	---	------	------

1	E h	0	8 h
---	-----	---	-----

frame 1Eh (nro 30 dec) (que según el enunciado es el area donde residen las variables globales)

En decimal sería $3C08h = 8 + 16 \cdot 16^2 + 3 \cdot 16^3 = 15368$

$15368 \% 512 = 512 \cdot 30 + 8$ (30 dec es el frame y 8 el offset)

b) Primero averiguamos para la primera posición del vector cuya dirección física se da, en que frame se encuentra y en que offset del frame esta y luego cuanto le queda libre en el frame. Luego al espacio libre lo dividimos por el tamaño de cada elemento y nos da cuanto podría ocupar ese array.

7	8	3	4 h
0111	1000	0011	0100
011	1100	0	0011
3	C h :	3	4

$34h = 52$ dec

Luego si el tamaño del frame es 512

$512 - 52 = 460$

Como cada posición son dos bytes $\Rightarrow 460 / 2 = 230$ elementos

_____ ○ ○ ○ _____

Ejercicio RM5: 17 Ejercicio de traducción de direcciones de memoria

Considere un computador que utiliza paginación de nivel único. El tiempo medio de acceso a memoria principal es de 80 nseg. El procesador posee una TLB cuyo tiempo de acceso es de 10 nseg. ¿Cuál es la tasa de aciertos mínima que ha de tener la TLB para que el tiempo medio de acceso a un dato sea inferior a 100 nseg.?

Suponga que en este computador se instala un sistema de paginación por demanda. Se obtiene, con una tasa de aciertos de la TLB del 90% y un tiempo medio de acceso al disco de 10 mseg., que el tiempo medio de acceso es de 200 nseg. ¿Puede usted estimar la tasa de fallos de página a partir de estos dato

Resolución del Ejercicio

Datos del sistema:

* Tiempo de acceso a la memoria principal = 80 nseg.

* Tiempo de acceso a la TLB = 10 nseg.

* Tiempo de acceso al disco = 10 mseg.

Tasa de aciertos mínima de la TLB

Se quiere mantener un tiempo medio de acceso a un dato inferior a 100 nseg.

Si p es la tasa de aciertos de la TLB, se ha de cumplir:

$$(10 \text{ nseg.} + 80 \text{ nseg.})p + (10 \text{ nseg.} + 80 \text{ nseg.} + 80 \text{ nseg.})(1-p) < 100 \text{ nseg.}$$

Resolviendo la inequación, se obtiene que $p > 7/8 = 87.5\%$

(si se supone que el acceso a la TLB se hace concurrentemente con una operación de lectura con el bus, la tasa mínima obtenida es del **85.7%**)

Tasa de fallos de página

En las condiciones del problema, se tiene que:

* Tasa de aciertos de la TLB = 90%

* Tiempo medio de acceso a un dato = 200 nseg.

La tasa de fallos de página se puede estimar a partir de la expresión:

$$T_m(1-p) + T_f p = T_{\text{medio}}$$

donde

* T_m = tiempo de acceso a un dato en memoria principal = $0.9 \times 90 + 0.1 \times 170 = 98 \text{ nseg.}$

* T_f = tiempo de acceso a un dato cuando hay un fallo de página = 10 mseg. (aprox.) = 10^7 nseg.

* p = probabilidad de un fallo de página = *incógnita*

* T_{medio} = tiempo medio de acceso a un dato = 200 nseg.

Sustituyendo, se obtiene que $p = (T_{\text{medio}} - T_m) / (T_f - T_m) = 102 / 10^7 = 1.02 \times 10^{-5}$

Es decir, una tasa de fallos de página de uno por cada cien mil accesos.

**Ejercicio RM5: 18 Simulacro de gestión de memoria**

Dada la siguiente cadena de referencias a páginas (en hexadecimal):

A0, 90, A0, 100, A0, B5, A0, 90, 80, 90, E5, 100

Suponiendo que en sistema existen tres marcos de página, represente cómo actuarían estas dos políticas: a) Menos recientemente usada (LRU); b) algoritmo óptimo.

- Menos recientemente usada (LRU).

(A0,-,-), (A0,90,-), (90,A0,-), (90,A0,100), (90,100,A0), (100,A0,B5), (100,B5,A0), (B5,A0,90), (A0,90,80), (A0,80,90), (80,90,E5), (80,90,100)

Nota: Los marcos se representan ordenados de izquierda a derecha según el criterio LRU.

- Algoritmo óptimo.

(A0,-,-), (A0,90,-), (A0,90,-), (A0,90,100), (A0,90,100), (A0,90,B5), (A0,90,B5), (A0,90,B5), (A0,90,80)(*), (A0,90,80), (E5,90,80)(**), (E5,90,100) (***)

(*) Sustituir A0 también sería válido.

(**) Sustituir 90 o 80 también sería válido.

(***) Sustituir E5 o 90 también sería válido.

**Ejercicio RM5: 19**

Considere un sistema de paginación bajo demanda, con un disco de paginación que tiene un tiempo medio de acceso y transferencia de 5 mseg. Las direcciones se traducen por medio de una tabla de páginas en memoria principal, con un tiempo de acceso de 1 μseg. por acceso a memoria. Así, cada referencia a memoria a través de la tabla de página precisa dos accesos. Para mejorar este tiempo, se ha añadido una memoria asociativa que reduce el tiempo de acceso a una referencia a memoria, si las entradas en la tabla de página se encuentra en la memoria asociativa. Suponiendo que el 80% de los accesos se encuentran en la memoria asociativa y que para el resto, el 10% (o el 2% del total) generan fallos de página. ¿Cuál es el tiempo efectivo de acceso a memoria?

Respuesta:

El 80% representa el hit ratio. O sea, el 80% de los accesos se encuentra en la memoria asociativa (tarda 1 μseg).

El 20% restante se divide en:

2% : tiempo de 5000 useg. (genera page fault).

18%: tiempo de 2 μseg. (1 μseg. + 1 μseg. = 2 μseg. , no genera page fault).

El tiempo efectivo de acceso a memoria es:

$$0.8 * 1 \mu\text{seg.} + 0.18 * 2 \mu\text{seg.} + 0.02 * 5000 \mu\text{seg.} = 101.16 \mu\text{seg.}$$

Ejercicio RM5: 20

Un computador cuyos procesos tienen 1024 páginas en sus espacios de direcciones mantiene su tabla de página en memoria. El overhead requerido para leer una palabra de la tabla es de 500 nseg. Para reducir el overhead, el computador tiene una memoria asociativa que posee 32 registros, y puede realizar la lectura en 100 nseg. ¿Cuál será el hit ratio necesario para reducir el overhead a 200 nseg.?

Respuesta:

El overhead producido para leer una palabra de la tabla: 500 nseg.

Tiempo de lectura (del registro asociativo): 100 nseg.

$$200 \text{ nseg.} = \frac{X * 100 \text{ nseg.} + (100-X) * 500 \text{ nseg.}}{100} = X + (100 - X) * 5 =$$

$$= X + 500 - 5X = 500 - 4X \quad \Rightarrow$$

$$\Rightarrow 300 = 4X \Rightarrow X = 75 \rightarrow \text{es el hit ratio. O sea, 75\%}$$

Ejercicio RM5: 21

Si ha observado que el número de instrucciones ejecutadas entre fallas de páginas es directamente proporcional al número de marcos asignados a un programa. Si la memoria disponible se duplica, el intervalo entre fallas de páginas también se duplica. Supongamos que una instrucción normal tarda 1 microseg (μseg), pero si ocurre una falla de página, toma 2001 microseg. (o sea, 2 mseg para manejar la falla). Si un programa tarda 60 segundos para ejecutar, cuánto tiempo le llevará atender 15000 fallas de páginas, y cuanto le llevará ejecutar si hay casi el doble de memoria disponible?.

RESOLUCIÓN

El programa tarda 60 seg. En ejecutarse con 15000 fallos de página con el doble de memoria se tendrán 7500 fallos de páginas.

Por lo tanto tardaría: 60 seg. – 7500*2000 = 60 seg. – 15 seg. = 45 seg.

Ejercicio RM5: 22

Una computadora provee a cada proceso con un espacio de direccionamiento de 65536 bytes dividido en páginas de 4096 bytes. Un programa particular tiene un código de 32768 bytes, datos 16386 bytes y pila de 15879 bytes. ¿Entrará este programa en el espacio de direcciones? Si el tamaño de cada página fuera de 512 bytes, ¿entraría?. Recuerde que una página no puede contener partes de dos segmentos diferentes.

Con páginas de 4096 bytes:

- Para texto 32768 / 4096 = 8 páginas.
- Para datos 16386 / 4096 = 5 páginas.
- Para pila 15870 / 4096 = 4 páginas.

El total de páginas es de 17 pero solo hay 16 páginas (65536 / 4096).

Con páginas de 512 bytes:

Para texto 32768 / 512 = 64 páginas.

- Para datos 16386 / 512 = 33 páginas.
- Para pila 15870 / 512 = 31 páginas.

Por lo tanto se necesita en total 128 páginas y son exactamente las que se dispone.

Ejercicio RM5: 23 FINAL 07- 03-2010 Stallings

Considerar una cadena de referencia a páginas de un proceso con un working set de M frames, inicialmente todos vacíos. La cadena de referencia de páginas es de longitud P con N páginas distintas en ella. Para cualquier algoritmo de reemplazo de página:

- a) ¿Cuál es el número mínimo de page faults que pueden ocurrir?.
- b) ¿Cuál es el límite superior de page faults que pueden ocurrir?.

RESPUESTA:

Para 1 la solución es N, considerando que $M \geq N$, de esta forma los únicos page faults que ocurren son los de la carga inicial de cada una de las páginas.

Para el 2 la solución es P. En este caso el número máximo es P, inclusive si se supone el peor de los casos donde $M = 1$, la cantidad de page faults dependerán de la cadena de referencia que se tenga, si en ella no existen pedidos consecutivos de una misma página, entonces siendo $M = 1$, la cantidad de page faults será P, sino dependerá de la cadena de referencia. Si $M > 1$ no sólo va a depender de la cadena de referencia sino también de el algoritmo de reemplazo empleado.

Ejercicio RM5: 24 Final del 19 – 12 - 2009

Considere un sistema de memoria paginada en dos niveles. El tiempo de acceso a memoria principal es de 60 nseg. El tiempo de acceso a la TLB es de 5 nseg. ¿Qué tasa de aciertos mínima ha de tener la TLB para que el tiempo medio de acceso a memoria esté por debajo de los 75 nseg.?

Respuesta

El enunciado indica una memoria paginada en dos niveles. Esto significa que, sin ayuda de la TLB, cada acceso a un dato requiere tres accesos a memoria principal: tabla primer nivel, tabla segundo nivel, dato. Si suponemos que la TLB almacena pares página lógica/marco físico, cuando hay acierto en TLB, sólo hay que realizar un acceso a memoria principal. En ambos casos hay que consumir un tiempo para revisar la TLB. Si además consideramos despreciables los restantes tiempos invertidos, se tiene:

tiempo de acceso a memoria principal: $T_m = 60$ ns

tiempo de acceso a la TLB: $T_l = 5$ ns

probabilidad de acierto de la TLB: p

tiempo medio de acceso: $T < 75$ ns

acceso con fallo de TLB: $T_a = T_l + 3 \cdot T_m$

acceso con acierto de TLB: $T_b = T_l + T_m$

tiempo medio de acceso: $T = (1-p) \cdot T_a + p \cdot T_b$

Sustituyendo, se obtiene:

$$T = (1-p) \cdot (T_l + 3 \cdot T_m) + p \cdot (T_l + T_m) = T_l + 3 \cdot T_m - p \cdot 2 \cdot T_m = 185 \text{ ns} - 120 \text{ ns} \cdot p < 75 \text{ ns}$$

$$p > 110/120$$

$$p > 91,7\%$$

La tasa de aciertos de la TLB, por tanto, ha de ser superior al 91,7%.

EJERCICIOS SIN RESOLVER

Ejercicio M5: 1

A un sistema que trabaja con el método de administración de memoria de particiones variables le llegan los siguientes procesos para cargar, y cuenta con las siguientes particiones de memoria libres (no contiguas):

Proceso	Tamaño	Part. Libres
1	175	500
2	400	300
3	220	600
4	280	200
5	310	100

- Indique cómo quedaría la memoria (que proceso en que partición), con los siguientes de asignación Best Fit, Worst Fit, y First Fit.
- Qué procesos no entran (si los hay), y si compactando quedaría alguna partición en la que podría entrar alguno de los procesos que no pudieron ingresar.

○ ○ ○

Ejercicio M5: 2

Un sistema que trabaja con una memoria segmentada que se encuentra asignada de la siguiente manera:

Número de Segmento	Dirección de comienzo	Tamaño
0	3000	500
1	2500	300
2	4000	250
3	2000	350

Indicar cuales serían las direcciones físicas para las siguientes direcciones lógicas:

- 0,200
- 2,249
- 3,350
- 1,200
- 0,0

○ ○ ○

Ejercicio M5: 3

Un proceso hace referencia a cinco (5) páginas, A,B,C,D, y E, según la siguiente cadena de referencias: A ;B ;C ;D ;A ;B ;E ;A ;B ;C ;D ;E, Se pide:

- Asumiendo que el algoritmo de reemplazo de páginas es FIFO (First In, First Out), es el único proceso en memoria, y hay 3 y 4 frames, calcule los fallos de página que hay en cada uno de los casos.
- Efectúe el mismo análisis para las algoritmos de reemplazo de páginas LRU, y Óptimo.

○ ○ ○

Ejercicio M5: 4

Dado un sistema con un tamaño de página de 100 bytes, y la siguiente cadena de referencias: 0745 / 0054 / 0125 / 0256 / 0098 / 0378 / 0083 / 0487 / 0263 / 0321 / 0041 / 0325 / 0296 / 0137 / 0254 / 0062 / 0135 / 0784 / 0061 / 0152

Indique la tasa de fallos de páginas para cada uno de los siguientes métodos : FIFO, LRU, ÓPTIMO

○ ○ ○

Ejercicio M5: 5

Explique cuales de los siguientes métodos de administración de memoria son implementables y cuales no. En caso de que sean implementables se debe indicar el formato de la dirección lógica y de la física, y bajo que circunstancias serían útiles. Segmentación paginada bajo demanda.

- Segmentación paginada sin demanda.
- Paginación de dos niveles.
- Segmentación paginada de dos niveles.
- Segmentación de dos niveles.
- Doble segmentación paginada bajo demanda.

○ ○ ○

Ejercicio M5: 6

Considere un sistema de paginación, se pide :

- 1 Si el acceso a memoria tarda 70 ns. Cuanto durará una referencia a memoria. (Considere que la tabla de páginas está ubicada en la memoria, que ocupa en total cuatro páginas, y que para el promedio de las búsquedas hay que leer dos páginas de la tabla de páginas).
- 2 Si se añaden 8 registros asociativos y el 75% de las referencias se encuentran en ellos, Cual es el tiempo efectivo de referencia a memoria ? (El tiempo de búsqueda en la memoria asociativa es de 5 ns., y considere las mismas condiciones que el punto anterior para la lectura de la tabla de páginas.)

_____ ☐ ☐ ☐ _____

Ejercicio M5: 7

Suponga un esquema de memoria paginada con memoria virtual, en el cual se están ejecutando cuatro procesos, A, B, C y D, con longitudes de 4K, 2K, 1K y 3K respectivamente. La longitud de página es de 0.5 K

- 1 ¿Cuál es la longitud máxima de páginas que puede tener un programa si las direcciones son de 24 bits?
- 2 Diseñe las tablas necesarias para el sistema suponiendo que el tamaño de la memoria es de 64K.

_____ ☐ ☐ ☐ _____

Ejercicio M5: 8

Una computadora proporciona a cada proceso 65.536 bytes, de espacio de direcciones, dividido en páginas de 4.096 bytes. Un programa tiene un tamaño de texto de 32.768 bytes, un tamaño de datos de 16.386 bytes y una pila de 15.870 bytes. ¿ Cabe este programa en el espacio de direcciones ? Si el tamaño de página fuera de 512 bytes ¿cabría ?

_____ ☐ ☐ ☐ _____

Ejercicio M5: 9

Considere un sistema de memoria virtual con dos niveles de paginación que posee una dirección de 32 bits (tanto lógica como física), repartidos de la siguiente manera:

- 8 bits representan la entrada en la PD (Page Directory).
- 12 bits representan la entrada en la PT (Page Table).
- 12 bits representan el Offset.

Se pide que indique:

- a. Tamaño máximo direccionable en la memoria real (en KB/MB/GB – uno solo).
- b. Tamaño de la página (en KB/MB/GB – uno solo).
- c. Cantidad de páginas lógicas (en cantidad).
- d. Cantidad de páginas físicas (en cantidad).
- e. Si se posee una memoria real de 256 MB (libre para los procesos), y un área de swap de 512 MB. indique que pasaría al intentar cargar el proceso PG, que tiene un tamaño de código de 275,678 bytes, un área de datos de 205,296 bytes, y un área compartida de 36,963 bytes. Tenga en cuenta que el administrador de memoria centraliza carga las páginas en memoria central, y en el área de swap.
- f. Considere que sucedería con el proceso descrito anteriormente si el administrador de memoria solo utiliza el área de swap cuando se queda sin memoria real.

_____ ☐ ☐ ☐ _____

Ejercicio M5: 10

Un programa debe leer una tabla de una base de datos, que tiene 15000.- (quince mil) registros con la siguiente estructura toda a memoria (trabaja con paginación, y el tamaño de página es de 1024).

- Código (Double)
- Descripción (70 caracteres)
- Dirección (23 caracteres)
- Código postal (12 caracteres)

Se pide:

- g. Indique la cantidad de páginas necesarias para cargar todos los registros a memoria.
- h. Indique cual es el porcentaje de desperdicio (si lo hay), de la memoria.
- i. Que cambios se podrían realizar para disminuir el desperdicio a la mínima sin cambiar el tamaño de página, y recalculé los dos puntos anteriores.

_____ ☐ ☐ ☐ _____

Ejercicio M5: 11

Un programa lee una matriz de 128 columnas por 30 filas, con tipo de dato Integer (ocupa 2 bytes), de la siguiente manera (no se toman en cuenta los encabezados del programa, ni las definiciones de variables):

```
for Filas := 0 to 29
    for Columnas := 0 to 127
        write matrix ( Filas, Columnas )
    next;
next;
```

El tamaño de la memoria asignada donde debe ejecutar el proceso es de 512 bytes, divididos en páginas de 256 bytes. El área de código del proceso ocupa una página, y está siempre cargado en memoria, quedando la otra página libre para cargar los datos desde la memoria virtual.

Se pide que indique la cantidad de fallos de página que generará dicho proceso (toda carga de página deberá ser considerada fallo de página, sin importar como estaba la página anteriormente).

_____ ☐ ☐ ☐ _____

Ejercicio M5: 12

Considere un sistema de memoria virtual con dos niveles de paginación con direccionamiento de 32 bits repartidos de la siguiente manera :

- j. Los 10 últimos bits marcan la entrada en la PD (Page Directory).
- k. Los 10 siguientes marcan la entrada en la PT (Page Table).
- l. Los 12 primeros marcan el Offset dentro de la página.

Considere además que en este momento se encuentran corriendo en el sistema 254 procesos cuyos tamaños van de 4.2 K a 6.32 MB. Algunas características son :

- a. El 41.23% de estos procesos son claramente I/O Bound.
- b. El 56.86% de estos procesos son claramente CPU Bound.
- c. El 28.11% de estos procesos tienen interacción con el usuario.
- d. El 39.18% de estos procesos cooperan entre sí de a grupos de 12.
- e. El planificador de procesos es preemptive.

La política de asignación es global, y el algoritmo que se usa es LIFO. Este algoritmo funciona como una pila con tamaño máximo. El tamaño máximo va a ser el total de frames que haya en el sistema.

El espacio de memoria virtual total es de 58 MB, la memoria real es de 6 MB.

Se pide describir la situación exacta del sistema y las causas que llevan a que ninguno de los procesos hayan terminado después de 243 minutos de ejecución

_____ ☐ ☐ ☐ _____

Ejercicio M5: 13

Asumir que una tarea está dividida en 4 segmentos de igual longitud, y que el sistema construye una tabla de páginas de 8 entradas para cada segmento. Entonces el sistema tiene una combinación de segmentación y paginación. Asumir también que el tamaño de la página es de 2 Kbytes.

- a) Cuál es el tamaño máximo de cada segmento?
- b) Cuál es el máximo espacio lógico de direcciones para la tarea?
- c) Asumir que un elemento en una locación física con dirección 00021ABCh , es accedido por la tarea.
 - i. Cuál es el formato de la dirección lógica que la tarea generó?
 - ii. Cual es el máximo espacio físico de direccionamiento para el sistema?

_____ ☐ ☐ ☐ _____

Ejercicio M5: 14

Se tiene la siguiente cadena de referencia de un proceso X: 1-2-4-1-2-4-5-7-1-2-3-4-5-1-7-1-2-5-3-4-1-2-2-1. Sabiendo que el proceso dispone de 3 frames (inicialmente vacíos) y que el algoritmo de elección de la víctima es el LRU (Least Recently Used), se pide:

- 1. a. La cantidad de fallos de página y el estado de la memoria en cada instante de tiempo.
- 1. b. Realice el mismo ejercicio con el algoritmo FIFO. ¿Se produce la anomalía de Belady? Justifique.
- 1. c. Realice el mismo ejercicio con el algoritmo Clock, sabiendo que el puntero del clock se encuentra apuntando al primer frame y que ninguna página ha sido referenciada anteriormente.

_____ ☐ ☐ ☐ _____

Ejercicio M5: 15

Nos encontramos frente a un Sistema Operativo que utiliza paginación por demanda, con direcciones de 32 bits. A su vez, se tiene un proceso que genera la siguiente secuencia de referencias a memoria: 1100100 – 100000001 – 1001100011010 – 11110 – 110010000000 – 1010100011000 (Direcciones

lógicas en binario). Se sabe que el tamaño total de la memoria principal es de 4 KB (2 frames para el SO y el código) y 2 frames disponibles para el proceso, inicialmente vacíos. Considerando que el algoritmo para la elección de la víctima es el LRU (Least recently Used) con asignación fija y alcance local, se pide que indique la cantidad de fallos de página que producen las referencias del proceso y las traducciones de las direcciones lógicas a físicas. ¿Cuántos bits se necesitan para hacer referencia a la página? ¿Y para el offset?

Ejercicio M5: 16

Sea un Sistema de 1536 Kb de memoria libre, el cual utiliza el algoritmo Buddy System para la administración de la memoria. Teniendo en cuenta que las operaciones que va a realizar un proceso ordenadas en el tiempo en este Sistema son: a) asignación 234Kb, b) asignación 356Kb, c) asignación 45Kb, d) liberación 234Kb, e) asignación 110 Kb y f) liberación 45 Kb, se pide, asignar y diagramar por cada liberación / asignación las particiones asignadas y las libres.

¿Cuál es la fragmentación que se produce por cada una de las operaciones?

_____ ○ ○ ○ _____

Ejercicio M5: 17

Considere un sistema de memoria virtual con dos niveles de paginación con direccionamiento de 32 bits repartidos de esta manera:

- 10 últimos bits marcan la entrada en la PD (page directory).
- 10 siguientes marcan la entrada en la PT (page table).
- 12 primeros marcan el offset dentro de la página.
- 1 página es de 4 KBytes.

Se sabe además que en este momento se encuentran corriendo en el sistema 245 procesos cuyos tamaños van de 4.2KB a 6.32MB. Algunas características son:

- El 41.23% de estos procesos son claramente I/O Bound.
- El 56.86% de estos procesos son claramente CPU Bound.
- El 28.11% de estos procesos tienen interacción con el usuario.
- El 39.18% de estos procesos cooperan entre sí de a grupos de 12 procesos.
- El planificador de procesos es preemptive.
- La política de paginación es global y el algoritmo que se usa es el LIFO. Este algoritmo funciona como una pila con tamaño máximo. El tamaño máximo va a ser el total de marcos (frames) que haya en el sistema. Como sabemos cuando no quedan más marcos libres y se necesita una nueva página, acontece un fallo de página. Con este algoritmo la víctima va a ser la última página que entró.
- El espacio de memoria virtual total es de 58 MB. La memoria RAM es de 6 MB.

Se pide que describa en un máximo de 10 líneas la situación exacta del sistema y las causas que llevan a él después de 243 min de ejecución.

_____ ○ ○ ○ _____

Ejercicio M5: 18

Considerando el siguiente esquema de paginación – segmentación:

Tabla de Segmentos

Numero de segmento	Tabla de paginas
0	2
1	0
2	1

Tabla de Página 0

Numero de Pagina Virtual	Marco de Página	Presente?	Solo lectura?
0	10	S	N
1	17	N	N
2	89	S	N
3	90	S	N
4	29	S	N
5	47	N	N
6	55	S	N
7	32	S	N
8	36	S	N
9	9	S	N

Tabla de Página 1

Numero de Pagina Virtual	Marco de Página	Presente?	Solo lectura?
0	3	N	N
1	22	N	N
2	73	S	N
3	74	S	N
4	85	S	N
5	29	S	N
6	63	S	N
7	93	S	N
8	85	S	N
9	15	S	N
10	27	S	N
11	34	S	N

Tabla de Página 2			
Numero de Pagina Virtual	Marco de Página	Presente?	Solo lectura?
0	33	S	S
1	46	S	S
2	54	N	S
3	6	S	S
4	99	N	S
5	67	S	S
6	21	S	S

- Sólo Lecturas es seteado con N si la página no se puede escribir.
- Asumir, que el tamaño de la página es de 1000 bytes utilizando direcciones en base 10 (no es realista, pero hace los cálculos más sencillos). Así, la primera página de memoria virtual corre desde la dirección virtual 0 hasta la dirección 999. (Todos los accesos a memoria son con palabras de 4 bytes).
- El numero máximo de entradas a tabla de paginas es de 100 (0 a 99).
- Cada proceso es separado en a lo sumo 3 segmentos (un segmento para el código, datos alocados en el heap, y el stack)
- Si la pagina es invalida (Presente = N), asumir que el fallo de página causa que el sistema operativo cargue la pagina que causo el fallo en el numero de marco de pagina dado, por ejemplo: un acceso a la pagina virtual 2 en la pagina de tabla causa un fallo de pagina donde el sistema operativo carga la pagina virtual 2 en el marco de pagina 54.

Teniendo en cuenta los datos mencionados se pide responder:

- 1) ¿Cuántos dígitos enteros se necesitan para el número de segmento, número de página, y offset?
- 2) ¿A Qué dirección física acceden las siguientes direcciones virtuales? (Asumir que los dígitos faltantes son ceros a la izquierda). Indicar si el acceso genera un error (segmento inválido, página inválida o violación de protección) o un fallo de página, según corresponda.
 - a) Leer desde la dirección virtual 21333.
 - b) Escribir la dirección virtual 5345.
 - c) Leer desde la dirección virtual 1810627
 - d) Leer desde la dirección virtual 104806
 - e) Leer la dirección virtual 200097

○ ○ ○

Ejercicio M5: 19

Suponga que la tabla de páginas para los procesos que actualmente están ejecutando sobre el procesador es la siguiente. Todos los números son decimales, todos se numeran comenzando de cero, y todas las direcciones son direcciones físicas de memoria. La longitud de la página es 1024 bytes.

Virtual Page Number	Valid Bit	Reference Bit	Modify Bit	Page Frame Number
0	1	1	0	4
1	1	1	1	7
2	0	0	0	-
3	1	0	0	2
4	0	0	0	-
5	1	0	1	0

- a) Describa exactamente como, en general, una dirección virtual generada por la CPU es traducida a una dirección de memoria física.
- b) A que dirección física corresponderían cada una de las siguientes direcciones virtuales?
 - (i) 1052
 - (ii) 2221
 - (iii) 5499

○ ○ ○

Ejercicio M5: 20

Un proceso tiene asignado 4 frames. El tiempo en que se cargó por última vez la página a memoria, el tiempo en que se accedió por últimavez a la página en memoria, el número virtual de la página en cada frame , el referenced bit (R) y el modified bit (M) para cada frame se especifican a continuación:

Virtual Page Number	Page Frame Number	Time Loaded	Time Referenced	R Bit	M Bit
2	0	60	161	0	1
1	1	130	160	0	0
0	2	26	162	1	0
3	3	20	163	1	1

Ha ocurrido un **page fault** por la página virtual 4. A qué frame se le reemplazará su contenido para cada una de las siguientes políticas. Especifique por qué en cada caso.

- a) FIFO (first-in, first-out)
- b) LRU (least recently used)
- c) NRU (not used recently).
- d) Clock
- e) Optimal. (considere el string de referencia del punto siguiente(f))
- f) Dado el estado de memoria mencionado anteriormente luego del fallo de página, considere el siguiente string de referencia de páginas virtuales:
4, 0, 0, 0, 2, 4, 2, 1, 0, 3, 2

Cuántos fallos de página ocurrirán si se utiliza la política 'working set' con una longitud de ventana de 4 en lugar de una asignación fija? Muestre claramente cuando sucede cada page fault.

_____ ☐ ☐ ☐ _____

Ejercicio M5: 21

En un sistema de administración de memoria paginada por demanda:

¿Cuál es el tamaño máximo posible de un programa, despreciando la porción de sistema operativo residente, que ejecuta en un sistema de capacidad de direccionamiento de 16 bits, con 8 bits para número de página y de una memoria real de 32 K?

_____ ☐ ☐ ☐ _____

Ejercicio M5: 22

Suponga un esquema de memoria paginada con memoria virtual.

Se están ejecutando tres programas A, B y C con longitudes 2K, 1.5K y 3K respectivamente. La longitud de la página es de 0.5K.

- a) ¿Cuál es la cantidad máxima de páginas que puede tener un programa si las instrucciones tienen direcciones de 16 bits?
- b) Diseñe las tablas necesarias para el sistema suponiendo que el tamaño de la memoria es 64 K.
- c) Determinar el contenido de las tablas para los programas A, B y C. Los bloques de memoria contienen:: A-0, B-0, C-5, A-1, A-2, C-3, C-1, B-2, C-4.
- d) En la dirección 0280h de A hay una instrucción de bifurcación incondicional. Usando las tablas del punto c, determinar la dirección de memoria donde está la instrucción. Indique cómo actúa el sistema, si la instrucción bifurcara a: 029Ah, 00BAh, y 0708h.

_____ ☐ ☐ ☐ _____

Ejercicio M5: 23

Tenemos un sistema de computación con M1 frames de memoria central y una memoria auxiliar (disco) con capacidad para M2 páginas que llamaremos a,b,c,d, etc.

La traza de P es abacabdbacd.

Para cada algoritmo de reemplazo (FIFO, LRU, LFU) responda:

- A. para M1= 2 y M2=4
 - a) ¿cuál es el contenido de los M1 frames en memoria central?
 - b) calcular el índice de hallazgos (s=1-f)
- B. para M1=4 y M2=4
 - a) ¿cuál es el contenido de los M1 frames en memoria central?
 - b) calcular el índice de hallazgos (s=1-f)

_____ ☐ ☐ ☐ _____

Ejercicio M5: 24

Suponga un sistema de administración de memoria paginada por demanda, con la TDP en el procesador.

Una operación de falta de página tarda 8 mseg si no hay remoción y 20 mseg si hay remoción.

La velocidad de acceso a memoria es 1 microseg. Se sabe que el 70% de las operaciones de falta de página implican remoción.

Se pide:

- a) ¿Cuál es el máximo aceptable de paginación (páginas / segundo) si se desea que el sistema no incurra en Trashing?
- b) ¿Y si se desea que el sistema por lo menos dedique el 60% de su tiempo a procesar trabajos?

_____ ☐ ☐ ☐ _____

Ejercicio M5: 25

Diseñe las tablas necesarias para un sistema de administración de memoria por segmentación y memoria virtual.

Diseñar todas las tablas necesarias para la administración de memoria paginada por demanda, que permita una eficiente remoción de páginas de cualquier programa en función de las necesidades de uno dado.

_____ ☐ ☐ ☐ _____

Ejercicio M5: 26

Una máquina tiene 48 bits de espacio de direccionamiento virtual y 32 bits de espacio de direccionamiento físico. Las páginas son de 8K. ¿Cuántas entradas son necesarias para una tabla de páginas convencional? ¿Para una tabla de páginas invertidas?

Ejercicio M5: 27

Consideremos la siguiente cadena de referencias de páginas:

1-2-3-4-2-1-5-6-2-1-2-3-7-6-3-2-1-2-3-6

¿ Cuántos fallos de página se producirán con los algoritmos de reemplazo siguientes:

LRU

FIFO

OPTIMO

Suponiendo 1,2,3,4,5,6 ó 7 celdas (page frames)? Recuerde que todas las celdas están inicialmente vacías de modo que la primera referencia a una página supondrá un fallo cada una.

Ejercicio M5: 28

Considere un sistema de paginación bajo demanda, con un tambor de paginación que tiene un tiempo medio de acceso y transferencia de 5 mseg. Las direcciones se traducen por medio de una tabla de páginas en memoria central, con un tiempo de acceso de 1 useg por cada acceso a memoria. Así, cada referencia a memoria a través de la tabla de páginas precisa dos accesos. Para mejorar este tiempo, se ha añadido una memoria asociativa que reduce el tiempo de acceso a una referencia a memoria, si la entrada en la tabla de páginas se encuentra en la memoria asociativa. Suponiendo que el 80% de los accesos se encuentran en la memoria asociativa y que para el resto, el 10% (ó el 2% del total) generan fallos de página, ¿cuál es el tiempo efectivo de acceso a memoria?

Ejercicio M5: 29

Supongamos un sistema con una memoria de núcleos de 1 useg y un sistema de almacenamiento secundario en tambor con un tiempo de latencia de 5 useg y una velocidad de transferencia de 1 millón de palabras por segundo.

Para un tamaño de página p y una tasa de fallos de página de x ¿cuál es el tiempo efectivo de acceso?

Supongamos que la tasa de fallos de página varía inversa y exponencialmente con el tamaño de página, $x=e^{-p/500}$. Así cuanto mayor sea el tamaño de página menor será la tasa de fallos de página. Que tamaño de página da el mínimo tiempo efectivo de acceso ?.

Ejercicio M5: 30

Supongamos que tenemos una memoria con paginación bajo demanda. La tabla de páginas se mantiene en registros. Servir un fallo de página lleva 8 mseg. si hay disponible una página vacía o la página reemplazada no ha sido modificada y 20 mseg. si la página ha sido modificada. El tiempo de acceso a memoria es de un useg. Suponiendo que la página ha reemplazar esta sucia (modificada) el 70 % del tiempo, ¿cuál es la tasa de fallos de página máxima aceptable para un tiempo efectivo de acceso de no más de 2 useg.

Ejercicio M5: 31

Supongamos un sistema de paginación con un tambor de paginación de 4 millones de palabras y con un tiempo de transferencia de 5 mseg. y una memoria de núcleos paginada de 262144 palabras, con un tiempo de acceso 2 useg. Si queremos que nuestro sistema de paginación se presente ante el usuario como una memoria de 4 millones con un tiempo de acceso de 4 useg. (en promedio), ¿qué porcentajes de accesos tienen que realizarse sin incurrir en fallos de página?

Ejercicio M5: 32

Un programa debe leer una tabla de una base de datos, que tiene 20000.- (veinte mil) registros con la siguiente estructura toda a memoria (trabaja con paginación, y el tamaño de página es de 1024).

- Código (Double)
- Descripción (60 caracteres)

- Dirección (35 caracteres)
- Código postal (8 caracteres)

Se pide:

- Indique la cantidad de páginas necesarias para cargar todos los registros a memoria.
- Indique cual es el porcentaje de desperdicio (si lo hay), de la memoria.
- Que cambios se podrían realizar para disminuir el desperdicio a la mínima sin cambiar el tamaño de página, y recalculé los dos puntos anteriores.

Ejercicio M5: 33

Considere un sistema de memoria virtual con dos niveles de paginación que posee una dirección de 32 bits (tanto lógica como física), repartidos de la siguiente manera:

- 8 bits representan la entrada en la PD (Page Directory).
- 12 bits representan la entrada en la PT (Page Table).
- 12 bits representan el Offset.

Se pide que indique:

- Tamaño máximo direccionable en la memoria real (en KB/MB/GB – uno solo).
- Tamaño de la página (en KB/MB/GB – uno solo).
- Cantidad de páginas lógicas (en cantidad).
- Cantidad de páginas físicas (en cantidad).
- Si se posee una memoria real de 256 MB (libre para los procesos), y un área de swap de 512 MB. indique que pasaría al intentar cargar el proceso PG, que tiene un tamaño de código de 275,678 bytes, un área de datos de 205,296 bytes, y un área compartida de 36,963 bytes. Tenga en cuenta que el administrador de memoria centrar carga las páginas en memoria central, y en el área de swap.

Considere que sucedería con el proceso descrito anteriormente si el administrador de memoria solo utiliza el área de swap cuando se queda sin memoria real

Ejercicio M5: 34

Dado un sistema con un tamaño de página de 100 bytes, y la siguiente cadena de referencias :

0745 / 0054 / 0125 / 0256 / 0098 / 0378 / 0083 / 0487 / 0263 / 0321 / 0041 / 0325 / 0296 / 0137 / 0254 / 0062 / 0135 / 0784 / 0061 / 0152

- Indique la tasa de fallos de páginas para cada uno de los siguientes métodos : FIFO, LRU, ÓPTIMO
- Indique cual es la imposibilidad para llevar a la práctica el método Óptimo, y cual es el motivo de su existencia.

Ejercicio M5: 35

Un proceso hace referencia a cinco (5) páginas, A,B,C,D, y E, según la siguiente cadena de referencias : A ;B ;C ;D ;A ;B ;E ;A ;B ;C ;D ;E, Se pide :

- Asumiendo que el algoritmo de reemplazo de páginas es FIFO (First In, First Out), es el único proceso en memoria, y hay 3 y 4 frames, calcule los fallos de página que hay en cada uno de los casos.
- Si nota algo fuera de los común, en los resultados anteriores, indique que es, y como se lo denomina.
- Efectúe el mismo análisis para las algoritmos de reemplazo de páginas LRU, y Óptimo.

Ejercicio M5: 36

Considere un sistema de swapping, en el que la memoria consta de los siguientes tamaños de espacios en el siguiente orden: 10K, 4K, 20K, 18K, 7K, 9K, 12K, 15K. ¿ En cuál de las particiones se ubicarán cada uno de los siguientes procesos ?, y ¿ cuál es el desperdicio ? : a = 12K, b = 10K, c = 9K. Si los métodos de asignación son : First Fit, Best Fit, Worst Fit, Next Fit.

Ejercicio M5: 37

En un determinado sistema que administra la memoria por un método de paginación bajo demanda, están ejecutándose 3 procesos, que hacen referencia a las páginas de la siguiente manera:

Se indica proceso, página, y operación realizada:

P0, L, 0 / P1, L, 0 / P0, E, 1 / P2, L, 0 / P0, L, 0 / P0, L, 2 / P1, E, 1 / P2, E, 3 / P0, L, 0 / P2, L, 1 / P2, L, 0 / P2, E, 3 / P2, E, 1

Sabiendo que el tiempo de direccionamiento de una dirección utilizando la TLB es de 10 ns (la TLB tiene dos registros), que el tiempo de direccionamiento sin usar la TLB (no se encontró en ella), toma 50 ns, y que el tiempo de Swap-In / Swap-out es de 20 ms, se pide:

1. Realizar el análisis de reemplazo de páginas, indicando en cada uno de los accesos el tiempo demorado en hacer toda la operación de direccionamiento, y si hay o no fallo de página. Considere cualquier acceso a memoria virtual como fallo de página, y el loader no carga nada en memoria.
2. Efectúe el cálculo anteriormente mencionado para los métodos de reemplazo de páginas FIFO, y LRU, considerando que la asignación es global.

Ejercicio M5: 38

Un sistema trabaja con segmentación con dos niveles de paginación bajo demanda, y tiene 10 bits de la dirección lógica destinados para el número de segmento, una cantidad máxima de 4096 tablas de página por segmento (primer nivel), de los cuales 4 dígitos en base hexadecimal son para el desplazamiento, y 26 dígitos en base binaria son para el número de página (segundo nivel), se pide:

- a. Tamaño del bus de direcciones del hardware (en bits).
- b. Tamaño máximo direccionable en la memoria.
- c. Tamaño de la página (en KB/MB/GB).
- d. Cantidad máxima de segmentos por proceso.
- e. Cantidad de entradas máximas que podrá tener la tabla de páginas de segundo nivel (en cantidad).

Ejercicio M5: 39

Dado un programa con 6 páginas, numeradas 0,1,2,3,4,5 que ejecuta dentro de un sistema operativo con administración paginada por demanda, con tres bloques, numerados 0,1,2, se pide:

- a) mostrar los reemplazos de páginas efectuados para ejecutar el programa, usando el algoritmo LRU para la traza T1=012452345015.
- b) Calcular el índice de fallas

Ejercicio M5: 40

Suponga un sistema operativo multiusuario, multiprocesamiento, con un esquema de memoria real de 2 Mb y memoria virtual de 32 Mb, con espacio de disco de 1 Gb. Los programas que se ejecutan en este sistema son de tipo variado, algunos escritos siguiendo las reglas de la programación estructurada, muchos batch y algunos (más bien bastantes) llenos de hermosos GOTO. Si Ud. tuviera que diseñar el método de administración de memoria, ¿elegiría Paginación por Demanda o Segmentación?. Justifique su respuesta, argumentando ventajas y desventajas de ambos métodos PARA ESTA IMPLEMENTACION PARTICULAR.

Ejercicio M5: 41

Un programa hace la siguientes referencias a sus segmentos y páginas durante su vida con el siguiente objetivo:

- | | | |
|------------------|---|------------------|
| ▪ Lectura-S0-P0 | | ▪ Lectura -S1-P0 |
| ▪ Lectura -S1-P0 | | ▪ Escribe -S2-P1 |
| ▪ Escribe-S2-P0 | | ▪ Lectura -S1-P0 |
| ▪ Lectura -S0-P0 | | ▪ Lectura -S2-P1 |
| ▪ Lectura -S2-P0 | | ▪ Lectura -S0-P0 |
| ▪ Escribe -S0-P1 | → | ▪ Escribe -S2-P2 |

El proceso cuenta con 3 frames asignados.

Se pide que efectúe el diagrama de uso de páginas y que indique la cantidad de fallos para los siguientes métodos. Considerando que una operación de Swap-In demora 30 ms y una operación de Swap-Out demora 60 ms.

Realice el cálculo de tiempos totales para cada método

- a) FCFS
- b) Segunda Oportunidad Mejorada

Ejercicio M5: 42

En un sistema que administra la memoria con una paginación bajo demanda, con asignación estática de frames, y política de reemplazo local (se asignan 5 frames por proceso, asignando 4 para el área de códigos y datos, y una para el área de stack). El tamaño de la página es de 512 By. El loader carga la página principal del proceso en memoria.

El tiempo en realizar un fallo de página sin guardar la víctima es de 20 ms., en cambio si hay que guardar la víctima el tiempo se triplica.

En este sistema se quieren ejecutar dos programas que definen una matriz de datos de 4 filas por 8 columnas de datos tipo float (ocupa 64 Bytes), y un vector de 8 posiciones también del tipo float. (en forma secuencial)

- Un proceso que su área de código ocupa 0.3 KB, que sumaria cada una de las filas guardando el resultado en el vector. El método de selección de víctima es FIFO.
- Un proceso que su área de código ocupa 0.4 KB, que lee la matriz, y calcula el total de cada una de las columnas, guardándolo en el vector (columna 0 de la matriz en la posición 0 del vector y así sucesivamente). El método de selección de víctima es LRU.

Se pide:

- Efectúe un gráfico con el contenido de cada una de las páginas en las que se dividirá cada proceso
- Arme la traza de referencia de páginas que generará cada proceso al ejecutarse
- Realice el cálculo del total de fallos de página para cada proceso y calcule el tiempo que se necesita para hacer todos los cambios de páginas.

Ejercicio M5: 43

Un programa debe leer una tabla de una base de datos, que tiene 20000.- (veinte mil) registros con la siguiente estructura (toda en memoria que trabaja con paginación, y el tamaño de página es de 1024 bytes).

Código (Double= 8 bytes)
 Descripción (60 caracteres)
 Dirección (35 caracteres)
 Código postal (8 caracteres)

Se pide:

- Indique la cantidad de páginas necesarias para cargar todos los registros en memoria.
- Indique cual es el porcentaje de desperdicio (si hay), de la memoria.
- Que cambios se podrían realizar para disminuir el desperdicio al mínimo sin cambiar el tamaño de página, y recalcule los dos puntos anteriores. (Todas las respuestas deben estar justificadas):

Ejercicio M5: 44

Considere un sistema de memoria virtual con dos niveles de paginación con direccionamiento de 32 bits repartidos de la siguiente manera :

- Los 10 últimos bits marcan la entrada en la PD (Page Directory).
- Los 10 siguientes marcan la entrada en la PT (Page Table).
- Los 12 primeros marcan el Offset dentro de la página.

Considere además que en este momento se encuentran corriendo en el sistema 254 procesos cuyos tamaños van de 4.2 K a 6.32 MB. Algunas características son :

- El 41.23% de estos procesos son claramente I/O Bound.
- El 56.86% de estos procesos son claramente CPU Bound.
- El 28.11% de estos procesos tienen interacción con el usuario.
- El 39.18% de estos procesos cooperan entre sí de a grupos de 12.

El planificador de procesos es preemptive.

La política de asignación es global, y el algoritmo que se usa es LIFO. Este algoritmo funciona como una pila con tamaño máximo. El tamaño máximo va a ser el total de frames que haya en el sistema. El espacio de memoria virtual total es de 58 MB, la memoria real es de 6 MB.

Se pide describir la situación exacta del sistema y las causas que llevan a que ninguno de los procesos hayan terminado después de 243 minutos de ejecución

Módulo 6: Administración de Entrada / Salida.**ENTRADA-SALIDA**

Con respecto a E/S, o su término en inglés (I/O), lo que usualmente se suele pedir es el tiempo que se tarda en atender a los pedidos y el orden en que éstos fueron atendidos. Para poder calcular esto, necesitamos varios datos: el tiempo que tarda la cabeza en pasar de una pista a otra (o de cilindro a cilindro), el tiempo que tarda en pasar de sector a sector y el tiempo de transferencia.

- **Tiempo de búsqueda:** Es el tiempo que tarda en pasar de cilindro a cilindro, o de pista a pista. Usualmente este dato viene en el problema.
- **Tiempo rotacional:** Es el tiempo que tarda en pasar de sector a sector. Suele venir como dato, o suele informar las revoluciones del disco. En este último caso, supongamos que nos dicen que el disco gira a 6000 rpm. Entonces, con una regla de tres simple podemos calcular cuanto tarda en dar una vuelta:

$$\begin{array}{rcl} 6000 \text{ vueltas} & \underline{\hspace{2cm}} & 1 \text{ min} \\ 1 \text{ vuelta} & \underline{\hspace{2cm}} & 0.0016 \text{ min} = 0.01 \text{ seg} = 10 \text{ ms} \end{array}$$

Si una pista tiene, por ejemplo, 10 sectores, entonces dividimos el tiempo que tarda en dar una vuelta (pasar los 10 sectores) por la cantidad de sectores y nos da el tiempo que tarda en pasar de sector a sector. En nuestro caso: $10 \text{ ms} / 10 \text{ sect.} = 1 \text{ ms/sector}$.

- **Tiempo de transferencia:** Cuando se informa la capacidad de los sectores, debemos calcular el tiempo que le lleva a la cabeza leer la información del sector; lo que se denomina tiempo de transferencia, y se calcula como:

$$T = b / r * N$$

b = cant. de bytes a transferir
 r = revoluciones por segundo
 N = cant. de bytes por pista

Muy bien, conocemos como calcular los tiempos, lo que necesitamos ahora es saber cuantos cilindros se recorren de pedido en pedido para poder aplicarle el tiempo de búsqueda y cuantos sectores hay que recorrer para atender el pedido. Para poder hacer esto, necesitamos tener las direcciones físicas de los pedidos, las cuales suelen venir expresadas como direcciones lógicas. Convertir una dirección lógica a física representa saber que cilindro, que sector y que cabeza lo lee. Para ello necesitamos conocer la cantidad de cilindros, sectores y cabezas (o platos) que tenemos.

Volviendo nuevamente a los 10 sectores por pista, y que también tenemos 50 cilindros y 3 platos (lo que serían 6 cabezas), primero debemos realizar las siguientes cuentas:

$$\text{Sectores por plato} \times \text{cant. de platos} = 20 \times 3 = 60$$

Recordemos que de un lado del plato tenemos 10 sectores, pero en la cara opuesta tenemos 10 sectores mas, por lo tanto, en un plato tenemos 20 sectores. Luego, también calculamos:

$$\text{Sectores} \times \text{pista} = 10 \quad (\text{ya lo sabíamos por dato})$$

Una vez calculado esto, podemos comenzar a convertir las direcciones lógicas a físicas. Supongamos la dirección lógica 1305. Lo primero que hacemos es dividir la dirección por los sectores por plato \times cant. de platos:

$$1305 / 60 = \text{Esta división da un cociente (C), del cual se toma la parte entera y un resto (R)}$$

C = representa el cilindro de la dirección

El resto, dado por "R", lo divido nuevamente, pero ahora por los sectores por pista:

$$R / 10 = \text{Esta división da un nuevo cociente (C'), del cual se toma la parte entera y un resto (R')}$$

C' = representa la cabeza que lee a la dirección

$R' + 1$ = representa el sector

El DOS hace esta conversión: El número de sector lógico se numera a partir del cero. Los sectores BIOS (para distinguirlos de los lógicos) se numeran a partir del 1.

Volviendo a nuestra dirección 1305, si la dividimos por 60:

$$C = 21$$

$$R = 45$$

Dividimos $R (=45)$ por 10:

$$C' = 4$$

$$R' = 5$$

$$\Rightarrow 1305 = (21, 4, 6) \quad \text{siendo (cilindro, cabeza, sector)}$$

Ahora ya tenemos las direcciones físicas y los tiempos calculados, solamente nos falta ver cuanto se tarda en pasar de una dirección a la otra. Para ello vamos a suponer dos direcciones lógicas más: 469 y 1040 cuyas direcciones físicas son (7,4,10) y (17,2,1) respectivamente. Entonces supongamos que se atienden los pedidos en ese orden:

DIR:	1305	469	1040
CIL:	21	7	17
SECTOR:	6	10	1

* **1305 → 469:** Supongamos que la dirección 1305 fue un pedido que se atendió, por lo tanto, cuando la cabeza leyó ese sector quedó al final del sector 6 (comienzo del 7). Para pasar a la dirección 469 debe atravesar 14 cilindros ($21 - 7 = 14$), que si lo multiplicamos por 1 ms que tarda de pasar de pista a pista, nos da un total de 14 ms. Ahora bien, durante esos 14 ms, el disco siguió girando, y los sectores siguieron pasando. Si tardé 14 ms y tardo 1 ms en pasar un sector, significa que pasé 14 sectores (es decir, una vuelta y 4 sectores más). Como me encontraba al principio del sector 7, cuando pegó una vuelta (pasó 10 sectores) quedó en el mismo lugar, pero como tardé 4 ms más (4 sectores más) me muevo hasta el comienzo del sector 1. Pero como el pedido de la dirección 469 está en el sector 10, y yo estoy al comienzo del sector 1, debo volver a pegar toda la vuelta (lo que sería el tiempo rotacional de búsqueda del sector) hasta ponerme al comienzo del sector 10, lo cual me lleva 9 ms (tengo que atravesar 9 sectores). Entonces, lo que tarde en pasar de la dirección 1305 a la 469 fueron $14 \text{ ms} + 9 \text{ ms} = 23 \text{ ms}$. Para atender este pedido, leo el 10º sector y la cabeza queda nuevamente al comienzo del sector 1.

* **469 → 1040:** Ahora me encuentro al comienzo del sector 1 ya que acabo de atender el pedido anterior; para atender el siguiente pedido realizo el mismo análisis que antes. Tengo que atravesar 10 cilindros ($17 - 7 = 10$), cada uno me lleva 1 ms, entonces tardo 10 ms. Durante esos 10 ms el disco siguió girando; como tardo 1 ms en pasar de sector a sector, esos 10 ms me hicieron pegar una vuelta entera (10 sectores) y volví a quedar nuevamente al comienzo del sector 1. Como el pedido que tengo que atender está en el sector 1, y me encuentro al inicio de éste, ya estoy listo para leerlo por lo que el tiempo rotacional es nulo. Por lo tanto, el tiempo que me llevó atender este pedido fueron solamente 10 ms, que si los sumo a los 23 ms del pedido anterior, el tiempo total para atender todos los pedidos fue de 33 ms.

Este es el análisis que se debe realizar cuando se deben atender pedidos. Por un tema de simplicidad no consideramos aquí el tiempo de transferencia, es decir, si tengo el tamaño en bytes de los sectores, cuando llego al comienzo del sector solicitado, debo atender el pedido. El tiempo que tarda esta atención se calcula con la fórmula que expresé al comienzo de la explicación de E/S, y que está denominada como "Tiempo de transferencia". Por supuesto que la suma de todos los tiempo (lo que tardé en atender cada uno de los pedidos y lo que tardo en transferirlos) me va a dar el tiempo total de atención.

Puede existir el caso en que simplemente se tenga como dato el tiempo que se tarda en pasar de una pista a la otra y se pida el tiempo promedio de atención. En tal caso, se calcula cuantas pistas se atravesaron entre pedido y pedido (como lo calculamos anteriormente, realizando la resta de los cilindros y multiplicando por el tiempo que se tarda de pista a pista), se suman todos estos tiempos y se divide por la cantidad de pedidos.

Otro punto a tener en cuenta en las solicitudes de E/S es en el orden en que se atienden. Para ello existen diferentes algoritmos que se detallan a continuación:

- **FIFO:** Como siempre, tenemos un algoritmo FIFO. Los pedidos se atienden en el orden en que llegan.
- **Shortest Seek Time First:** Se atiende al pedido que mas cercano está a la posición actual del brazo; en otras palabras, el que hace un menor desplazamiento del brazo. Puede producir inanición de aquellos pedidos que están en direcciones alejadas.
- **Scan:** Va “escaneado” las pistas del disco. Es decir, atiende a los pedidos moviéndose en una dirección hasta que llega al límite del disco, cuando no hay mas pistas, comienza a atender los pedidos en la dirección contraria.
- **C-scan:** Es igual que el anterior, pero no sube y baja sino que siempre atiende los pedidos en una dirección. Cuando llega al final, baja (o sube) abruptamente hasta el extremo contrario y vuelve a atender los pedidos en la misma dirección. El tiempo que le lleva realizar el retroceso se considera despreciable.
- **Look:** Es igual que el Scan, pero no llega hasta el final del disco para comenzar a moverse en la dirección contraria sino que cambia de dirección cuando atiende el último pedido en ese sentido. Supongamos que el brazo está ascendiendo, cuando atiende el último pedido en esta dirección, entonces comienza a descender y atender solicitudes en esta nueva dirección hasta que llega a la última y vuelve a cambiar de sentido.
- **C-Look:** Es igual que el C-scan, pero utiliza el mismo criterio que el Look. Cuando atendió el último pedido en una dirección, desciende (o asciende) abruptamente hasta la primera dirección a atender en el sentido que utiliza. Para dejarlo más claro, supongamos que está atendiendo una solicitud en la dirección 150, y ésta es la última en dirección ascendente, entonces baja abruptamente hasta el próximo pedido “más bajo” (supongamos que hay un pedido en la dirección 15) y a partir de este pedido comienza a ascender nuevamente.
- **N-scan:** Es el algoritmo Scan pero no se tiene una única cola para encolar los pedidos, sino que se tienen N colas. Cuando se están atendiendo los pedidos de una cola, los nuevos que lleguen NO pueden ingresar a la cola en uso, se deben encolar en alguna de las otras colas. Cuando se atendieron todos los pedidos de una lista, se pasa a atender los pedidos de otra con el mismo criterio (no se pueden encolar nuevos pedidos en esta nueva cola atendida)
- **F-scan:** Es un caso particular del N-scan; solo que tiene 2 colas. Los criterios a tener en cuenta son los mismos: cuando se atiende una cola, los nuevos pedidos se encolan en la otra.

EJEMPLOS DE EJERCICIOS RESUELTOS:

Ejercicio RM6: 1

Suponga que un driver de disco recibe peticiones de bloques de disco para los cilindros 2, 35, 46, 23, 90, 102, 13 y 34, estrictamente en ese orden y en intervalos de 7 ms. exactos (es decir, entre cada dos peticiones consecutivas transcurren exactamente 7 ms.).

El disco tiene 150 cilindros y gira a 3000 rpm, el tiempo de búsqueda entre cilindros consecutivos es 0,1 ms. (es decir, el tiempo de búsqueda es igual a $0,1 \cdot C$ ms., siendo C el número de cilindros recorridos) y el tiempo de transferencia real de cada bloque es un valor constante e igual a 5 ms.

Suponiendo que la cabeza de lectura/grabación se encuentra inicialmente posicionada en el cilindro 75, calcule la suma de los tiempos de acceso a cada bloque de disco para los algoritmos de planificación del brazo del disco SSTF y C-SCAN (variante del ascensor o SCAN) ascendente.

Solución:

La clave de este ejercicio consiste en aplicar el algoritmo correspondiente (SSTF o C-SCAN ascendente) sólo entre las demandas que hayan llegado en cada momento. Así, si consideramos que la petición para un bloque de disco del cilindro 2 llega en el instante 0, tendremos que aplicar el algoritmo sólo para esa demanda (resulta evidente que, sea cual sea el algoritmo que se aplique, siempre se atenderá en primer lugar la demanda del cilindro 2). Ante esta premisa, para ambos algoritmos tenemos que comenzar calculando el tiempo de acceso al bloque de disco del cilindro 2:

$T_{\text{acceso}} = T_{\text{búsqueda}} + \text{Retraso Rotacional} + T_{\text{transferencia}} = 7,3 \text{ ms.} + 10 \text{ ms.} + 5 \text{ ms.} = 22,3 \text{ ms.}$

$T_{\text{búsqueda}} = 0,1 \cdot C \text{ ms.} = 0,1 \cdot (75 - 2) \text{ ms.} = 0,1 \cdot 73 \text{ ms.} = 7,3 \text{ ms.}$

$\text{Retraso Rotacional} = \frac{1}{2} \cdot T_{\text{giro}} = \frac{1}{2} \cdot 20 \text{ ms.} = 10 \text{ ms.}$

$T_{\text{transferencia}} = 5 \text{ ms.}$

$T_{\text{giro}} = 20 \text{ ms.}$ (obtenido mediante regla de 3, sabiendo que el disco da 3000 vueltas en 1 minuto)

(Para las siguientes demandas, no se volverán a calcular el Retraso Rotacional y el $T_{\text{transferencia}}$, puesto que tendrán exactamente estos mismos valores)

En los 22,3 ms. que tarda en atenderse la demanda de un bloque del cilindro 2, da tiempo a que lleguen las 3 siguientes demandas (a los 7 ms. llega la demanda del cilindro 35, a los 14 ms. llega la del cilindro 46 y a los 21 ms. llega la del cilindro 23). Esto significa que habrá que aplicar el algoritmo únicamente para estas tres demandas.

Así, tanto el algoritmo SSTF como el C-SCAN ascendente, seleccionarán la demanda del cilindro 23 como la próxima a atender (en el primer caso, por ser la más cercana al cilindro 2, y en el segundo, por ser la próxima en sentido ascendente), siempre teniendo en cuenta que los algoritmos sólo se aplican a las demandas que, hasta ese momento, hayan llegado al sistema.

Calculemos, por tanto, el tiempo de acceso al bloque de disco del cilindro 23:

$T_{\text{acceso}} = T_{\text{búsqueda}} + \text{Retraso Rotacional} + T_{\text{transferencia}} = 2,1 \text{ ms.} + 10 \text{ ms.} + 5 \text{ ms.} = 17,1 \text{ ms.}$

$T_{\text{búsqueda}} = 0,1 \cdot C \text{ ms.} = 0,1 \cdot (23 - 2) \text{ ms.} = 0,1 \cdot 21 \text{ ms.} = 2,1 \text{ ms.}$

Cuando termina de atenderse la demanda del cilindro 23, habrán transcurrido exactamente 39,4 ms. (los 22,3 ms. que tarda en atenderse la demanda del bloque de disco del cilindro 2 más los 17,1 ms. que tarda en atenderse la demanda del bloque de disco del cilindro 23) desde el instante 0 que se consideró al principio del ejercicio. Eso significa que habrán llegado al sistema las 5 peticiones siguientes a la primera (es decir, dos nuevas sobre las que habían llegado en la anterior aplicación del algoritmo).

Tendremos, por tanto, que aplicar el algoritmo correspondiente sobre las peticiones de los cilindros 35, 46, 90 y 102 (hay que tener en cuenta que las demandas a los cilindros 2 y 23 ya han sido atendidas).

Ante esta situación, los algoritmos SSTF y C-SCAN ascendente volverán a seleccionar la misma demanda (concretamente la del cilindro 35) como la próxima a atender, por ser la más próxima a la posición actual de la cabeza de lectura/grabación (en el caso del algoritmo SSTF) y la más próxima en orden ascendente (en el caso del algoritmo C-SCAN ascendente).

Calculemos ahora el tiempo de acceso al bloque de disco del cilindro 35:

Tacceso= Tbúsqueda+ Retraso Rotacional + Ttransferencia= 1,2 ms. + 10 ms. + 5 ms. = 16,2 ms.

Tbúsqueda= $0,1 * C \text{ ms.} = 0,1 * (35 - 23) \text{ ms.} = 0,1 * 12 \text{ ms.} = 1,2 \text{ ms.}$

Cuando termina de atenderse la demanda del cilindro 35, habrán transcurrido exactamente 55,6 ms. (los 39,4 ms. que tardaron en atenderse las dos primeras demandas más los 16,2 ms. que tarda en atenderse la demanda del bloque de disco del cilindro 35) desde el instante 0 que se consideró al principio del ejercicio. Ese tiempo es suficiente para que hayan llegado al sistema todas las peticiones del enunciado del ejercicio (téngase en cuenta que la última petición llegará exactamente 49 ms. después de que llegue la primera).

Ya podremos, por tanto, aplicar el algoritmo correspondiente sobre todas las restantes peticiones (teniendo en cuenta que las demandas a los cilindros 2, 23 y 35 ya han sido atendidas).

Una vez que han llegado todas las demandas, podemos establecer el orden en que serán atendidas según el algoritmo SSTF: 34, 46, 13, 90 y 102. Y también podemos establecer el orden según el algoritmo C-SCAN: 46, 90, 102, 13 y 34. Para responder a la pregunta que nos plantea el ejercicio, ya sólo nos quedaría calcular el tiempo de acceso a cada bloque de disco y sumar por separado los tiempos correspondientes a cada algoritmo.

Algoritmo SSTF:

Tiempo de acceso al bloque de disco del cilindro 34:

Tacceso= Tbúsqueda+ Retraso Rotacional + Ttransferencia= 0,1 ms. + 10 ms. + 5 ms. = 15,1 ms.

Tbúsqueda= $0,1 * C \text{ ms.} = 0,1 * (35 - 34) \text{ ms.} = 0,1 * 1 \text{ ms.} = 0,1 \text{ ms.}$

Tiempo de acceso al bloque de disco del cilindro 46:

Tacceso= Tbúsqueda+ Retraso Rotacional + Ttransferencia= 1,2 ms. + 10 ms. + 5 ms. = 16,2 ms.

Tbúsqueda= $0,1 * C \text{ ms.} = 0,1 * (46 - 34) \text{ ms.} = 0,1 * 12 \text{ ms.} = 1,2 \text{ ms.}$

Tiempo de acceso al bloque de disco del cilindro 13:

Tacceso= Tbúsqueda+ Retraso Rotacional + Ttransferencia= 3,3 ms. + 10 ms. + 5 ms. = 18,3 ms.

Tbúsqueda= $0,1 * C \text{ ms.} = 0,1 * (46 - 13) \text{ ms.} = 0,1 * 33 \text{ ms.} = 3,3 \text{ ms.}$

Tiempo de acceso al bloque de disco del cilindro 90:

Tacceso= Tbúsqueda+ Retraso Rotacional + Ttransferencia= 7,7 ms. + 10 ms. + 5 ms. = 22,7 ms.

Tbúsqueda= $0,1 * C \text{ ms.} = 0,1 * (90 - 13) \text{ ms.} = 0,1 * 77 \text{ ms.} = 7,7 \text{ ms.}$

Tiempo de acceso al bloque de disco del cilindro 102:

Tacceso= Tbúsqueda+ Retraso Rotacional + Ttransferencia= 1,2 ms. + 10 ms. + 5 ms. = 16,2 ms.

Tbúsqueda= $0,1 * C \text{ ms.} = 0,1 * (102 - 90) \text{ ms.} = 0,1 * 12 \text{ ms.} = 1,2 \text{ ms.}$

Suma de los tiempos de acceso según el algoritmo SSTF:

22,3 ms. + 17,1 ms. + 16,2 ms. + 15,1 ms. + 16,2 ms. + 18,3 ms. + 22,7 ms. + 16,2 ms. = 144,1 ms.

Algoritmo C-SCAN ascendente:

Tiempo de acceso al bloque de disco del cilindro 46:

Tacceso= Tbúsqueda+ Retraso Rotacional + Ttransferencia= 1,1 ms. + 10 ms. + 5 ms. = 16,1 ms.

Tbúsqueda= $0,1 * C \text{ ms.} = 0,1 * (46 - 35) \text{ ms.} = 0,1 * 11 \text{ ms.} = 1,1 \text{ ms.}$

Tiempo de acceso al bloque de disco del cilindro 90:

Tacceso= Tbúsqueda+ Retraso Rotacional + Ttransferencia= 4,4 ms. + 10 ms. + 5 ms. = 19,4 ms.

Tbúsqueda= $0,1 * C \text{ ms.} = 0,1 * (90 - 46) \text{ ms.} = 0,1 * 44 \text{ ms.} = 4,4 \text{ ms.}$

Tiempo de acceso al bloque de disco del cilindro 102:

Tacceso= Tbúsqueda+ Retraso Rotacional + Ttransferencia= 1,2 ms. + 10 ms. + 5 ms. = 16,2 ms.

Tbúsqueda= $0,1 * C \text{ ms.} = 0,1 * (102 - 90) \text{ ms.} = 0,1 * 12 \text{ ms.} = 1,2 \text{ ms.}$

Tiempo de acceso al bloque de disco del cilindro 13:

Tacceso= Tbúsqueda+ Retraso Rotacional + Ttransferencia= 8,9 ms. + 10 ms. + 5 ms. = 23,9 ms.

Tbúsqueda= $0,1 * C$ ms. = $0,1 * (102 - 13)$ ms. = $0,1 * 89$ ms. = 8,9 ms.

Tiempo de acceso al bloque de disco del cilindro 34:

Tacceso= Tbúsqueda+ Retraso Rotacional + Ttransferencia= 2,1 ms. + 10 ms. + 5 ms. = 17,1 ms.

Tbúsqueda= $0,1 * C$ ms. = $0,1 * (34 - 13)$ ms. = $0,1 * 21$ ms. = 2,1 ms.

Suma de los tiempos de acceso según el algoritmo C-SCAN ascendente:

22,3 ms. + 17,1 ms. + 16,2 ms. + 16,1 ms. + 19,4 ms. + 16,2 ms. + 23,9 ms. + 17,1 ms. = 148,3 ms.

Ejercicio RM6: 2

Un disco cuenta con 180 pistas, 16 sectores diagramados geométricamente con interleave 2 y tarda 16ms en dar una vuelta completa. El tiempo necesario entre una lectura y otra (tiempo de canal) es de 3 ms. Demora 1 ms para cambiar de una pista adyacente a otra, y 18 ms para ir de una punta a la otra del disco. La cabeza se encuentra en la pista 25 al final del sector 2 y la anterior pista leída fue la 21. Le llegan las siguientes peticiones de lectura: 10/0, 130/5, 22/7, 22/8, 2/4. Calcular los tiempos de acceso para cada lectura y el tiempo total para el método **C-LOOKUP**

Solución:

Orden de lectura: 130/5, 2/4, 10/0, 22/7, 22/8

1) 130/5

T. búsqueda: 105 ms

T. latencia: 15 ms

T. Transferencia: 1 ms

T. acceso: 121 ms

2) 2/4

T. búsqueda: 18 ms

T. latencia: 10 ms

T. Transferencia: 1 ms

T. acceso: 29 ms

3) 10/0

T. búsqueda: 8 ms

T. latencia: 11 ms

T. Transferencia: 1 ms

T. acceso: 20 ms

4) 22/7

T. búsqueda: 12 ms

T. latencia: 8 ms

T. Transferencia: 1 ms

T. acceso: 21 ms

5) 22/8

T. búsqueda: 0 ms

T. latencia: 18 ms

T. Transferencia: 1 ms

T. acceso: 19 ms

Ejercicio RM6: 3

Un disco cuenta con 200 pistas, 15 sectores diagramados geométricamente con interleave 1 y gira a 4000 rpm. El tiempo necesario entre una lectura y otra (tiempo de canal) es de 1 ms. Demora 2 ms para cambiar de una pista adyacente a otra, y 18 ms para ir de una punta a la otra del disco. La cabeza se encuentra en la posición 25/8 (pista/sector) y la anterior pista leída fue la 21. Le llegan las siguientes peticiones de lectura: 10/0, 130/5, 22/12, 22/13, 2/14. Calcular los tiempos de acceso para cada lectura y el tiempo total para el método C-SCAN.

Solución

Cálculo del tiempo de una vuelta:

$$4000 \text{ v} \dots\dots\dots 60000 \text{ ms}$$

$$1 \text{ v} \dots\dots\dots \frac{1 \text{ v} \times 60000 \text{ ms}}{4000 \text{ v}} = 15 \text{ ms}$$

Ordenamiento por método C- SCAN: 130/5 – 2/14 – 10/0 – 22/12 – 22/13

1. T.B.= $105 \times 2 = 210 \text{ ms}$
T.L.= 8 ms
T.T.= 1 ms
T.A.= 219 ms
2. T.B.= $69 \times 2 + 18 + 2 \times 2 = 160 \text{ ms}$
T.L.= 7 ms
T.T.= 1 ms
T.A.= 168 ms
3. T.B.= $8 \times 2 = 16 \text{ ms}$
T.L.= 0 ms
T.T.= 1 ms
T.A.= 17 ms
4. T.B.= $12 \times 2 = 24 \text{ ms}$
T.L.= 14 ms
T.T.= 1 ms
T.A.= 39 ms
5. T.B.= 0 ms
T.L.= 1 ms
T.T.= 1 ms
T.A.= 2 ms

Tiempo total = 445



Ejercicio RM6: 4

Un disco cuenta con 180 pistas, 9 sectores diagramados geoméricamente con interleave 1 y tarda 18ms en dar una vuelta completa. El tiempo necesario entre una lectura y otra (tiempo de canal) es de 3 ms. Demora 2 ms para cambiar de una pista adyacente a otra, y 18 ms para ir de una punta a la otra del disco. La cabeza se encuentra en la pista 25 al final del sector 2 y la anterior pista leída fue la 21. Le llegan las siguientes peticiones de lectura: 12/4, 125/14, 22/6, 22/5, 2/0. Calcular los tiempos de acceso para cada lectura y el tiempo total para el método SSTF

Solución:

Ordenamiento por método SSTF: 22/5 – 22/6 – 12/4 – 2/0 – 125/14

1. T.B.= $3 \times 1 = 3 \text{ ms}$
T.L.= $5 \times 1 = 5 \text{ ms}$
T.T.= 1 ms
T.A.= 9 ms
2. T.B.= 0
T.L.= $2 \times 1 + 16 = 18 \text{ ms}$
T.T.= 1 ms
T.A.= 19 ms
3. T.B.= $10 \times 1 = 10 \text{ ms}$
T.L.= $15 \times 1 = 15 \text{ ms}$
T.T.= 1 ms
T.A.= 26 ms
4. .B.= $10 \times 1 = 10 \text{ ms}$
T.L.= $9 \times 1 = 9 \text{ ms}$
T.T.= 1 ms
T.A.= 20 ms
5. T.B.= $123 \times 1 = 123 \text{ ms}$
T.L.= $14 \times 1 = 14 \text{ ms}$
T.T.= 1 ms
T.A.= 138 ms

Tiempo total = 212 ms



Ejercicio RM6: 5 FINAL UTN DEL 18/02/2006**C) Resuelva en forma clara y detallada los siguientes ejercicios.**

Un sistema cuenta con un disco rígido con las siguientes características: Tiempo "pista a pista": 1ms; tiempo de 1 rotación entera: 10ms; Sectores de 512 Bytes; Sectores por pista: 10; cilindros: 100; Cabezas: 2. Asumiendo que los pedidos que solicitan los procesos P1 y P2 están al mismo tiempo en el buffer de pedidos, calcule el tiempo necesario para satisfacer dichos pedidos sabiendo que el sistema operativo implementa el algoritmo F-SCAN y que en un momento dado los procesos quieren leer los archivos representados por las siguientes tablas:

Archivo 1: registros de 512 bytes

Registro	0	1	2	3	4	5
Dirección lógica	10	20	15	22	100	50

Archivo 2: registros de 1024 bytes

Registro	0	1	2	3	4	5	6	7
Dirección lógica	12/13	52/60	80/81	110/51	200/201	11/101	204/205	202/203

P1: Leer (Archivo1, 0, 2) siendo (archivo a leer, primer registro a leer, cantidad de registros a leer)

P2: Leer (Archivo2, 4, 2)

Asuma que la cabeza de lectura / escritura se encuentra en el sector físico (0, 0, 1), que la organización de los registros de los archivos es contigua y que el brazo se mueve hacia los cilindros mayores.

DATOS:

Seek time = 1ms

Rotation time = 10 ms

Sectores por pista = 10 por lo que tarda un sector = 1 ms.

Cilindros = 100

Cabezas = 2

Sectores = 512 By

En un segundo hace 100 rotaciones,

Transfer time = (cant. By) / (t_{RPS} * By por Pista) = $512 / (100 * 5120) = 1$ ms por sector

Se asume que los pedidos están al mismo tiempo en el buffer, el F-SCAN degenera en SCAN

Pedidos Lógicos: 10, 20, 200, 201, 11, 101

Dir. Lógica	Dir. Física
10	(0,1,1)
20	(1,0,1)
200	(10,0,1)
201	(10,0,2)
11	(0,1,2)
101	(5,0,2)

Por ser SCAN los pedidos se ordenan: 10, 11, 20, 101, 200, y 201.

Direcciones	Seek time	Rotation Delay Time	Transfer time	Observaciones
(0,0,1) → (0,1,1)	0 ms	0ms	1 ms	Estan en el buffer
(0,1,2) → (0,1,2)	0 ms	0 ms	1 ms	idem
(0,1,3) → (1,0,1)	1 ms	7 ms	1 ms	
(1,0,2) → (5,0,2)	4 ms	6 ms	1 ms	
(5,0,3) → (10,0,1)	5 ms	3 ms	1 ms	
(10,0,2) → (10,0,2)	0 ms	0 ms	1 ms	
Subtotales:	10 ms	16 ms	6 ms	
Tiempo total:	10 ms + 16 ms + 6 ms = 32 ms			

Respuesta: el tiempo necesario para satisfacer dichos pedidos es de 32 ms.



Ejercicio RM6: 6 Final UTN del 28/05/03

Se tiene un disco con la siguiente geometría: 3 platos, 100 sectores por pista, un total de 500 cilindros y sectores de 512 bytes. En un momento determinado se leen los primeros 3 registros del archivo 1 y los primeros 4 registros del archivo 2.

Sean los archivos con sus respectivos sectores lógicos:

Archivo 1 (tiene registros de 1024 bytes): 1301 / 1902, 2503 / 3704, 599 / 600.

Archivo 2 (tiene registros de 512 bytes): 30023, 30024, 3800, 60102.

Si se sabe que el brazo del disco se encuentra en el cilindro 4, graficar el orden de ejecución de los pedidos, asumiendo que están en la cola, para cada uno de los siguientes algoritmos:

- SSTF.
- F-SCAN – El brazo está ascendiendo.

SOLUCIÓN

Direcciones Físicas

(cilindro, cabeza, sector)

(2, 1, 2)

(3, 1, 3)

(4, 1, 4)

(6, 1, 5)

(0, 5, 100)

(1, 0, 1)

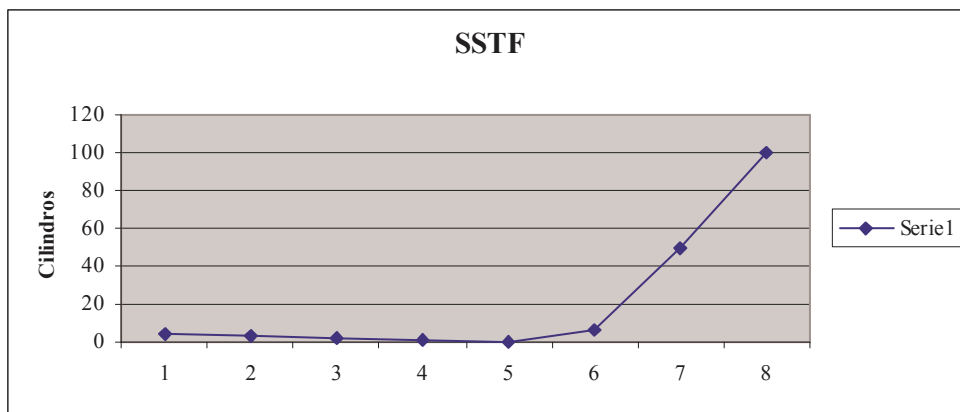
(50, 0, 24)

(50, 0, 25)

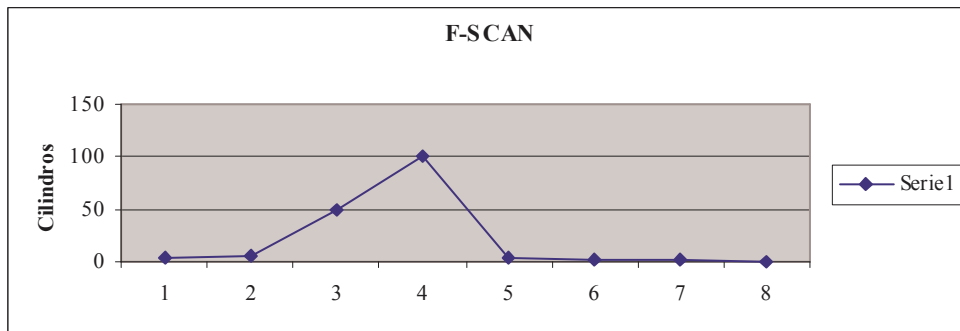
(6, 2, 1)

(100, 1, 3)

a) 4 – 3 – 2 – 1 – 0 – 6 (x2) – 50 (x2) – 100



b) 4 – 6 (x2) – 50 (x2) – 100 – 3 – 2 – 1 – 0



Ejercicio RM6: 7 Final UTN del 29/05/06

Se tiene un Sistema con un procesador P-IV con 1 GB de memoria y un disco rígido de aproximadamente 8 Gigabytes. Este disco contiene físicamente 256 cilindros y el brazo tarda 256 ms en recorrer todos los cilindros, numerados del 0 al 255. El sistema tiene los siguientes pedidos de lectura de cilindro encolados en el instante 0:

94, 47, 226

El sistema recibe los siguientes pedidos de lectura de disco en los siguientes momentos:

Instante (en ms)	2	3	4	4	4
Pedido de lectura del cilindro:	38	81	198	67	124

Considerando que la cabeza del disco se encuentra actualmente en el cilindro 65 y se esta moviendo hacia el cilindro externo (cilindro 255), indicar cual es el tiempo requerido para atender los pedidos si se utilizan los siguientes algoritmos:

- FCFS
- LOOK
- SSTF
- C-SCAN

SOLUCIÓN

A) FCFS

POSICIÓN	INSTANTE	ORDEN PEDIDOS Cyl	Estado de la Cola	Tiempos Parciales
65	0	94	47,226	
67	2		47,226,38	
68	3		47,226,38,81	
69	4		47,226,38,81,198,67,124	
94	29	47	226,38,81,198,67,124	29
47	76	226	38,81,198,67,124	47
226	255	38	81,198,67,124	179
38	443	81	198,67,124	188
81	486	198	67,124	43
198	603	67	124	117
67	734	124	---	131
124	791	--	---	57
TIEMPO TOTAL:				791 ms

A) LOOKUP

POSICIÓN	INSTANTE	ORDEN PEDIDOS Cyl	Estado de la Cola	Tiempos Parciales
65 subiendo	0	94	226,47	0
67 sub	2		226,47,38	
68 sub	3		226,81,47,38	
69 sub.	4		124,198,226,81,67,47,38	
94sub	29	124	198,226,81,67,47,38	29
124sub.	59	198	226,81,67,47,38	30
198 sub	133	226	81,67,47,38	74
226 baja	161	81	67,47,38	28
81 baja	306	67	47,38	145
67 baja	320	47	38	14
47	340	38	---	20
38	349	---	---	9
TIEMPO TOTAL:				349 ms

B) SSTF

POSICIÓN INICIAL	INSTANTE	ORDEN PEDIDOS Cyl	Cola	Tiempos Parciales
65	0	47	94,226	0
63	2		38,94,226	
62	3		38,81,94,226	
61	4		38,67,81,94,124,198,226	
47	18	38	67,81,94,124,198,226	18
38	27	67	81,94,124,198,226	9

67	56	81	94,124,198,226	29
81	70	94	124,198,226	14
94	83	124	198,226	13
124	113	198	226	30
198	187	226	---	74
226	215	---	---	28
TIEMPO TOTAL:				215 ms

c) C-SCAN

POSICIÓN	INSTANTE	ORDEN PEDIDOS Cyl	Estado de la Cola	Tiempos Parciales
65 subiendo	0	94	226,47	0
67 sub	2		226,47,38	
68 sub	3		226,81,47,38	
69 sub.	4		124,198,226,81,67,47,38	
94sub	29	124	198,226,81,67,47,38	29
124sub.	59	198	226,81,67,47,38	30
198 sub	133	226	81,67,47,38	74
226 sub	161	Max	81,67,47,38	28
255 invertir	190		38,47,67,81	29
0	190	38	47,67,81	0
38	229	47	67,81	39
47	238	67	81	9
67	258	81	---	20
81	272	---	---	14
TIEMPO TOTAL:				272 ms

Ejercicio RM6: 8**Final del 25/02/2006**

Se tiene un disco rígido de 48000 KB cuyo cabezal tarda 2 ms en pasar de una pista a otra. Las cabezas leen del sector más chico hacia el más grande (1,2,...,n). La configuración del disco es la siguiente: 6 pistas por cilindro, 100 cilindros y un tamaño de sector de 4KB. La cabeza se encuentra en la dirección lógica 100, ascendiendo.

Los pedidos a disco que realiza un proceso en el Sistema se encuentran representados por la siguiente tabla:

T	0	1	4	4	5	7	8	8
DL	200	380	240	390	490	310	20	150

Siendo:

T: El tiempo en que el pedido llega a la cola, expresado en ms.

DL: Dirección lógica a acceder.

Disco: cilindros = 100 (0 → 99)

Cabezas = 6 (0 → 5)

Sectores = 20 sectores por pistas que se calcula por :

48000 kB/ 4 kB = 12.000 bloques

100 pistas/cabezas * 6 cabezas * ¿?sectores/pistas = 12000 / 600 = 20 sectores por pista

Tiempo de búsqueda = ST = 2 mseg. Por cilindro

T: tiempo	DL (DIRECCIÓN LÓGICA)	DF (DIRECCIÓN FÍSICA: cilindro, cabeza, sector)
-	100	(0, 5, 1)
0	200	(1, 4, 1)
1	380	(3, 1, 1)
4	240	(2, 0, 1)
4	390	(3, 1, 11)
5	490	(4, 0, 11)
7	310	(2, 3, 11)
8	20	(0, 1, 1)
8	150	(1, 1, 11)

B2.1. Si se sabe que se tiene un Sistema Operativo que utiliza el C – LOOK como política de planificación de disco, indique el orden en que fueron atendidos los pedidos.

T	cilindro	Pedidos atendidos	cola
0	0	-	200 (1)
2	1	200	380 (3)
4	2	240	380 (3), 240 (2), 390 (3)
6	3	380, 390	490 (4)
8	4	490	490 (4), 310 (2), 20 (0), 150 (1)
16	0	20	310 (2), 150 (1)
18	1	150	310 (2)
20	2	310	

RESPUESTA AL PUNTO B2.1

ORDEN DE ATENCIÓN: 200, 240, 380, 390, 490, 20, 150, 310.

B2.2. Considerando para este punto que la velocidad de rotación del disco es de 6000 RPM, y se ha cambiado el algoritmo de planificación de disco a SCAN, indique el Tiempo medio de atención de los siguientes pedidos: 400, 124, 360, considerando que la diferencia de tiempo de llegada entre pedidos es de 4 ms y que la cabeza se encuentra nuevamente en la dirección lógica 100, ascendiendo.

DISCO: 6000 REVOLUCIONES POR MINUTO → 100 REV. POR SEG. → 1 REV. EN 10 mseg.

20 sectores en 10 mseg → 1 sector = 0,5 mseg.

Seek time = ST = 2 mseg por cilindros

Rotation time = RT = 10 mseg por giro

Posición actual : (0, 5, 1)

LLEGADA	DL	DF
0	100	(0, 5, 1)
0	400	(3, 2, 1)
4	124	(1, 0, 5)
8	360	(3, 0, 1)

PLANIFICACIÓN SCAN

LLEGA DA	DL	DF	destino	ST (tiempo de busqueda)	RT (tiempo rotacion al)	TL (tiemp o de lectura)	TA (tiempo de atención)
0	10 0	(0, 5, 1)	(3, 2, 1)	-	-	-	0
Cola: (3,0,1) (1, 0,5) entonces (3, 2, 2) → (3,0,1)							
0	40 0	(3, 2, 2)	(3, 0, 1)	6	4	0,5	10,5
8	36 0	(3, 0, 2)	(1, 0, 5)	0	9,5	0,5	10
Debe ir hasta el último cilindro y volver atendiendo hacia abajo (3, 0, 2) → (99, x, x) tarda 192 mseg → (99, x, 6) → (1,0, 5) en que tarda 196 ms + 2 ms en total 390 ms de ST							
4	12 4	(1, 0, 5)	(1, 0, 18)	390	1,5	05	392
Tiempo medio de atención:				(10,5 + 10 + 392) / 3 = 137,5 ms			

Ejercicio RM6: 9

Final del 22/09/2005

En el desarrollo de un nuevo kernel para el sistema GNU/Stv-Linux se necesita saber que tan buena es la performance implementando 4 discos bajo una arquitectura RAID 4 y donde el Short Term Scheduler es

$NroBloque \bmod (cantidadDeSectPorPista) = sectorDeEsaPista$ (2)
 por (1) $NroPista \% (cantidadDePistasPorCilindro) = cilindroActual$ (3)
 por (1) $NroPista \bmod (cantidadDePistasPorCilindro) = superficieActual$ (4)

Deducción: tenemos dos cuentas de dividir, como divisor * cociente + resto = dividendo =>
 por (1) y (2)

$$nroPista * (cantidadDeSectPorPista) + sectorDeEsaPista = NroBloque \quad (5)$$

por (3) y (4)

$$(cantidadDePistasPorCilindro) * cilindroActual + superficieActual = nroPista \quad (6)$$

Reempl nroPista de (5) en (6) quedaría :

$$(cantidadDePistasPoCilindro * cilindroActual + superficieActual) * cantDeSectPorPista + sectorDeEsaPista = NroBloque$$

$$o \text{ lo que es igual } (i*t + j)*s + k = b$$

Aplicando (1) y (2) :

$$*) NroBloque = 355$$

$$355 \underline{) 10}$$

$$5 \quad 35 \underline{) 8}$$

$$3 \quad 4 \quad (sect:5, sup:3, cil:4)$$

$$Verificación : (i*t + j)*s + k = b. \quad (4*8 + 3)*10 + 5 = 355$$

$$*) NroBloque = 1552$$

$$1552 \underline{) 10}$$

$$2 \quad 155 \underline{) 8}$$

$$3 \quad 19 \quad (sect:2, sup:3, cil:19)$$

$$Verificación : (i*t + j)*s + k = b. \quad (19*8 + 3)*10 + 2 = 1552$$

$$*) NroBloque = 110$$

$$110 \underline{) 10}$$

$$0 \quad 11 \underline{) 8}$$

$$3 \quad 1 \quad (sect:0, sup:3, cil:1)$$

$$Verificación (i*t + j)*s + k = b. \quad (8*1 + 3)*10 + 0 = 110$$

Concluimos que las peticiones son :

	pet1	pet2	pet3
cil	4	19	1
sect	5	2	0
sup	3	3	3

2) De acuerdo a los dos movimientos del disco, sugen dos tiempos, (t. posicionamiento y latencia), para ello necesitamos :

2.1) t.posicionamineto entre pistas en una unidad común (ms) :

$$1000 \mu\text{seg} \underline{\hspace{1cm}} 1 \text{ ms}$$

$$800 \mu\text{seg} \underline{\hspace{1cm}} 0,8 \text{ ms.}$$

2.2) t dar una vuelta :

$$7000 \text{ v} \underline{\hspace{1cm}} 60 \text{ seg}$$

$$1 \text{ v} \underline{\hspace{1cm}} 60\text{seg} / 7000 \text{ v} = 0,00857 \text{ seg} = 8,57 \text{ ms.}$$

2.3) t. de recorrer un sector

$$10 \text{ sect} \underline{\hspace{1cm}} 8,57 \text{ ms}$$

$$1 \text{ sect} \underline{\hspace{1cm}} x = 0,857 \text{ ms}$$

3) Análisis del interleave (10 sectores - interleave 2)

3
 6 0
 9 7
 2 4
 5 1
 8

Se recorrerá en sentido de las agujas del reloj

4) Aplicación de los algoritmos de planificación.

De acuerdo al algoritmo scan, en subida debe ir hasta el final y al bajar atenderá las peticiones

19
2

$$t.pos = 159 \text{ pista} + 180 \text{ pista} = 339 \text{ pistas} * 0,8 \text{ mseg} / \text{pista} = 271,2 \text{ ms}$$

Para calcular el t de latencia :

$$8,57 \text{ mseg} \underline{\hspace{1cm}} 10 \text{ sect}$$

$$271,2 \text{ mseg} \underline{\hspace{1cm}} x = 316,45 \text{ sect}$$

$$10 \text{ sect} \underline{\hspace{1cm}} 1 \text{ vue}$$

$$316,45 \text{ sect} \underline{\hspace{1cm}} 316,4 \underline{\hspace{1cm}} 10$$

$$7 / 31 \text{ vue}$$

luego habiéndose movido 7 sectores desde la 0 estando al final, queda al principio de la pista 6, para llegar hasta la 2 restan 8 sectores :

$$t. \text{ latencia} = 8 * 0,857 \text{ ms} = 6,856 \text{ ms}$$

$$t. \text{ lectura} = 0,857 \text{ mseg}$$

$$t. \text{ transf} = 14 \text{ ms}$$

Por algoritmo scan le sigue la 4 - 5

4
5

$$t.pos = 15 \text{ pistas} * 0,8 \text{ ms} / \text{pista} = 12 \text{ mseg}$$

Para calcular el t de latencia :

$$8,57 \text{ ms} \underline{\hspace{1cm}} 10 \text{ sect}$$

$$12 \text{ ms} \underline{\hspace{1cm}} 14,002 \text{ sect} = 15 \text{ sect}$$

$$10 \text{ sect} \underline{\hspace{1cm}} 1 \text{ v}$$

$$15 \text{ sect} \underline{\hspace{1cm}} 15 \underline{\hspace{1cm}} 10$$

$$5 / 1$$

Desde el final de la 2, si se movió 5 sectores y debo ir al 5, debo moverme 3

$$t. \text{ latencia} = 3 * 0,857 = 2,57 \text{ ms}$$

SE VALIDA SI $t.transf > t.latencia + t.posicionamiento$ (da Verd)

$$t.lect = 0,857 \text{ ms}$$

$$t.transf = 14 \text{ ms}$$

1
0

$$t.pos = 3 \text{ pistas} * 0,857 \text{ mseg} / \text{pista} = 2,571 \text{ ms}$$

$$8,57 \text{ ms} \underline{\hspace{1cm}} 10 \text{ sect}$$

$$2,5713 \text{ ms} \underline{\hspace{1cm}} 3,001 \text{ sect} = 4 \text{ sectores}$$

Para ir al sector 0 , ya está delante de él pero

SE VALIDA SI $t.transf > t.latencia + t.posicionamiento$ (da Falso)

para que esto se cumpla debería dar dos vueltas :

$$t.latencia = 0 \text{ ms} + 8,57 \text{ ms} + 8,57 \text{ ms}$$

$$t.lect = 0,857 \text{ ms}$$

$$t.transf = 14 \text{ ms}$$

$$\text{El tiempo total} = 271,2 \text{ ms} + 6,856 \text{ ms} + 0,857 \text{ ms} + 14 \text{ ms} + 12 \text{ ms} + 2,57 \text{ ms} + 0,857 \text{ ms} + 14 \text{ ms} + 2,57 \text{ ms} + 8,57 \text{ ms} + 8,57 \text{ ms} + 0,857 \text{ ms} + 14 \text{ ms} = \mathbf{356.9 \text{ ms}}$$

Ejercicio RM6:11



EJERCICIOS SIN RESOLVER

Ejercicio M6: 1

¿Sería ventajoso transferir desde un puerto serie hacia la CPU usando DMA los caracteres recibidos, suponiendo que existe una terminal conectada a dicho puerto?.

Ejercicio M6: 2

Supongamos la siguiente red de conexión de dispositivos

Canal 1	Unidad de Control "A"	Dispositivo 1
Canal 1	Unidad de Control "A"	Dispositivo 2
Canal 2	Unidad de Control "B"	Dispositivo 1
Canal 2	Unidad de Control "B"	Dispositivo 3
Canal 2	Unidad de Control "A"	Dispositivo 2
Canal 2	Unidad de Control "A"	Dispositivo 1

Se pide:

- ¿Cuál es la razón para que existan distintas rutas de acceso a los dispositivos?
- ¿Es posible que los dispositivos 1 y 2 transfieran información en forma simultánea? Justifique
- ¿Qué módulo establece la ruta para llevar a cabo la operación de entrada/salida?
- Diseñe la base de datos que maneja el software en esta red de conexión de dispositivos

Ejercicio M6: 3

¿Cuál es el porcentaje de cinta desperdiciada si se graba información con longitud de registro de 40 bytes, bloqueados de a 5 registros por bloque? Considere un espacio entre registros físicos de $\frac{3}{4}$ pulgada y densidad de grabación de 800 BPI.

Ejercicio M6: 4

Una placa de interfase con un Hard Disk transfiere los datos desde el disco hacia la CPU de a UN caracter por vez, generando una interrupción por cada byte enviado. La rutina encargada de procesar dicha interrupción tarda en ejecutarse por completo 2,32 micro segundos (overhead incluido). ¿Cuál es la máxima transferencia que se puede obtener en ese sistema ?

Ejercicio M6: 5

Dado un programa que debe leer totalmente un archivo secuencial de 2354 registros de 80 bytes cada uno, grabados sobre un disco cuyos sectores son de 240 bytes, se pide:

- ¿Cuántas operaciones de E/S deben lanzarse durante la ejecución del programa?
- Si el archivo fuese de acceso directo y el programa lo trata como tal, ¿cuántas operaciones de E/S se lanzan si el programa quiere leer 650 registros al azar?

Ejercicio M6: 6

Se desea grabar un archivo de 25.200.000 bytes de longitud, cuyo registro lógico tiene 125 bytes, y el factor de bloqueo es de 30 (o sea 30 registros lógicos es un registro físico). Se dispone de una unidad de disco de 8 superficies, cada superficie tiene 100 pistas, y cada pista tiene 63 sectores de 512 bytes cada sector. Considere que el disco sólo almacena ese archivo, y que no existen directorios, boot sector, u otra cosa. Además considere que un registro no puede ocupar 2 sectores distintos. El disco gira a 360 r.p.m., un procesador graba mediante la técnica de E/S por interrupciones (Una interrupción por byte). Cada interrupción lleva 2.5 nano segundos de procesamiento. El buffer de E/S es de 32256 bytes. Se pide:

- ¿Qué ocurre si se desea agregar un registro al archivo ?
- ¿Cuál es el porcentaje de tiempo que le dedica el procesador para atender la E/S ?
- Ídem a b), pero utilizando la técnica por DMA. (Considere una interrupción por sector)

Ejercicio M6: 7

Suponiendo que un disco tiene 200 pistas, y al cual le llegan en un determinado momento las siguientes solicitudes: 100, 199, 45, 131, 10, 128, 68, 73, 31, 0, y 22. La cabeza se encuentra actualmente en la posición 50, y anteriormente había leído la posición 49. Se pide que realice un gráfico mostrando los movimientos de la cabeza, con los siguientes métodos. También debe calcular la cantidad de movimientos en total que fueron necesario en ese método para atender todas las solicitudes.

1. FCFC
2. SSTF
3. SCAN
4. C-SCAN
5. LOOK-UP
6. C- LOOK-UP

Ejercicio M6: 8

Suponiendo un disco con una cabeza móvil, el cual cuenta con 200 pistas, 9 sectores, que tarda en efectuar una revolución 9 ms., pasa entre pistas adyacentes cada 0,8 ms. y tarda en ir de una punta a la otra del disco sin atender solicitudes 1.6 ms.

La cabeza lectora se encuentra actualmente en la posición 53/1, y anteriormente estaba en la 25/0.

Dada la siguiente secuencia de lecturas: (Pista / sector): 98/5 - 67/7 - 183/2 - 37/4 - 122/8 - 14/3 - 124/0 - 65/0

Se pide efectuar el orden de ejecución, y el tiempo demorado en atender todas las solicitudes para los siguientes algoritmos:

- FCFS
- SSTF

Ejercicio M6: 9

Suponiendo un disco con una cabeza móvil, el cual cuenta con 200 pistas, 15 sectores, 7500 rpm., y que pasa entre pistas adyacentes cada 1 ms. y tarda en ir de una punta a la otra del disco sin atender solicitudes 3.6 ms.

La cabeza lectora se encuentra actualmente en la posición 53,8 (pista, sector), y anteriormente estaba en la 25,7.

Dada la siguiente secuencia de lecturas: 98,4 / 98,5 / 183,8 / 37,6 / 122,0 / 14,3 / 124,4 / 65,7 / 67,8

Se pide efectuar el orden de ejecución, y los cálculos del tiempo demorado en posicionarse para cada una de las lecturas (y el total) para los métodos

- SSTF
- C-LOOK

Ejercicio M6: 10

Un proceso realiza los siguientes pedido al driver del disco: 1-2-44 -55-11-32-46-78-11-2-34-98 -22 -10 -52 -22-76 -73. Sabiendo que el algoritmo de planificación del brazo es el SSTF (Shortest Seek Time First) y que el brazo tarda en pasar entre pistas 1 ms, se pide que indique el tiempo de atención de todos los pedidos.

1a. Realice el ejercicio 1, teniendo en cuenta los algoritmos SCAN, C-SCAN, LOOK y C-LOOK, sabiendo que el disco cuenta con un total de 100 pistas (numeradas de 0 a 99) y que la cabeza se encuentra actualmente posicionada en la pista 32 ascendiendo.

Ejercicio M6: 11

Sean las siguientes direcciones lógicas emitidas por un proceso al driver de un disco: 100 - 2000 - 4529 - 7222 - 720 - 7286 - 1276. El disco consta de: 300 cilindros, cada una de sus istas tiene 10 sectores y consta de 3 platos conformando un total de 6 cabezas. Si se sabe que el tiempo entre pistas es de 2 ms, que el tiempo entre sectores es de 1 ms y que las cabezas leen del sector mas chico al mas grande, se pide: El tiempo de atención y de posicionamiento de los pedidos, sabiendo que el algoritmo de planificación es el SSTF (Shortest Seek Time First).

Ejercicio M6: 12

Se dispone un dispositivo de almacenamiento compuesto por un RAID nivel uno. Cada uno de los discos del RAID están compuestos por 300 cilindros, cada una de sus pistas tiene 10 sectores y consta de 3 platos conformando un total de 6 cabezas. La velocidad de rotación es de 6000 RPM y el cabezal tarde 1ms en pasar de una pista a otra. Las cabezas leen del sector más chico hacia el más grande, es decir, primero leen el sector 1, luego el 2, etc.

Actualmente uno de los discos del RAID está ubicado en el sector lógico 1107 y se dirige al sector 1108.

En su cola están los siguientes pedidos (sectores lógicos) encolados: 100, 1244, 15, 955.

Se pide que dé el orden en que se lee cada uno de los sectores lógicos, el tiempo que se tardó en posicionar el cabezal para leerlo (desde el instante que se toma el pedido) y que cilindro, cabeza, sector lo lee.

Para la resolución de este ejercicio considere las siguientes dos políticas:

a. SCAN

b C-LOOK

Ejercicio M6: 13

Sea un Sistema que implementa memoria virtual, para lo cual utiliza un algoritmo de elección de la víctima LRU (Least Recently Used). Las referencias a memoria que realiza un proceso urante su ejecución son: 100 – 3817 – 83 – 7353 – 9822 – 72 – 82 -8821 – 3400 (Direcciones lógicas en decimal). Se sabe que se utilizan 10 bits para la página y 22 bits para representar el offset y que el proceso dispone de 3 frames, inicialmente vacíos, donde uno de ellos es utilizado para el código. A su vez el Sistema posee un disco rígido con 100 cilindros, numerados de 0 a 99.

Considere, para simplificar la resolución de este ejercicio, que el número de página coincide con el número de cilindro. Así por ejemplo la página 1 se encuentra ubicada en el cilindro 1 del disco. Si se sabe que el algoritmo de planificación del disco es el SSTF (Shortest Seek Time First) y que el tiempo que se insume en moverse entre cilindros es de 2 ms, se pide que indique:

a. El tiempo total que se insume en atender los pedidos a disco.

b. El estado de la memoria en cada instante de tiempo.

Ejercicio M6: 14

Se tiene un Sistema Operativo que utiliza un esquema SRT (Shortest Remaining Time) para gestionar los procesos que se encuentran en la cola de ready. A su vez, se sabe que existen dos dispositivos de Entrada / Salida, a saber: Un disco, el cual planifica los pedidos que llegan al driver mediante el algoritmo SSTF (Shortest Seek Time First) y una impresora, que según las características técnicas, imprime 5 páginas a color y 8 páginas en blanco y negro cada 20 segundos.

Le surge la necesidad a un alumno de Ingeniería en Sistemas de Información en saber cuanto tiempo le insumirá imprimir la versión final del trabajo práctico de Sistemas Operativos, debido a que sabe que va a llegar con muy poco tiempo a rendir la entrega final.

Es por este motivo que ha decidido realizar una estimación de las operaciones de CPU y Entrada / Salida que requerirá para finalizar el Trabajo, los cuales se encuentran representados por la siguiente tabla:

Proceso	T. Llegada	CPU	E/S	CPU	E/S	CPU
P1	0	3	Disco(3,1,222)	2	Impresora(10, B/N)	2
P2	0	2	Disco(1,3,56)	4	Impresora(2, COLOR)	6
P3	2	5	Impresora(2, B/N)	1	Disco(72, 2, 90)	1
P4	3	3	Disco(56, 2, 87)	3	Disco(69,7,200)	3
P5	5	4	Impresora(10, COLOR)	5	Disco(70, 8, 97)	4

Los pedidos que llegan al driver de la impresora se atienden FIFO y todos los tiempos se encuentran expresados en segundos. Teniendo en cuenta que la cabeza del disco se encuentra ubicada en la dirección lógica 2097 (correspondiente a la dirección física (10, 4, 18)) ascendiendo y que el tiempo entre pistas es de 1 segundo. Se pide:

a) El tiempo de finalización de los procesos. Justifique su respuesta, **mediante la confección de un diagrama de GANTT**.

b) Si se considera que el disco cuenta con 10 cabezas. ¿Cuántos sectores por pista posee este disco? **Justifique** los cálculos que lo llevan a su conclusión.

NOTA: La función "Disco" recibe como parámetro la dirección física a la cual acceden los procesos y la función "Impresora" recibe como parámetro la cantidad de hojas a imprimir y el tipo de impresión.

Ejercicio M6:15 LAS MOROCHAS, EL CABALLERO, LAS RUBIAS Y EL REMERO.

En cierto paraje existe un caballero que, debido a su soledad, desea casarse rápidamente. Como está ocupado y en el pueblo no existe ninguna mujer que le guste, contrata a un remero entrado en años que debe recorrer un río bastante turbulento, a lo largo del cual existen pequeñas islas en donde viven muchas señoritas. El remero trataría de ver y encontrar alguna candidata y convencerla de que vaya al pueblo y se case con el caballero. El hombre del bote tiene prohibido pisar tierra, salvo la de dichas islas. En ciertas islas sólo viven mujeres rubias y en otras solamente chicas morochas. Como el caballero tiene preferencia por estas últimas y el remero, debido a su edad no puede estar remando mucho tiempo seguido, surge una incógnita: tendrá altas probabilidades de casarse este hombre si las chicas morochas viven en las islas cercanas a las costas?

Responder, justificar y trazar el paralelo con el tema de E/S

Ejercicio M6:16 NUEVA VERSIÓN DE LAS MOROCHAS, EL CABALLERO, LAS RUBIAS Y EL REMERO

En cierto paraje existe un caballero al que, recientemente, su novia del barrio abandonó decidiendo irse a vivir a una isleta ubicada en medio de un largo río. Por dicho río navega eternamente un remero entrado en años que recorre las islas en las cuales debe realizar alguna tarea. Como el caballero extraña a su amada, se las ingenia para avisarle al remero que acuda a la isla en donde vive su chica y la convenza para que vuelva con él (Es el único que podría hacerlo). Ahora bien, como el remero está viejito, este no puede estar remando mucho tiempo seguido, por lo tanto surge una incógnita:

Tendrá altas posibilidades de recuperar el caballero a su ex-novia si ésta vive en una isla cercana a la costa?

Responder, justificar y comentar de que caso de entrada-salida se está tratando.

Ejercicio M6:17

Dado un programa que debe leer totalmente un archivo secuencial de 2354 registros de 80 bytes cada uno, grabados sobre un disco cuyos sectores son de 240 bytes, se pide:

- ¿Cuántas operaciones de E/S deben lanzarse durante la ejecución del programa?
- Si el archivo fuese de acceso directo y el programa lo trata como tal, ¿cuántas operaciones de E/S se lanzan si el programa quiere leer 650 registros al azar?

¿Cuál es el porcentaje de cinta desperdiciada si se graba información con longitud de registro de 40 bytes, bloqueados de a 5 registros por bloque? Considere un espacio entre registros físicos de $\frac{3}{4}$ pulgada y densidad de grabación de 800 BPI.

Sería ventajoso transferir desde un puerto serie hacia la CPU usando DMA los caracteres recibidos, suponiendo que existe una terminal conectada a dicho puerto ?.

Si se posee una sola impresora conectada a un sistema multiusuario, y se generan dos pedidos simultáneos de impresión . ¿De cuantas formas se puede resolver el conflicto?.

¿Por que los dispositivos poseen, en un sistema de computación, un 'device driver' encargado de administrarlos?

Ejercicio M6:18

Poseemos un sistema con un disco de 400 cilindros, divididos en 16 sectores de medio kilobyte cada uno. Dicho disco gira a 6.000 r.p.m. y posee una tasa de transferencia de 300 KBy/segundo. Actualmente la cabeza se encuentra posicionada en la pista 190 y sector 7, habiendo estado previamente sobre el cilindro 180 y sector 15. A partir de este momento, tiene que atender las siguientes peticiones:

PISTA	283	25	1	278	104	73
SECTOR	2	0	9	9	14	6

Representar gráficamente la atención de solicitudes y calcular el tiempo que insumirán, teniendo en cuenta que el tiempo que tarda la cabeza en ir de una pista a la otra es de 5 mseg y que la penúltima petición atendida será en la pista 278.

Ejercicio M6:19

Suponga que tenemos una unidad de almacenamiento de discos con una cabeza móvil. Existen 200 posiciones (pistas) desde donde la cabeza puede leer o grabar información (pistas 0 a 199). Sobre cada pista hay 8 registros (0..7). El disco realiza una revolución cada 8 mseg. La cabeza es movida entre pistas adyacentes cada 0,8 mseg.

La rutina que maneja el disco recibe la siguiente lista de solicitudes de lectura:

Pista	0	10	10	10	20	20
Registro	2	2	3	5	4	3

- ¿Cuánto tiempo tardará en realizar los requerimientos de E/S en el orden recibido (FIFO)? Suponga que la posición inicial de la cabeza es (0,0).
- ¿Cuál es el orden óptimo para satisfacer los requerimientos? Describa en palabras el algoritmo que utiliza.
- Explique cómo cada uno de los siguientes puntos pueden ser usados en la rutina para optimizar los tiempos de E/S:
 - Una instrucción para tomar conocimiento de la posición actual de la cabeza sobre la pista.
 - Una instrucción que mueve la cabeza liberando al canal mientras la instrucción se complete.
 - Un tiempo de movimiento de cabeza entre pistas adyacentes de 0,2 mseg.
- ¿Qué efectos espera obtener sobre el tiempo de respuesta del disco si:
 - Agrega más cabezas
 - Agrega una segunda cabeza en el lado opuesto del disco.

Ejercicio M6: 20

Suponiendo un disco con una cabeza móvil, el cual cuenta con 200 pistas, 9 sectores, que tarda en efectuar una revolución 9 ms., pasa entre pistas adyacentes cada 0,8 ms. y tarda en ir de una punta a la otra del disco sin atender solicitudes 1.6 ms.

La cabeza lectora se encuentra actualmente en la posición 53/1, y anteriormente estaba en la 25/0.

Dada la siguiente secuencia de lecturas:

- (Pista / sector): 98/5 - 67/7 - 183/2 - 37/4 - 122/8 - 14/3 - 124/0 - 65/0

Se pide efectuar el orden de ejecución, y el tiempo demorado en atender todas las solicitudes para los siguientes algoritmos:

- FCFS
- SSTF

Ejercicio M6: 21

Suponiendo un disco con una cabeza móvil, el cual cuenta con 150 pistas, 16 sectores, 5400 rpm., y que pasa entre pistas adyacentes cada 0.5 ms. , y tarda en ir de una punta a la otra del disco sin atender solicitudes 36 ms. El disco está formateado a bajo nivel con un interleave de 0

La cabeza lectora se encuentra actualmente en la posición 53,8 (pista, sector), y anteriormente estaba en la 52,7.

Dada la siguiente secuencia de lecturas:

20,6 / 47,4 / 74,3 / 102,4 / 149,0

Se pide efectuar el orden de ejecución, y los cálculos del tiempo demorado en posicionarse para cada una de las lecturas (y el total) para los métodos

SSTF

SCAN

Ejercicio M6: 22

Poseemos una unidad de disco rígido con 500 cilindros, divididos en sectores de 1024 bytes y 2 caras.

Considerando que un bloque es igual a un sector, la asignación es contigua, y que :

- Los archivos del sistema, tablas del sistema, etc. se encuentran desde el comienzo del disco (bloque 0), y ocupan los primeros 3.891.200 bytes.
- Luego hay un bloque libre de 48.128 bytes.

- c. Luego existe una tabla de base de datos de 55.000 registros, con los siguientes campos: código (15 bytes), descripción (60 bytes), e importe (15 bytes)

Un registro no puede dividirse en dos bloques. Está tabla posee, además, un bloque de encabezamiento.

- d. Luego de éste archivo existen 2.890 sectores libres.
e. Luego un archivo binario que ocupa 62.800 bytes.
f. Un área libre de 200

Se pide :

- a. Tamaño total del disco (bloques / tamaño).
b. Construir la tabla de bloques libres (por lista de bloques libres contiguos).
c. Determinar el porcentaje de espacio libre del disco.
d. Calcular la cantidad de sectores por pista (en el disco)

Ejercicio M6: 23

Se posee un disco rígido con las siguientes características físicas:

- 2 platos
- 300 pistas por plato
- 18 sectores por pista
- Formateado con Interleave de 0
- La operación de escritura o lectura de un sector demora 1 ms.
- Tiempo para pasar entre pistas 1 ms
- El disco cuenta con un buffer de 1 sector y el tiempo necesario para enviar la información desde el buffer hasta el controlador es de 18 ms.
- Tiempo punta a punta 10 ms.

En un determinado momento se encuentran encoladas las siguientes peticiones (Plato-Pista-Sector):

- P1-30-12 / P0-200-9 / P1-25-0 / P1-88-15 / P1-202-2 / P0-72-4

Se pide que efectúe el orden de ejecución y los tiempos demorados en cada solicitud para el método SSTF, teniendo en cuenta que recién termina de leer en P1-80-3, y el brazo del disco se encuentra en sentido ascendente.

Ejercicio M6: 24

Dado un programa que debe leer totalmente un archivo secuencial de 9740 registros de 75 bytes cada uno, grabados sobre un disco cuyos sectores son de 512 bytes, se pide :

- a. ¿Cuántas operaciones de E/S ejecuta el programa para terminar?
b. ¿Cuántas operaciones de E/S son necesarias para leer desde el registro 1050 al 1900?
c. ¿Cuántas operaciones de E/S serían necesarias si el sector del disco es de 256 bytes?
d. Si el archivo fuese de acceso directo y el programa lo trata como tal, ¿cuántas operaciones de E/S se lanzan si el programa quiere leer todos los registros al azar?

Ejercicio M6: 25

Un disco cuenta con 200 pistas, 15 sectores diagramados geométricamente con interleave 1 y gira a 4000 rpm. El tiempo necesario entre una lectura y otra (tiempo de canal) es de 1 ms. Demora 2 ms para cambiar de una pista adyacente a otra, y 18 ms para ir de una punta a la otra del disco. La cabeza se encuentra en la posición 25/8 (pista/sector) y la anterior pista leída fue la 21. Le llegan las siguientes peticiones de lectura: 10/0, 130/5, 22/12, 22/13, 2/14. Calcular los tiempos de acceso para cada lectura y el tiempo total para el método C-SCAN.

Ejercicio M6: 26

Módulo 7: Administración de la información.

I-NODOS

Realicemos un repaso de los tipos de archivos que reconoce UNIX.

- **Ordinarios:** Son los típicos archivos de los usuarios, para almacenar información o para realizar otra función, pero básicamente no tienen propiedades particulares.
- **Directorios:** Son también un tipo de archivo pero que son propiedad del SO y que poseen derechos e información adicional para poder ser reconocidos como tales y poder manejar los archivos contenidos en ellos.
- **De enlace:** Son los que permiten hacer vinculaciones dentro del sistema de archivos. Existen dos tipos de archivos de enlace:
 - **Enlace duro:** No permite hacer vínculos entre distintos filesystem; solo permite en archivos dentro de una misma partición.
 - **Enlace blando:** Permite hacer enlaces entre distintos filesystem ya que los archivos son referencias al archivo original. Pueden existir estos tipos de archivos incluso si no existe el archivo original.
- **Pipes:** Son archivos que permiten la comunicación entre procesos. Existen también dos tipos de estos archivos:
 - **Nombrados:** Permiten comunicar a procesos que no se conocen o son independientes.
 - **Sin nombre:** Permiten comunicar a procesos que se relacionan entre sí, como ser un proceso padre y su hijo.
- **Especiales:** Son archivos que representan a todo dispositivo en el sistema. Estos dispositivos pueden estar orientados a bloques (utilizan buffer) o a caracteres (utilizan cola de caracteres)

Ya sea cualquiera de estos archivos, UNIX los maneja mediante i-nodos. Estos son estructuras de control que tienen la información necesaria para poder manipular a los archivos.

Hay que tener en claro que varios archivos pueden referenciar a un mismo i-nodo, pero un i-nodo solo controla un archivo y un archivo es siempre controlado por un solo i-nodo. Basándonos en esto, podemos ver que el tamaño máximo de un archivo estará determinado por la capacidad de direccionamiento que tenga el i-nodo. En cambio, el tamaño máximo de la partición del filesystem estará relacionada con la capacidad máxima que yo tenga de almacenamiento.

Los i-nodos: son estructuras que se basan en punteros, estos pueden ser:

- **Directo** (apuntan directamente al bloque de datos),
- **Indirecto** (apunta a un bloque, el cual tiene punteros que apuntan directamente a los datos),
- **Doblemente indirecto** (apunta a un bloque, el cual tiene punteros y cada uno de ellos apunta a nuevo bloque basado también en punteros que apuntan a los bloques de datos) y
- **Triplemente indirecto** (igual que el doblemente indirecto, solamente que se agrega un nivel más de bloques).

Para calcular la capacidad de direccionamiento del i-nodo, nos bastaría con saber cuánto pueden direccionar los punteros directos, cuánto los indirectos, cuánto los doble y triplemente indirectos, y sumando todo, obtendríamos el tamaño máximo del archivo. Para calcular esta capacidad de direccionamiento por cada tipo de puntero contamos con una fórmula (cuya explicación queda para razonar), que es la siguiente:

$$\text{CAPACIDAD} = T_b \times (T_b / T_p)^n \times \text{cant. de Punteros de } n$$

Donde n tendrá el valor:

- = 0 ; si estamos calculando la capacidad de los punteros directos
- = 1 ; si estamos calculando la capacidad de los punteros indirectos
- = 2 ; si estamos calculando la capacidad de los punteros doblemente indirectos
- = 3; si estamos calculando la capacidad de los punteros triplemente indirectos

En cuanto al tamaño del filesystem, solo tenemos que calcular la capacidad de almacenamiento que poseemos en nuestros discos. Los discos pueden adoptar diferentes organizaciones, la cuales se denominan RAID y se dividen en 6 niveles:

- **NIVEL 0:** *Un disco lógico y n discos físicos. Las n primeras stripes del disco lógico representarán las primeras stripes de cada uno de los discos físicos.*
- **NIVEL 1:** *Utiliza la técnica de espejo, es decir, duplica la información. Por lo tanto, siempre habrá un número par de discos, lo cual no significa que todos ellos estén disponibles. Solamente se podrá utilizar la mitad de la cantidad total de discos ya que la otra mitad contendrá la misma información duplicada.*
- **NIVEL 2:** *Utiliza un código de detección y corrección de errores (el código de Hamming)*
- **NIVEL 3:** *Cambia el código de Hamming por un bit de paridad, el cual los almacena en discos de paridad.*
- **NIVEL 4:** *Utiliza un solo disco de paridad, donde guarda los bits por stripes. Es decir, para las primeras stripes de todos los discos calcula un stripe de paridad que lo guarda en este último disco de paridad.*
- **NIVEL 5:** *Es idéntico que el nivel 4, pero en vez de tener todas las strips de paridad agrupadas en un solo disco, distribuye las bandas de paridad por los diferentes discos.*

EJEMPLOS DE EJERCICIOS RESUELTOS:

Ejercicio RM7: 1 FINAL UTN del 07/12/2002

Un sistema de archivos utiliza una política de asignación de espacio indexada. En este sistema, los bloques de datos de un archivo, ¿se pueden ubicar de forma contigua? ¿Es necesario que los bloques estén ubicados de forma contigua? ¿Da igual que lo estén? Justifique con mucha claridad su respuesta. ¿Cambiaría en algo su contestación si la asignación de espacio fuera enlazada?

RESOLUCIÓN

La asignación indexada permite que los bloques de datos de un archivo se encuentren dispersos por el disco, pero evidentemente no obliga a ello. Así que los bloques de un archivo indexado pueden estar contiguos.

Sin embargo, no es indiferente el que los bloques estén contiguos. Los accesos secuenciales al archivo serán más eficientes en ese caso, porque el desplazamiento de los cabezales del disco será menor, comparado con el recorrido que tendría que hacer el cabezal si los bloques estuvieran desperdigados por la superficie del disco. La

contigüidad ocasiona un menor tiempo de acceso. En el mejor caso, todos los bloques del archivo estarían en un mismo cilindro y la cabeza lectora no tendría que hacer ningún movimiento para leer el archivo completo.

Si la asignación de espacio fuera enlazada, la respuesta sería similar. Incluso tendría más impacto en el rendimiento el hecho de que los bloques estén contiguos, ya que en la asignación enlazada cualquier acceso al archivo, ya sea secuencial o ya sea directo, exige recorrer los enlaces de los bloques. Si los bloques están contiguos, el cabezal del disco hará un recorrido más pequeño.

Ejercicio RM7: 2 FINAL UTN del 11/12/2004

Un Sistema Operativo utiliza organización encadenada para la gestión de los archivos. La maquina donde corre este Sistema utiliza un disco rígido que gira a 6000 RPM (Revoluciones Por Minuto), el cual posee 3 platos, formando un total de 6 cabezas, 10 sectores por pista y un total de 300 cilindros. Considere que el tamaño de un registro es igual a un sector y es de 512 bytes. Los sectores se recorren en sentido ascendente.

Existe un proceso que en un momento dado deberá ejecutar el siguiente código (sin errores de compilación):

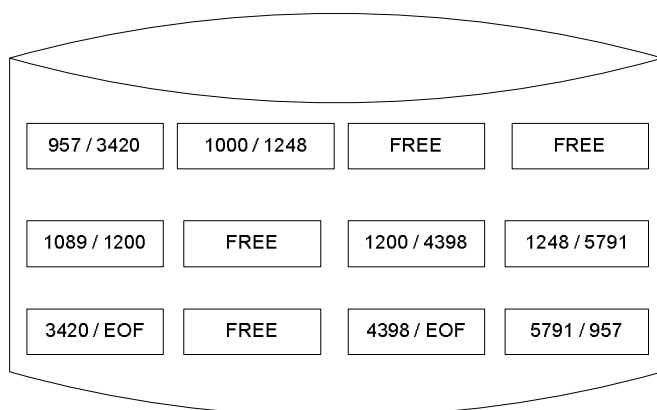
```
archivo = fopen("/home/guest/final.bin", "wb");
For (i = 0 ; i < (filesize(archivo) / size (REGISTRO)) ; i ++)
```

Bytes = fread (dato, archivo, size (REGISTRO));

A su vez se cuenta con un FAT (File Allocation Table) como la que se describe a continuación:

File Name	Start Block (Dirección lógica)	Lenght (En cantidad de Bloques)
File A	1089	10
final.bin	1000	5

El siguiente gráfico muestra la organización del disco en un instante determinado:



TIP: La estructura de bloques se encuentra dada por la Dirección lógica del bloque / (Barra) Puntero de 32 bits cuyo contenido es una dirección lógica.

Sabiendo que el algoritmo de planificación del disco es el C-SCAN, que no se utiliza el programa "Disk Defragmenter" para la compactación del disco y que la cabeza se encuentra en la dirección lógica 5000 dirigiéndose a la 7100, se pide:

- A. El tiempo que tardó el proceso en ejecutar el código, considerando **ÚNICAMENTE** el tiempo que se insume en atender los pedidos a disco.
- B. La traducción de las direcciones lógicas a físicas a las cuales hace referencia el proceso. **Deduzca** la fórmula que le permite realizar dichas traducciones.

RESOLUCIÓN

6000 rev. Por min. = 10 rev. Por segundo = 10 mseg una revolución. Entonces para leer un sector = 1 mseg.

Dir. lógica	Cilindro	Cabeza	Sector
1000	16	4	0
1248	20	4	8
5791	96	3	1
957	15	5	7
3420	57	0	0
5000	83	2	0
7100	118	2	0

Si consideramos C-SCAN, el orden de lectura de las direcciones lógicas será:

DIR. LOG.(CILINDRO, SECTOR)

5000(83,0); 5791(96,1); 7100(118,0) (**); 17999 (299,9)(*); 0(0,0)(**); 957(15,7); 1000(16,0); 1248(20,8); 3420(57,0)

(*) LLEGA AL MAXIMO DEL DISCO (CILINDRO =299, SECTOR = 9) Y LUEGO CAE BRUSCAMENTE (**) AL CILINDRO = 0, SECTOR = 0. Considerandose este tiempo de caída, nulo.

Del enunciado se desprende que no se tiene en cuenta esta consideración y si solo lo que tarda en ejecutar el código (Punto A), POR LO QUE NO SE LEE (*),(**) Y (***) y no se tiene en cuenta la planificación del brazo.

EL ORDEN SERÁ: 5000(83,0); 5791(96,1); 3420(57,0); 1248(20,8); 1000(16,0); 957(15,7);

5000(83,0); ⇒ 5791(96,1); 13 cil.+ 3 sectores + (7+1) sectores queda posicionado (96,2) = 21 mseg
 5791(96,1); ⇒ 3420(57,0); (39 cil.+0, (queda pos. 59,1 por lo que debiera esperar por 9 sect)) = 48 mseg
 3420(57,0); ⇒ 1248(20,8); (37 cil.+7, (queda pos. 20,7 por lo que debiera esperar por 1 sect)) = 38 mseg
 1248(20,8); ⇒ 1000(16,0); (4 cil.+0, (queda pos. 16,3 por lo que debiera esperar por 8 sect)) = 12 mseg
 1000(16,0); ⇒ 957(15,7); (1 cil.+0, (queda pos. 15,2 por lo que debiera esperar por 5 sect)) = 6 mseg

para ejecutar el código se necesitan 131 mseg

D) la formula sería:

dir física = (dir logica) / ((cant. de cabezas / cil.) * (Sectores / pista) ⇒ parte entera = cant de cil

Ejercicio RM7: 3 Final UTN del 29/05/06

B3. Se dispone de un disco de 30 MB de capacidad formateado para que trabaje con un sistema de archivos tipo UNIX cuyas características se describen a continuación:

1. Tamaño de bloque 512 bytes
2. Tamaño de la dirección de bloque: 4 bytes
3. Número de i-nodos: 500
4. Campos del i-nodo:
 - Atributos del archivo (496bytes)
 - 2 punteros directos
 - 1 puntero indirecto simple
 - 1 puntero indirecto doble

Se pide:

- a) ¿Qué tamaño máximo podrá tener un archivo en este sistema de archivos?
- b) ¿Es posible crear enlaces físicos (Hard Links) en este sistema de archivos?
- c) ¿Qué ventajas tiene la alocaión indexada respecto de la alocaión contigua?

RESOLUCIÓN:

¿Qué tamaño máximo podrá tener un archivo en este sistema de archivos?

$$T_{MAX} = [2 + 1 \times (512/4) + 1 \times (512/4)^2] = 8.455.168 \text{ bytes o } 8257 \text{ kiby o } 8,06 \text{ Meby}$$

¿Es posible crear enlaces físicos (Hard Links) en este sistema de archivos?

SI, PORQUE EL I-NODO PROVEE BYTES PARA ATRIBUTOS DEL ARCHIVO, COMO SER EL CONTADOR DE LOS LINKS

¿Qué ventajas tiene la alocaión indexada respecto de la alocaión contigua?

PERMITE ACCEDER DE MANERA NO SECUENCIAL A LOS REGISTROS MAS RÁPIDOS: NO ES NECESARIO MANTENER EL ORDEN DE ACCESO.

_____ ○ ○ ○ _____

Ejercicio RM7: 4

Calcular el tamaño máximo que puede tener un Archivo en Unix considerando un tamaño de bloque de 512 bytes. Las direcciones de bloque son de 4 bytes.

Resolución:

Se debe calcular el número de bloques que pueden direccionarse desde un bloque índice. Como el tamaño de bloque es 512 bytes, y las direcciones son de 4 bytes, obtenemos $512/4 = 128$ direcciones por bloque.

Entonces el número de bloques será: $10 + 128 + 1282 + 1283$.

El tamaño máximo se obtiene, entonces, multiplicando el número de bloques por el tamaño de cada bloque (512 bytes), lo cual nos da un tamaño aproximadamente de 1GB.

_____ ○ ○ ○ _____

Ejercicio RM7: 5

Con un sistema de archivos, en UNIX, con un tamaño de inodo de 128 bytes, un tamaño de bloque de 1024 bytes y donde la zona de inodos ocupa 2048 bloques, ¿cuántos bloques ocupa el mapa de bits de inodos libres?

Resolución:

En primer lugar se calcula cuantos inodos caben en un bloque:

Entonces, $1024(\text{bytes por bloque}) / 128 (\text{bytes por inodo}) = 8 \text{ inodos por bloque}$.

El número total de inodos será entonces $2048 * 8 = 2^{11} * 2^3 = 2^{14}$ inodos.

Por cada inodo, existe un bit en el mapa de bits de inodos. De esta forma, el tamaño del mapa de bits será: $2^{14} \text{ bits} / 2^3 \text{ bits/byte} = 2^{11} \text{ bytes} = 2^{11} \text{ bytes} / 2^{10} \text{ bytes por bloque} = 2 \text{ bloques}$.

_____ ○ ○ ○ _____

Ejercicio RM7: 6

El espacio libre del disco se puede gestionar mediante una lista de bloques libres o mediante un mapa de bits. Suponiendo que las direcciones del disco requieren d bits, y que el disco tiene b bloques, de los cuales l están libres, indicar en que condiciones la lista de bloques libres utiliza menos espacio que el mapa de bits.

Resolución:

El mapa de bits corresponde a una lista de bits en el que cada uno indica si un bloque está ocupado o libre. Por lo tanto, su tamaño será igual a tantos bits como número de bloques exista e independiente del número de bloques libres, es decir b bits.

La lista de bloques libres es una lista enlazada de bloques en el que cada uno contiene tantas direcciones de bloques libres como pueda. Si no se consideran los punteros de enlace entre bloques de la lista, el tamaño que ocupa la lista de bloques libres será lo que ocupa cada dirección por el número de bloques libres, es decir, $d * l$ bits.

La condición para que la lista de bloques libres ocupe menos espacio que el mapa de bits es:

$$d * l < b.$$

_____ ○ ○ ○ _____

Ejercicio RM7: 7 Ejercicio de Algoritmo de traducción de direcciones en un sistema de archivos:

Suponga un sistema de archivos que gestiona el espacio de disco empleando asignación enlazada o encadenada. **Describe algorítmicamente** el proceso de traducción de una dirección de archivo a nivel de usuario, expresada en forma de desplazamiento en bytes desde el origen del archivo, a dirección física de disco. Asuma los siguientes datos:

- ◆ Disco de 2 caras con 80 pistas y 32 sectores por pista. El tamaño de sector es de 512 *bytes*.
- ◆ El sistema de archivos gestiona el espacio para archivos mediante bloques de 1024 *bytes*, de los que 2 *bytes* se utilizan como enlace.

Se supone que la información de control del archivo (File Control Block) ya se encuentra disponible en memoria.

Resolución al Ejercicio

Sean:

- * **Offset_Archivo** = Puntero del archivo (dato de entrada)
- * **Bloque_Inicio** = Bloque de comienzo del archivo
- * **Long_punt** = Longitud del enlace en los bloques de datos
- * **"/"** = Operador de división entera
- * **"%"** = Operador de resto

Se quiere obtener:

- * Superficie, que tomará un valor entre 0 y 1
- * Cilindro, que tomará un valor entre 0 y 79
- * Sector, que tomara un valor ente 0 y 31

/* Obtenemos bloque relativo de sistema de archivos */

Bytes_Efectivos = 1024 - Long_punt

Bloque_relativo = Offset_archivo / Bytes_Efectivos;

Offset_Bloque = Offset_archivo % Bytes_Efectivos;

/* Obtenemos bloque absoluto de sistema de archivos */

Bloq_Absoluto = Bloque indicado en la entrada de directorio

Acceder a Bloq_Absoluto

Para Bloq=1 hasta Bloque_relativo Hacer

Bloque_Siguiente = enlace existente en Bloq_Absoluto

Bloq_Absoluto = Bloque_Siguiente

Acceder a Bloq_Absoluto

Fin Para

/* Obtenemos sector lógico */

Sector_Logico = Bloque_Absoluto + Offset_Bloque / 512;

/* Finalmente obtenemos dirección física */

Sectores_Cilindro = 64

Cilidro = Sector_Logico / Sectores_Cilindro;

Superficie = (Sector_Logico % Sectores_Cilindro) / 32;

Sector = (Sector_Logico % Sectores_Cilindro) % 32;

Nota: Los cálculos del último paso dependen de la regla seguida para asignar sectores lógicos a sectores físicos. En este algoritmo se utiliza la siguiente: se empieza por el primer cilindro y primera superficie, cuando ésta se completa se pasa a la segunda superficie. Cuando se completan las dos superficies de un cilindro se repite el esquema de asignación con el siguiente cilindro y así sucesivamente. Recuerde que este asunto es importante, pues influye en los tiempos de acceso a los datos de los archivos almacenados en el disco. Se propone como ejercicio plantear una regla más eficiente que la descrita.

Ejercicio RM7: 8

El sistema de administración de archivos propuesto permite la lectura de un mismo archivo por varios trabajos en forma concurrente o sólo uno escribiendo. Si se quisiera permitir procesos leyendo y

escribiendo en forma concurrente, ¿debería implementarse algún tipo de protección contra interferencias a fin de preservar la integridad de los datos?. Se pide:

- a. Modifique el lenguaje de control propuesto al principio de la práctica para permitir este tipo de accesos.
- b. ¿Qué tipo de problemas pueden presentarse?
- c. Debe tener el sistema de archivos conocimiento de la organización interna del archivo para poder efectuar una protección eficiente del archivo? ¿Por qué?
- d. ¿En qué punto de la secuencia de operaciones desencadenadas por una solicitud de entrada/salida efectuaría a Ud. los controles de prevención de interferencia?

_____ ○ ○ _____

Ejercicio RM7: 9

EJERCICIOS SIN RESOLVER

Ejercicio M7: 1

Se puede simular una estructura multinivel de directorios en una estructura de un solo nivel?. Considere que no hay restricción en la longitud de los nombres. Si la respuesta es afirmativa, explique como lo puede llevar a cabo, y compare ésta estructura con la de múltiples niveles. Si la respuesta es negativa, explique por qué no se puede llevar a cabo. Cómo alteraría su respuesta si el nombre está limitado a siete (7) caracteres.

○ ○ ○ _____

Ejercicio M7: 2

Dada la siguiente asignación de bloques libres en formato Bit Vector, construya una tabla de bloques libres contiguos.

110011000011001011110010010

○ ○ ○ _____

Ejercicio M7: 3

Indique cuál es el estado del sistema de archivos si tienen las siguientes estructuras (todas las respuestas deberán estar justificadas, y en caso de existir algún error indique cual sería la implicancia para el sistema.):

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Bloques en uso	1	1	0	1	0	1	1	1	1	0	0	1	1	1	0	0
Bloques libres	0	0	1	0	1	0	0	0	0	1	1	0	0	0	1	1

b)

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Bloques en uso	1	1	0	1	0	1	1	1	1	0	0	1	1	1	0	0
Bloques libres	0	0	0	0	1	0	0	0	0	1	1	0	0	0	1	1

c)

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Bloques en uso	1	1	0	1	1	1	1	1	1	0	0	1	1	1	0	0
Bloques libres	0	0	1	0	1	0	0	0	0	1	1	0	0	0	1	1

d)

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Bloques en uso	1	1	0	1	0	2	1	1	1	0	0	1	1	1	0	0
Bloques libres	0	0	1	0	1	0	0	0	0	1	1	0	0	0	1	1

○ ○ ○ _____

Ejercicio M7: 4

Poseemos una unidad de disco rígido con 500 cilindros, divididos en sectores de 512 bytes y 3 caras

Considerando que un bloque es igual a un sector, la asignación es contigua, y que :

- j. Los archivos del sistema, tablas del sistema, etc. se encuentran desde el comienzo del disco (bloque 0), y ocupan los primeros 3.891.200 bytes.
- k. Luego hay un bloque libre de 20.992 bytes.
- l. A continuación de esto hay un archivo de texto de 7.213 caracteres.
- m. A continuación existen 19.968 bytes sin usar
- n. Luego existe una tabla de base de datos de 25.000 registros, con los siguientes campos: código (5 bytes), descripción (9 bytes), e importe (6 bytes). Un registro no puede subdividirse en dos sectores. Esta tabla posee, además, un bloque de encabezamiento.
- o. Luego de éste archivo existen 3.304 sectores libres.

Se pide :

- e. Tamaño total del disco (bloques / tamaño).
- f. Construir la tabla de bloques libres (por lista de bloques libres contiguos).
- g. Determinar el porcentaje de espacio libre del disco.
- h. Calcule la cantidad de sectores existentes por pista (en el disco).

○ ○ ○ _____

Ejercicio M7: 5

Dados los siguientes l-nodos, estructura de bloques libres (mantenida en forma ordenada, y suponiendo que por bloque físico entran 20 direcciones de bloques libres, contando el puntero al próximo bloque), y

Directorio, se pide que analice el estado actual del File System (la columna de inicio deberá ser completada por Ud., luego de hacer el punto 2). Si encuentra errores justifique la respuesta. Solo se representan el área de datos, las del sistema no están dentro de los datos presentados.

Directorio:

Archivo	Inicio	I-nodo
Tato		0
Tito		1
Toto		2
Tuto		3
Tutu		4
Tete		0

I-NODOS

I-Nodo 0	10	I-Nodo 1	20	I-Nodo 2	21	I-Nodo 3	I-Nodo 4
5	25	7	98	40	50	11	13
6	33	8		41	51	3	15
2		90		42	52	9	16
14		91		43	53	4	17
0		92		44			18
80		93		45			71
81		94		46			
22		95		47			
23		99		48			
24		1		49			
10		20		21			

Lista de bloques libres:

Pos.	0	96	97
0	12	58	78
1	19	59	79
2	26	60	82
3	27	61	83
4	28	62	84
5	29	63	85
6	30	64	86
7	31	65	87
8	32	66	88
9	34	67	89
10	35	68	97
11	36	69	
12	37	71	
13	38	72	
14	39	73	
15	54	74	
16	55	75	
17	56	76	
18	57	77	
19	96	97	

- 1) Dada la estructura anterior, se pide que la transforme en una estructura FAT, con la administración de bloques libres a través de Bit Vector. En caso de que el punto anterior tenga algún error se debe realizar el traspaso de la versión corregida.
- 2) Arme una tabla con cada uno de los archivos existentes en el file system donde haga la relación entre los sectores lógicos, y los sectores físicos (considere una relación 1 a 1).

Ejercicio M7: 6

Si considera que se está utilizando un SO de la familia UNIX, en una implementación de I-nodo con direcciones de 64 bits.

1. ¿Cuál es el tamaño máximo que puede tener un archivo almacenado en un disco cuyos sectores son 2KB?
2. ¿Cómo es el i-nodo que tendría que representar un archivo en el disco (debe ser dibujado)?

3. ¿Qué tamaño mínimo debería tener el disco para que sea factible guardar un archivo con todo el i-nodo completo?, teniendo en cuenta que el super bloque ocupa 10 MB y que además de ese archivo también se van a almacenar 78 archivos de texto, de 40 bytes cada uno.

_____ ○ ○ ○ _____

Ejercicio M7: 7

Dada la siguiente FAT, estructura de Bit Vector, y Directorio, se pide que analice el estado actual del File System. Si encuentra errores justifique la respuesta. Solo se representan el área de datos, las del sistema no están dentro de los datos presentados.

Directorio:

Archivo	Inicio	I-nodo
Tato	5	
Tito	7	
Toto	40	
Tuto	11	
Tutu	13	
Tete	5	

FAT

--	0	1	2	3	4	5	6	7	8	9
0	80	98	14	9	EOF	6	2	8	90	4
1		3		15	0	16	17	18	70	
2			23	24	25	33				
3				EOF						
4	41	42	43	44	45	46	47	48	49	50
5	51	52	53	EOF						
6										
7	EOF									
8	81	22								
9	91	92	93	94	95	99			EOF	1

Bit Vector:

--	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9
0	1	1	1	1	0	1	1	1	1	1	0	1	0	1	1	1	1	1	1	0
2	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	0	0	0	1

_____ ○ ○ ○ _____

Ejercicio M7: 8

Si se desea recorrer un archivo de principio a fin, es más eficiente realizarlo en un Sistema que implementa listas enlazadas que uno que implementa asignación continua. V o F. Justificar.

_____ ○ ○ ○ _____

Ejercicio M7: 9

La alocaación contigua de archivos lleva a una fragmentación interna del disco. V o F. Justificar.

Ejercicio M7: 10

Si se quiere acceder al último byte de un archivo extremadamente grande, es más conveniente realizarlo en un Sistema de Archivos FAT que en un Sistema de Archivos tipo UNIX. V o F. Justificar.

Ejercicio M7: 11

Se tiene un Sistema el cual utiliza un Filesystem del tipo UNIX para la gestión de los archivos. La maquina posee 8 discos rígidos formando un RAID 1, donde cada uno de ellos tiene una capacidad de almacenamiento de 40 GB. A su vez, existe un archivo denominado Inter.bin con la siguiente estructura de datos:

- Legajo (10 caracteres)
- Nombre (40 caracteres)
- Edad (Integer)
- Materias Aprobadas (Integer)

Considerando que lo máximo que se puede direccionar en este filesystem es de 24.000 **KB** y que el tamaño de un cluster es de 256 KB con bloques de 50 KB, se pide:

- a) La cantidad de accesos al disco necesarios para leer el campo "Materias Aprobadas" del registro 280 del archivo Inter.bin que se encuentra en el directorio /home/guest/sisop, sabiendo que el i-nodo correspondiente no se encuentra cargado en memoria principal.

El tamaño del filesystem y el tamaño máximo teórico de un archivo. ¿Cuál es la diferencia entre el tamaño máximo teórico y el real?. **Justifique.**

○ ○ ○

Ejercicio M7: 12

Sea un Sistema de archivos tipo UNIX que gestiona bloques de datos de 24 Bytes. Cada i-nodo, además de otra información, contiene 10 punteros directos a bloques de datos, un puntero de indirección simple y un puntero de indirección doble.

Suponiendo que tenemos una cache de 20 bloques de datos y otra de 20 i-nodos inicialmente vacías, que el i-nodo del directorio raíz se encuentra en memoria principal y que sólo ejecuta en el Sistema un proceso, se pregunta:

- a) ¿Cuántos accesos al disco son necesarios para leer el byte 299, sabiendo que el tamaño total del archivo es de 3,9 KB?
- b) Teniendo en cuenta los datos calculados en el punto anterior. ¿Cuántos accesos al disco son necesarios para leer hasta el byte 299?

○ ○ ○

Ejercicio M7: 13

Si tengo dos discos, uno de 25 G y otro de 2G en un sistema UNIX, con tamaño Bloque 4 k y punteros de 64 bits Asuma que hay 20 punteros directos a bloques, y que hay una indirección simple, una doble y una triple en cada I-Nodo.

- a) ¿Cuánto espacio en KB en el disco ocupa un archivo de 150000 registros de 2kb cada uno. ?
- b) ¿Cuál es el tamaño máximo de un archivo en este sistema y cual es el tamaño el file system?
- c) Si este archivo estuviera en un sistema Fat32 cuanto espacio en disco ocuparía.

○ ○ ○

Ejercicio M7: 14

Dada la siguiente FAT, estructura de Bit Vector, y Directorio, se pide que analice el estado actual del File System (la columna de I-nodo deberá ser completada por Ud., luego de hacer el punto 2). Si encuentra errores justifique la respuesta. Solo se representan el área de datos, las del sistema no están dentro de los datos presentados.

Directorio:

Archivo	Inicio	I-nodo
Tato	5	
Tito	7	
Toto	40	
Tuto	11	
Tutu	13	
Tete	5	

FAT

--	0	1	2	3	4	5	6	7	8	9
0	80	98	14	9	EOF	6	2	8	90	4
1		3		15	0	16	17	18	70	
2			23	24	25	33				
3				EOF						
4	41	42	43	44	45	46	47	48	49	50
5	51	52	53	EOF						
6										
7	EOF									
8	81	22								

	9	91	92	93	94	95	99			EOF	1
--	---	----	----	----	----	----	----	--	--	-----	---

Bit Vector:

--	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9
0	1	1	1	1	0	1	1	1	1	1	0	1	0	1	1	1	1	1	1	0
2	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	0	0	0	1

- a) Dada la estructura FAT, anterior, se pide que la transforme en una estructura de i-nodos, con la administración de bloques libres a través del método Lista de bloques libres (mantenida en forma ordenada, y suponiendo que por bloque físico entran 20 direcciones de bloques libres, contando el puntero al próximo bloque). En caso de que el punto anterior tenga algún error se debe realizar el traspaso de la versión corregida.
- b) Arme una tabla con cada uno de los archivos existentes en el file system donde haga la relación entre los sectores lógicos, y los sectores físicos (considere una relación 1 a 1). Basándose en la tabla anterior conteste cada una de las siguientes preguntas (tenga en cuenta que las estructura FAT, Bit Vector, I-List se encuentran mapeadas en memoria. De la lista de bloques libres solo se encuentra en memoria el primero de los bloques):

Pregunta	FAT/BV		I-nodos	
	Sec. Fís.	Acc. Mem.	Acc. Disco	Acc. Mem.
a) Leer el sector lógico 7 del archivo Tato				
b) Leer el sector lógico 13 del archivo Toto				
c) Leer el sector lógico 11 del archivo Tito				
d) Leer el secuencialmente el archivo Tutu				
e) Crear el archivo Teto con 7 posiciones reservadas en forma contigua (marcar en las tablas de alguna forma que se distinga los lugares ocupados, o generar el I-nodo según corresponda).		Indicar los pasos realizados para seleccionar el área en cada una de las administraciones (indicando los accesos a memoria y disco en cada caso)		
f) Crear el archivo Tati con 26 posiciones contiguas (ídem anterior)		Ídem anterior		

Ejercicio M7: 15

Considerando que se tiene un FS que tiene 400 posiciones en su tabla FAT32 (y las correspondientes en el bit vector), y que el sector ocupa 1 KB, ¿cuál es el tamaño del archivo más grande que se podría almacenar? Indique cómo y porque realizó los cálculos para justificar su respuesta.

Ejercicio M7: 16

- 1) Se desea grabar un archivo de 25.200.000 bytes de longitud, cuyo registro lógico tiene 125 bytes, y el factor de bloqueo es de 30 (o sea 30 registros lógicos es un registro físico). Se dispone de una unidad de disco de 8 superficies, cada superficie tiene 100 pistas, y cada pista tiene 63 sectores de 512 bytes cada sector. Considere que el disco sólo almacena ese archivo, y que no existen directorios, boot sector, u otra cosa. Además considere que un registro no puede ocupar 2 sectores distintos. El disco gira a 360 r.p.m., un procesador graba mediante la técnica de E/S por interrupciones (Una interrupción por byte). Cada interrupción lleva 2.5 nano segundos de procesamiento. El buffer de E/S es de 32256 bytes. Se pide :
- Qué ocurre si se desea agregar un registro al archivo ?
 - Cuál es el porcentaje de tiempo que le dedica el procesador para atender la E/S ?
 - Ídem a b), pero utilizando la técnica por DMA. (Considere una interrupción por sector)

Ejercicio M7: 17

Se desea grabar un archivo de 12.800.000 bytes de longitud, cuyo registro lógico tiene 250 bytes. Se dispone de una unidad de disco de 8 superficies, cada superficie tiene 100 pistas, y cada pista tiene 16

sectores de 1024 bytes cada sector. Considere que el disco sólo almacena ese archivo, y que no existen directorios, boot sector, u otra cosa. Además considere que un registro no puede ocupar 2 sectores distintos. Se pide :

- Qué ocurre si se desea agregar 250 bytes al archivo?
- Cual es el porcentaje de utilización del disco, y cual es el porcentaje de fragmentación interna (si hay).

Ejercicio M7: 18

- Indique cual es el tamaño máximo que puede tener un archivo en un sistema operativo que arma una estructura de I-nodo, y el cual cuenta con las siguientes características: Bloque de 1 KB., el I-nodo tiene diez bloques directos, tres bloques simplemente indirectos (de 256 posiciones cada uno), dos bloques doblemente indirecto, y un bloque triplemente indirecto.
- Basándose en el sistema anteriormente descrito, y suponiendo que solo puede haber un archivo en el File System, se pide que indique la cantidad de memoria que será necesaria para almacenar la FAT (el tipo de dato utilizado es un DOUBLE – 8 bytes), y para el Bit Vector.

_____ ○ ○ ○ _____

Ejercicio M7: 19

- Se tiene una FAT que administra un File System que ocupa 16.911.440 bytes en memoria (considerando que cada dato ocupa 8 bytes), se pide que indique cuantas entradas existen en la tabla, cuantas entradas deberá tener el Bit Vector, y cuanta memoria se necesitará para almacenarlo.
- Considerando la FAT anterior, diseñe un I-nodo que pueda soportar la misma cantidad de entradas en un solo archivo. Tenga en cuenta que no se pueden tener más de 10 bloques directos, ni más de 10 bloques indirectos (entre los simples, dobles y triples), y que cada indirección almacena hasta 128 registros.

_____ ○ ○ ○ _____

Ejercicio M7: 20

Ud. es el encargado de diseñar un I-nodo que cumpla con las siguientes características:

- El FS trabajará con bloques de 1024 KB. como mínimo.
 - Dentro de la I-Lista deberán existir al menos 80000 I-Nodos. El tamaño total de la I-Lista no puede superar los 16000 bloques.
 - El I-nodo tiene que tener direcciones de 64 bits.
 - Se necesitan almacenar un total de 10 datos numéricos (de 64 bits cada uno) utilizados para la administración además de los datos de direcciones e indirecciones.
 - El archivo mínimo a soportar por el I-Nodo tiene que ser de entre 2.1 GB. y 3.0 GB.
 - Por cuestiones de performance se determinó que no se puede tener una indirección de más de 3 niveles, pero si se podrían tener más de una indirección del mismo nivel.
- Se pide que cree el I-nodo, detalle la estructura y cuanto ocupa cada parte.
 - Dibuje el I-nodo que soporta un archivo de 1.8 GB. Indicando que indirecciones están ocupadas y cuales no.
 - Ahora se desea duplicar el tamaño del archivo máximo. Explique por lo menos tres formas de realizarlo y las ventajas y desventajas de cada una.
 - Suponiendo que le asignan un disco con 900000 sectores de 1024 bytes y se necesita diseñar la forma de administrar espacio libre, teniendo como premisa que no más del 1% de los sectores se pueden usar para almacenar la lista de bloques libres dentro del área de catálogo. Se pide que indique por lo menos tres formas de implementar la lista de bloques libres, con sus ventajas y desventajas.

_____ ○ ○ ○ _____

Ejercicio M7: 21

Se dispone de un disco rígido formateado a bajo nivel con interleave 1 y sectores de 1024 bytes. Si instalamos un SO de la familia Unix, ¿Cuántos I-Nodos necesitará un archivo de 1 GB para ser almacenado en ese disco? Justifique su respuesta.

_____ ○ ○ ○ _____

Ejercicio M7: 22

Módulo 8: Seguridad y Protección

En cuanto a seguridad debemos tener en cuenta los requerimientos que deben ser protegidos por la misma:

- **Secreto o privacidad:** La información debe ser confidencial para las personas que no sean autorizadas
- **Integridad:** Las “posesiones” del sistema no pueden ser modificadas por personas que no estén autorizadas
- **Disponibilidad:** Los elementos del sistemas deben estar disponibles para toda persona autorizada al uso de los mismos
- **Autenticidad:** Debe asegurarse la autenticidad de las personas que ingresan al sistema; un ejemplo clásico es mediante las passwords.

Por supuesto que estos requerimientos son atacados constantemente por diferentes amenazas, dichas amenazas afectarán tanto a uno como a varios de estos aspectos según la función que adopte esta amenaza. Para verlo mas claro, podemos decir que tenemos 4 tipos de amenazas en un sentido general:

- **Intercepción:** Afecta sobre todo al secreto. Un flujo de información es interceptado por un ente que no está autorizado.
- **Modificación:** Factor clave que afecta a la integridad del sistema. Los datos, o incluso el hardware es modificado para acceder a la información.
- **Interrupción:** Ataca a la disponibilidad. La información no llega a destino ya que se interrumpe en el camino.
- **Fabricación:** Un ente no autorizado crea o inventa información que la hace pasar como si fuese auténtica; obviamente afecta a la autenticidad.
- **Robo:** es el caso de los programas malignos que extraen información de un sistema y lo envían a un dado destino. Afecta a la privacidad.

Hasta ahora pareciera ser que solo la información del sistema es afectada por las diferentes amenazas, y si bien es una de las posesiones del sistema que suele ser atacada con mayor frecuencia, no es la única. Tanto el hardware, como el software, las líneas de comunicaciones y las redes son usualmente afectadas por diferentes tipos de amenazas. El hardware se ve amenazado principalmente desde el aspecto de la disponibilidad, el software suele correr el riesgo de ser borrado, modificado (integridad) y también peligra su uso no autorizado (secreto). En cuanto a las líneas de comunicación y las redes podemos hacer una diferencia entre lo que son las amenazas activas y las amenazas pasivas. Las primeras, las amenazas activas, suelen atacar contra la disponibilidad y la integridad; ejemplos clásicos son la modificación o creación de mensajes. En cuanto a la segunda clase de amenazas, las pasivas, son aquellas que suelen atacar contra el secreto; son del tipo de escuchas, es decir, no modifican información pero es accedida por personas no autorizadas.

Ahora bien, como el sistema operativo debe ocuparse de mantener la seguridad en el sistema, debe adoptar alguna técnica para cumplir con los requisitos que ésta plantea. Dentro de todas las posibilidades que hay, aquí veremos una de las mas utilizadas que es la de la **matriz de acceso**. Para ello, debemos definir tres términos:

- **objeto:** cualquier cosa a la cual se controla su acceso.
- **sujeto:** entidad capaz de utilizar objetos.
- **derechos de acceso:** la manera en que un sujeto puede hacer uso de un objeto.

Con estos tres conceptos, el sistema operativo arma una matriz referenciando a los **recursos** por columna y los **dominios** por cada fila; en las intersecciones se ubican los derechos que tiene dicho dominio para tal objeto. De esta forma queda organizada una matriz de acceso, pero también debemos tener en cuenta los términos que hacen referencia a lo que se denomina “**lista de control de acceso**” y “**lista de capacidades**”. Estas no son ni mas ni menos que distintos puntos de vista de ver a la matriz: si armamos una lista por cada dominio (tomamos cada una de las filas por separado) obtendremos una **lista de capacidades**. Por otro lado, si armamos una lista por cada objeto (tomamos ahora cada una de las columnas por separado) obtendremos listas de control de accesos.

Otra forma de controlar los accesos a los recursos que tiene el sistema operativo es mediante el uso de contraseñas o claves. En cuanto a las **contraseñas**, podemos comenzar mencionando que existen diferentes estrategias de selección de las mismas:

- **Educación al usuario:** Darle reglas al usuario para la elección de passwords. Explicarle la importancia de seleccionar contraseñas que mezcle diferentes caracteres (blancos, números,

etc.), que no sean contraseñas relacionadas con direcciones o nombres conocidos, que no sean excesivamente cortas, evitar palabras de diccionarios, y así diferentes consideraciones.

- **Generadas por computadora:** Son difíciles de adivinar, lo cual es ventajoso, pero también son difíciles de recordar, lo cual tienta al usuario a anotarlas, cosa sumamente indeseable.
- **Detector Reactivo:** Periódicamente el sistema ejecuta un programa para adivinar passwords y detecta si posee contraseñas vulnerables, es decir, fáciles de adivinar.
- **Detector Proactivo:** En el momento que el usuario elige su password, el sistema controla que se cumpla con las reglas establecidas; caso contrario, rechaza la contraseña.

Por supuesto que las contraseñas deben estar guardadas en algún sitio para que cuando un usuario desee ingresar al sistema se pueda controlar la password ingresada con la ya almacenada. Para evitar un uso indeseado de este archivo que contiene contraseñas, se pueden tomar dos medidas: una es la de controlar que personas tienen acceso a dicho archivo, y por otro lado tenemos la **técnica de cifrado o encriptación**. Encriptar una clave no es ni mas ni menos que modificarla mediante un algoritmo de tal manera que no pueda ser visible su contenido.

Uno de estos algoritmos de encriptación es el DES (Data Encryption Estándar). Esta función tiene dos inputs: un texto a encriptar de 64 bits y la clave de 56 bits. Los bloques mayores de texto se encriptan en bloques de 64 bits. El DES procesa el texto a través de 16 iteraciones, produciendo un valor intermedio de 64 bits al final de cada iteración. Cada una de estas iteraciones es la misma función compleja que involucra una permutación de los bits y sustituye un patrón de bit por otro. El input de cada paso es la salida del paso anterior más una permutación en los bits clave, y esta permutación se conoce como subclave. En el caso específico de las claves en UNIX, cada usuario elige una clave de hasta 8 caracteres. Esta clave es convertida a un valor de 56 bits que sirve de input a una rutina de encriptación, la cual es una modificación del algoritmo DES. Esta modificación se basa en tomar un valor de 12 bits llamado SALT. Este valor guarda relación con la hora en que la password se asignó al usuario.

De esta forma podemos ver que el algoritmo DES se basa en realizar una permutación en los bits, esta idea nos da pie a clasificar dos tipos de encriptación de claves:

- **Encriptado convencional o simétrico:** Su algoritmo se basa en operaciones simples sobre patrones de bits. Por otro lado, en este tipo de cifrado, se utiliza la misma clave para cifrar como para descifrar el mensaje. Como hemos visto, el DES es un ejemplo de este tipo de algoritmo.
- **Encriptado de clave pública o asimétrico:** Su algoritmo se basa en operaciones matemáticas y no en operaciones sobre los bits. Como diferencia podemos decir también que este algoritmo utilizar una clave para cifrar los mensajes y otra para descifrarlos.

Finalmente vamos a decir también que la encriptación por clave pública soluciona la distribución de la clave; en el caso del encriptado simétrico es necesario la transmisión de la clave ya que con ésta se cifra y descifra el mensaje. El sentido de utilizar una clave privada y una clave pública en los cifrados asimétricos es que permite la autenticación de un mensaje; es decir, la clave privada lo que hace es "firmar" el mensaje. Cuando éste se recibe, utilizando la clave pública del remitente es posible asegurar la procedencia del mensaje.

EJEMPLOS DE EJERCICIOS RESUELTOS:

Ejercicio RM8: 1 FINAL UTN del 02/10/2003

1. Dada la siguiente situación se requiere administrar la protección para garantizar la política definida.

- El dominio esta definido por el grupo de usuarios.
- Las operaciones permitidas son **Read**, **Write** y **eXecute**.
- Hay un administrador: Carlos.
- Hay dos docentes de Sistemas Operativos: Graciela y Pepe.
- Hay dos docentes de arquitectura: Maria y Jose.
- Hay tres Alumnos: Diego, Matias y Dario.
- Los grupos son: Administradores (Admins), Docentes de S.O (Doc-SO), Docentes de Arquitectura (Doc-Arq) y cada alumno tiene su propio grupo: Diego, Matias y Dario.
- Se tienen 12 archivos:

Nombre	Dueño	Nombre	Dueño	Nombre	Dueño
Horarios	Carlos	R-KUMANA-B	Matias	ARQ1	maria
TPSO1	Graciela	R-KUMANA-C	Matias	ARQ2	Jose
TPSO2	Pepe	R-KUMANA-D	Diego	R-DPACMAN1	Dario
R-KUMANA-A	Dario	R-KUMANA-E	Diego	R-DPACMAN2	Dario

- El administrador puede leer, modificar y ejecutar todos los archivos suyos y puede leer los de todos.
- Los docentes de SO pueden leer, modificar y ejecutar todos los archivos de los docentes de SO y pueden leer y ejecutar todo los de sus alumnos.
- Los docentes de arquitectura solo pueden leer, modificar y ejecutar todos los archivos de los docentes de arquitectura y pueden leer todos los de sus alumnos.
- Cada alumno puede leer, modificar y ejecutar todos los archivos suyos y puede leer los de los docentes.

Se pide que arme la matriz y proponga la implementación por medio de ACL (Access Control List) , C-List (capability List) y bits de protección tipo **UNIX**.

RESOLUCIÓN

Nombre	CARLOS	GRACIELA	PEPE	MARIA	JOSE	DIEGO	MATIAS	DARIO
Horarios	RWX			RWX	RWX	R	R	R
TPSO1	R	RWX	RWX			R	R	R
TPSO2	R	RWX	RWX			R	R	R
R-KUMANA-A	R	RX	RX	R	R			RWX
R-KUMANA-B	R	RX	RX	R	R		RWX	
R-KUMANA-C	R	RX	RX	R	R		RWX	
R-KUMANA-D	R	RX	RX	R	R	RWX		
R-KUMANA-E	R	RX	RX	R	R	RWX		
ARQ1	R			RWX	RWX	R	R	R
ARQ2	R			RWX	RWX	R	R	R
R-DPACMAN1	R	RX	RX	R	R			RWX
R-DPACMAN2	R	RX	RX	R	R			RWX

Ejercicio RM8: 2

Dada la siguiente matriz de acceso, indique los permisos que tienen cada uno de los usuarios en el sistema:

	F1	F2	F3	Imp. láser	D1	D2	D3	D4
U1	read		read			switch		
U2			write	print	write		switch	switch
U3		read	execute					
U4	read/write		read/write		switch			

RTA Punto 2 os permisos deben ser:

U1: F1,F3, Read; D2, Switch

Etc.

Bart y Lisa, luego de la negativa de Chester de reinvertir el dinero obtenido en seguir produciendo dibujos animados, empezaron a investigar y a buscar los medios de volver a producir los dibujos animados hasta que por fin hallaron una estrategia para salvar a sus personajes favoritos.

El problema era como enviarle la respuesta a Roger Meyers Jr estando seguros de garantizar los requerimientos de confidencialidad, autenticidad e integridad.

Se sabe que el texto plano a enviar es un archivo muy grande y que el medio de transmisión disponible es inseguro.

Los niños Simpson estaban confundidos sobre como poder hacer para cumplir con todos estos requerimientos, ya que ninguno de ellos todavía había cursado Sistemas Operativos. Es por eso que te pedimos que realices un diagrama explicando los pasos necesarios para poder cumplir con todos los requerimientos de seguridad pedidos y garantizando que la solución sea optima en cuanto al consumo de recursos del sistema, así los niños de Springfield podrán volver a ver sus caricaturas favoritas.

SOLUCION

EMISOR A

AEA ⇒ Alg. Encr. Asimétrico

AES ⇒ Alg. Encr. Simétrico

CS ⇒ Clave Sesión

CSE ⇒ Clave Sesión Encriptada

CPu ⇒ Clave Pública

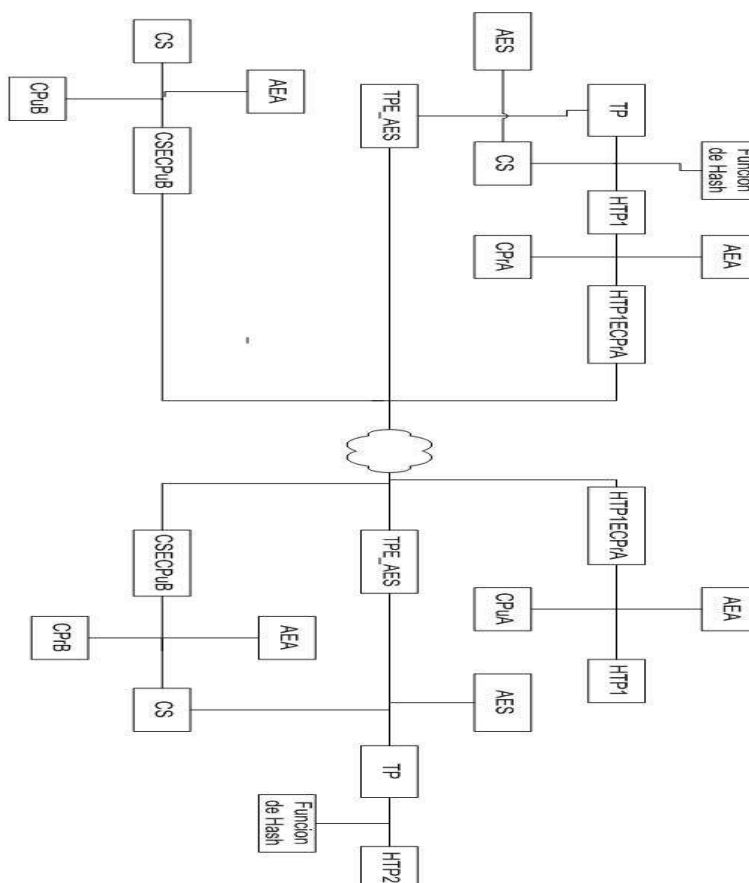
CPr ⇒ Clave Privada

HTP ⇒ Hash del Texto Plano

TP ⇒ Texto Plano

TPE ⇒ Texto Plano Encriptado

RECEPTOR B



EMISOR A

1. Se genera un hash del texto plano (HTP1). Esto permitirá que se cumpla el requerimiento de integridad.
2. Mediante la aplicación del algoritmo de encriptación asimétrica y la clave privada del emisor se encripta el hash del texto plano (HTP1ECPrA). Esto permitirá que solo se pueda desencriptar el hash con la clave pública del emisor, lo que permite que se cumpla el requerimiento de autenticidad.
3. Aplicando un algoritmo de encriptación simétrica y una clave de sesión se encripta el texto plano (TPE_AES).
4. Para enviar la clave de sesión por el vínculo inseguro y poder garantizar el requerimiento de confidencialidad se usa un algoritmo de encriptación asimétrica y la clave pública del receptor. (CSECPuB)
5. Se transmite mediante el vínculo inseguro: HTP1ECPrA, TPE_AES, CSECPuB

RECEPTOR B

1. Recibe mediante el vínculo inseguro: HTP1ECPrA, TPE_AES, CSECPuB.
2. Aplicando un algoritmo de encriptación asimétrica y la clave pública del emisor desencripta el HTP1ECPrA para obtener HTP1.
3. Aplicando un algoritmo de encriptación asimétrica y la clave privada del receptor se desencripta la clave de sesión (CS) que se usará para desencriptar el TPE_AES.
4. Aplicando un algoritmo simétrico y la CS se obtiene el texto plano (TP).
5. Se genera un hash del texto plano (HTP2). Esto permitirá que se cumpla el requerimiento de integridad.

REQUERIMIENTOS:

Autenticidad: El receptor puede garantizar que el emisor es quien dice ser ya que solo el emisor conoce la clave privada utilizada aplicada al HTP1 para generar el HTP1ECPrA, por lo que solo puede ser desencriptado usando la clave pública del emisor.

Confidencialidad: Solamente el receptor conoce su clave privada para poder obtener la clave de sesión necesaria para descryptar el TPE_AES. Como el CSECPuB fue encriptado usando la clave publica del receptor, se puede garantizar que este será el único capaz de descryptarlo, al aplicar su clave privada.

Integridad: Este requerimiento se obtiene de comparar el HTP1 con el HTP2.

Nota: No es correcto aplicar el algoritmo de encriptación asimétrico con la clave privada del emisor sobre el texto plano y luego el algoritmo de encriptación asimétrico con la clave publica del receptor, debido a que los algoritmos de encriptación asimétrico tienen la desventaja de un alto consumo de recursos, por lo que la solución no sería óptima.

Ejercicio RM8: 6 Final 19 – 12 - 2009

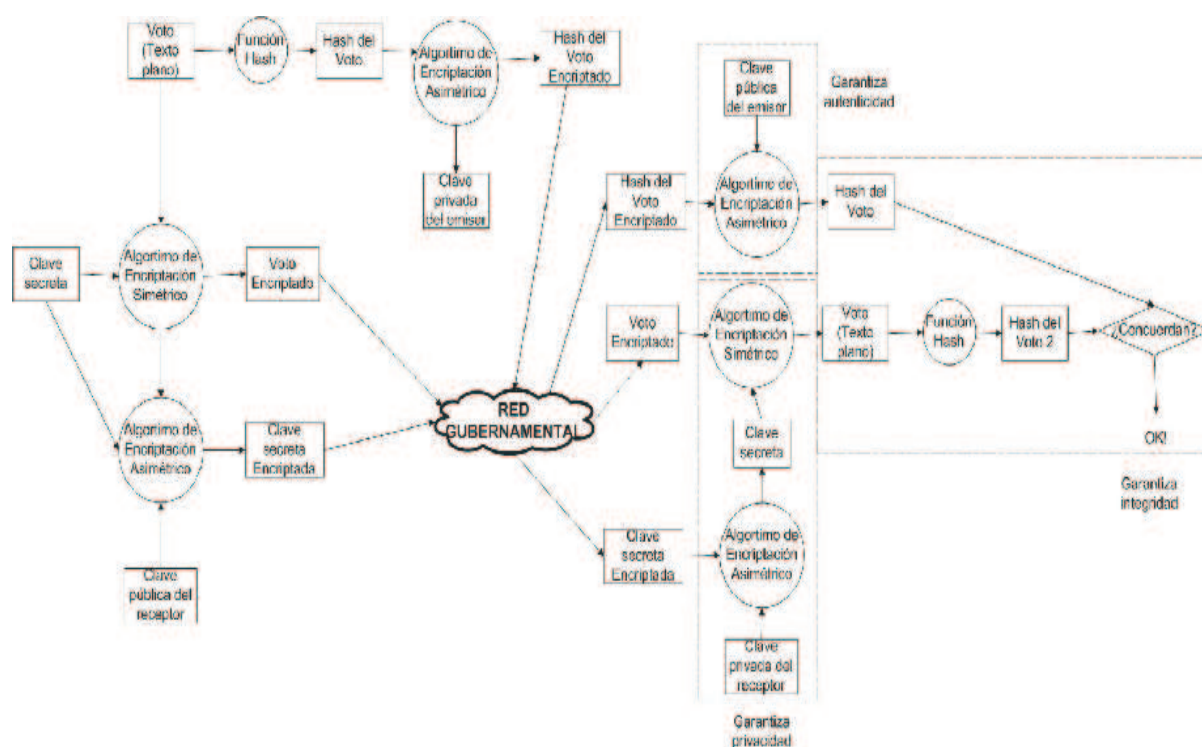
El gobierno de la ciudad de Buenos Aires desea implementar el sistema de voto electrónico para las próximas elecciones gubernamentales. Demanda como características primordiales que los datos del voto sean íntegros al momento de computarlos, que se pueda verificar la autenticidad de éste y obviamente que mantenga su privacidad.

¿Qué método criptográfico utilizaría para la tarea? Desarrolle a través de un diagrama. Indique cómo se cumplen los requisitos pedidos.

Solución: Dadas las características mencionadas, lo más adecuado en este caso sería implementar el método de sobre digital, ya que satisface justamente la integridad, autenticidad y privacidad del texto plano.

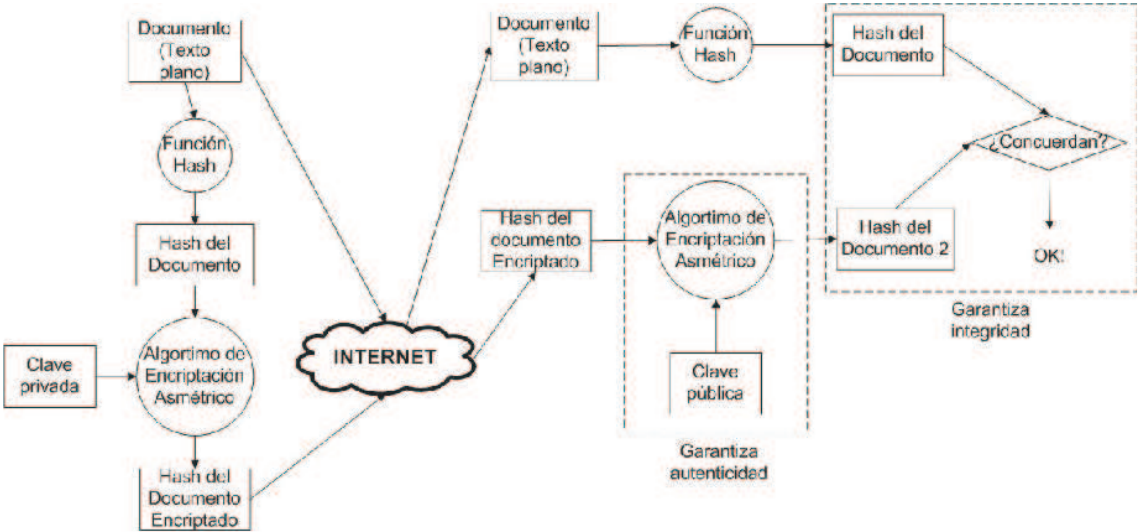
Ejercicio RM8: 7 Final 07 – 03 - 2010

Un grupo de investigación científico argentino vuelca los avances del proyecto en el que está involucrado en un documento de texto. Éste deber ser enviado a otro equipo de investigación residente en el exterior vía Internet. Si se pretende que el documento sea recibido con sus datos íntegros y que se pueda comprobar la autenticidad del mismo,



¿Qué método criptográfico utilizaría para la tarea? Desarrolle a través de un diagrama. Indique cómo se cumplen los requisitos pedidos.

Solución: Dadas las características mencionadas, lo más adecuado en este caso sería implementar el método de firma digital, ya que satisface justamente la integridad y autenticidad del texto plano.



EJERCICIOS SIN RESOLVER

Ejercicio M8: 1

Cifre por sustitución el siguiente texto usando la clave repetida NAC, y aplicando como función la suma:
 Espacio=00; A=01; C=03; D=04; E=05; G=07; I=09; N=14; O=15; P=16; R=18; S=19; T=20; U=21;
 Y=25
 Sí (TEXTO+CLAVE) => 27 Entonces CIFRADO=(TEXTO+CLAVE) – 27
 Cifrar el texto: SEGURIDAD Y PROTECCION
 ¿Cuál o cuáles son las debilidades de este método

Ejercicio M8: 2

Dada la siguiente matriz de acceso, indique los permisos que tienen cada uno de los usuarios en el sistema:

	F1	F2	F3	Imp. láser	D1	D2	D3	D4
U1	read	write	read			switch		
U2	read		write	print	write		switch	switch
U3	read	read	execute			write		
U4	read/write	write	read/write	print	switch			

Ejercicio M8: 3

Transforme la matriz de acceso del punto anterior en los siguientes métodos:

- Tabla global.
- Lista de accesos.
- Lista de capacidades.
- Mecanismo Llave / Cerradura (Lock / Key).

Ejercicio M8: 4

Supongamos un sistema en el cual el derecho al uso de terminales está distribuido de la siguiente manera:

Alumnos	00:00 a 08:00 hs
Investigadores	07:00 a 24:00 hs
Admin. Sistema	00:00 a 24:00 hs

Construya la matriz de accesos e implemente la Tabla Global, las Listas de Accesos, las Listas de Capacidad y el Lock-Key. ¿Cuál es la más eficiente?.

Ejercicio M8: 5

Realizar el cifrado de la palabra ABEJA por:

- El método de sustitución con la palabra clave "SISTEMAS OPERATIVOS". Detalle los pasos que realiza.
- El método de cifrado por sustitución para el n=4. Detalle los pasos realizados.

Ejercicio M8: 6

Dados los siguientes algoritmos de encriptación de datos, indique las ventajas, desventajas de cada uno. Dentro de la respuesta se debe realizar un ejemplo de funcionamiento de la rutina, realizando la prueba de pisaron de la corrida. Cada una de las rutinas debe ser comentada y documentada. Indique a su entender cual de las dos rutinas es mejor:

a. Rutina "A"

```
void encriptar (char *buf, char *clave, int cpal)
{
    int i,j=0,n=0;

    for(i=0;i!=cpal;i++)
    {
        if (clave[j]=='\0')
            j=0;
        buf[i]=buf[i]^clave[j];
    }
}
```

```

        j++;
    }
    if (clave[j]!='\0')
        j=0;
    buf[i]=buf[i]^clave[j];
}

```

b. Rutina "B"

```

#include <string.h>
#include <stdlib.h>
int Funcion1(char* txt)
{
    int lar;
    lar=strlen(txt);
    {
        char linea[lar];
        int i;
        strcpy(linea,txt);
        for(i=0;i<=(strlen(txt)-1);i++)
            txt[i]=linea[(strlen(txt)-1)-i];
    }
    return (0);
}

int Encriptar(char *txt,int key)
{
    int i;
    srand(key);
    key=rand();
    Funcion1(txt);
    for(i=0;i<=strlen(txt)-1;i++)
        txt[i]=(txt[i]+key);
    return (0);
}

```

○ ○ ○

Ejercicio M8: 7

En un Sistema X la protección se orienta a la información, es por esto que se posee la siguiente matriz de accesos:

	P0	P1	P2	P3
S1	R	R	RW	RW
S2	RW	R	R	R
S3	R	RW	R	RW

Se pide determinar en cuantas formas se puede agrupar esta tabla y como quedaría organizada según esos criterios.

○ ○ ○

Ejercicio M8: 8

En un sistema UNIX Linux, existe una tabla de usuarios, que se encuentra protegida, donde se registran los usuarios del sistema y sus respectivos passwords. Al sistema ingresan los siguientes usuarios, formando la siguiente tabla.

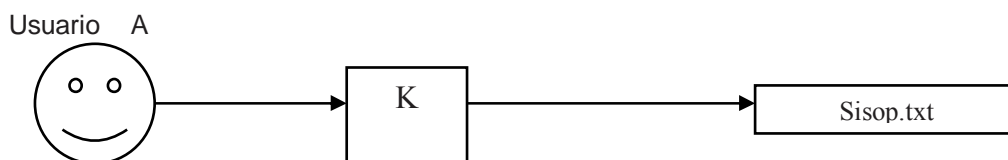
Usuarios	Contraseña
Diego Maradona	boca
Gabriel Batistuta	fiorentina
Ariel Ortega	river
Javier Saviola	river

Es valida esta tabla? Se encuentra esta tabla completa? Es posible que dos usuarios tengan la misma contraseña? Justificar.

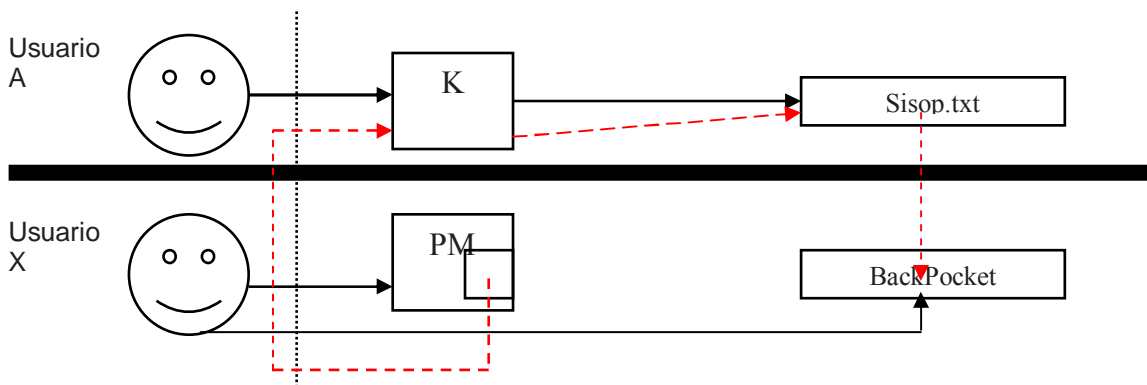
○ ○ ○

Ejercicio M8: 9

En un sistema existen distintos niveles de usuarios, a los cuales se les otorgan distintos permisos. En este sistema el usuario A, es el único con acceso a "sisop.txt", el cual lo hace a través del proceso K.



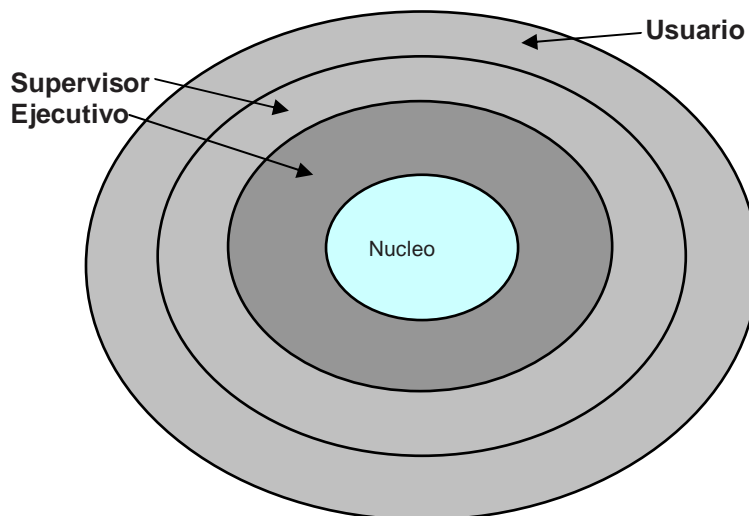
Luego un usuario X, ejecuta un programa malicioso que realiza lo siguiente



Se pide identificar el programa malicioso por su forma de operar y graficar como operaría sobre un trusted system, si es posible que opere.

Ejercicio M8: 10

El esquema del VMS/VAX suele conocerse como estructura de protección de anillo, como muestra el gráfico.



La desventaja principal de esta estructura es que no permite el principio de “tener que conocer”. Más concretamente, si un objeto tiene que estar accesible en un dominio D_j , pero no en un dominio D_i , entonces debe cumplir que $j < i$. Pero esto significa que todos los segmentos accesibles en D_i también son accesibles en D_j .

Existe alguna estructura que basada en este sistema solucione el problema anteriormente dicho.

Ejercicio M8: 11 FINAL 18/02/2006

Dos usuarios X e Y disponen de las siguientes cuatro herramientas: a) Algoritmo de hashing H unidireccional de generación de claves en base a un documento; b) Un algoritmo S de encriptación de claves simétricas; c) Un algoritmo A de encriptación de claves asimétricas; d) Una red de comunicaciones pública que los vincula.

X desea enviarle un documento D a Y. Ese envío debe asegurar que solo X e Y pueden leer el contenido de D, debe asegurarle a Y que el origen es X y debe asegurar la consistencia de D. Explique paso a paso

las acciones que realizan los dos usuarios para lograr que Y obtenga a D con las condiciones anteriormente detalladas. En su explicación aclare como se van cumpliendo las condiciones. Para que se considere correcto el punto deberá garantizar UNICAMENTE lo que se pide. Si se garantiza más o menos de lo pedido, se considerará el punto EN SU TOTALIDAD como incorrecto.

Ejercicio M8: 12

Algunos sistemas reescriben las áreas que fueron ocupadas por archivos ya borrados. ¿Bajo qué esquema es esto necesario?

Ejercicio M8: 13

¿Cuál es la principal diferencia entre Lista de Acceso y Lista de Capacidad?

Ejercicio M8: 14

En un sistema de Administración de Memoria Paginada por Demanda, ¿cuándo es necesario utilizar claves de protección para las Páginas / Bloques?

Ejercicio M8: 15

Supongamos un sistema en el cual cada proceso tiene asociado un número n y cada objeto un número m . Si se quiere que un proceso pueda acceder a un objeto sólo si $n > m$, ¿qué tipo de protección se debe implementar?

Ejercicio M8: 16

Supongamos un sistema en el cual la Capacidad de Acceso a un objeto le está permitida sólo n veces. ¿Qué esquema de protección implementaría?

Ejercicio M8: 17

Si todos los derechos de acceso a un objeto fueron eliminados, el objeto puede ser también eliminado. ¿Qué implementación utilizaría?

Ejercicio M8: 18

Las listas de Capacidad son generalmente guardadas dentro del espacio de direcciones del programa. ¿Qué implementación utilizaría?

Ejercicio M8: 19

Utilice alguna clave criptográfica para el método DES para cifrar la palabra SECRETO escrita en ASCII. Verifique su descifrado.

Ejercicio M8: 20

Verifique para algún mensaje que el método RSA funciona con $p=7$, $q=13$, $e=5$, $d=29$.

Ejercicio M8: 21

Explique claramente quién posee la capacidad (dominio u objeto) en los casos:

- Lista de Accesos
- Lista de Capacidad
- Lock-Key

Ejercicio M8: 22

La lista de todas las passwords se mantiene dentro del sistema operativo. Luego, si un usuario administra en forma de lectura tal lista el sistema de protección de passwords se torna muy débil. Sugiera un esquema que impida este problema (ayuda: utilice una representación interna que difiera de la representación externa).

_____ ○ ○ ○ _____

Ejercicio M8: 23

Supóngase que las contraseñas se eligen como combinaciones de 4 caracteres entre 26 caracteres alfabéticos. Supóngase que un adversario es capaz de intentar contraseñas a razón de una por segundo.

- Suponiendo que no hay realimentación al adversario hasta que se completa cada intento, ¿cuál es el tiempo que se estima necesario para descubrir la contraseña correcta?
- Suponiendo que hay realimentación al adversario que es notificado con cada caracter incorrecto que introduzca, ¿cuál es el tiempo que se estima necesario para descubrir la contraseña correcta?

_____ ○ ○ ○ _____

Ejercicio M8: 24

Supóngase que unos elementos de origen de longitud k se traducen de alguna forma uniforme a elementos de destino de longitud p . Si cada dígito puede tomar uno de entre r valores, el número de elementos de origen es r^k y el número de elementos de destino es el número menor r^p . Un elemento de origen concreto x_i se traduce a un elemento de destino concreto y_j .

- ¿Cuál es la probabilidad de que el adversario seleccione el elemento correcto de origen en un intento?
- ¿Cuál es la probabilidad de que el adversario genere un elemento de origen x_k diferente ($X_i \neq X_k$) que resulte en el mismo elemento de destino y_j ?
- ¿Cuál es la probabilidad de que el adversario genere el elemento de destino correcto en un intento?

_____ ○ ○ ○ _____

Ejercicio M8: 25

Un generador fonético de contraseñas toma dos segmentos aleatorios de cada contraseña de seis letras. La forma de cada segmento es CVC (consonante, vocal, consonante), donde

$$V = \langle a, e, i, o, u \rangle \text{ y } C = \overline{V}$$

- ¿Cuál es la población total de contraseñas?
- ¿Cuál es la posibilidad de que un adversario adivine una contraseña correctamente?

_____ ○ ○ ○ _____

Ejercicio M8: 26

La necesidad de la regla de "no leer arriba" para un sistema de seguridad multinivel es bastante obvia. ¿Cuál es la importancia de la regla de "no escribir abajo"?

_____ ○ ○ ○ _____

Ejercicio M8: 27

Dar algunos ejemplos donde el análisis de tráfico podría comprometer la seguridad. Describir situaciones donde el cifrado de extremo a extremo combinado con el cifrado de enlaces seguirían permitiendo que el análisis de tráfico fuera peligroso.

_____ ○ ○ ○ _____

Ejercicio M8: 28

La necesidad de seguridad y el deseo de compartir los datos han estado siempre refiidos uno con el otro. Cuanta mas gente accede a las redes de computadores de miles de nodos, ¿cuáles son las implicaciones de los gusanos y virus en esta interacción creciente? Considérese no sólo las ramificaciones técnicas, sino también el impacto potencial en la comunicación humana.

_____ ○ ○ ○ _____

Ejercicio M8: 29

Los esquemas de distribución de claves que emplean un centro de control de acceso y/o un centro de distribución de claves tienen puntos centrales vulnerables a los ataques. Discutir las implicaciones en la seguridad de dicha centralización.

_____ ○ ○ ○ _____

Ejercicio M8: 30

La protección de los usuarios de una red contra gusanos y amenazas similares, ¿debería ser responsabilidad de la propia red o de las máquinas que la utilizan? Razónese la respuesta.

☐ ☒ ☐

Ejercicio M8: 31

Debido a los riesgos conocidos del sistema de contraseñas de UNIX, la documentación del SunOS 4.0 recomienda que el archivo de contraseñas sea eliminado y sustituido con un archivo públicamente leíble llamado /etc/publickey. Una entrada en el archivo para el usuario A consta de un identificador del usuario, ID_A , la clave pública del usuario, KU_A y la clave privada correspondiente, KR_A . Esta clave privada se cifra mediante el DES con una clave obtenida de la contraseña de conexión del usuario, P_A . Cuando A se conecte al sistema, descifra $E_{P_A}[KU_A]$ para obtener KU_A .

- a) El sistema verifica entonces que P_A fue suministrada correctamente. ¿Cómo lo hace?
b) ¿Cómo puede atacar el sistema un adversario?

☐ ☒ ☐

Ejercicio M8: 32

Se dijo que la inclusión de la base en el esquema de contraseñas de UNIX aumenta la dificultad de adivinar una en un factor de 4.096. Pero la base está almacenada en claro en la misma entrada que la contraseña cifrada correspondiente. Por tanto, dichos caracteres son conocidos para el atacante y no tienen que ser adivinados. Entonces, ¿por qué se afirma que la base aumenta la seguridad?

☐ ☒ ☐

Ejercicio M8: 33

Suponiendo que se ha respondido correctamente al problema anterior y se ha comprendido la importancia de la base, aquí va otra cuestión. ¿No sería posible frustrar por completo a todos los averiguadores de contraseñas incrementando drásticamente el tamaño de la base hasta, por ejemplo, 24 ó 48 bits?

☐ ☒ ☐

Ejercicio M8: 34

Supóngase que las contraseñas están limitadas a emplear 64 caracteres (A-Z, a-z, 0-9, "." y "/") y que todas las contraseñas son de ocho caracteres de longitud. Supóngase un averiguador de contraseñas con una tasa de cifrado de 14,4 millones de cifrados por segundo.

- a) ¿Cuánto tardará en probar exhaustivamente todas las contraseñas posibles de un sistema UNIX?
b) Si un archivo de contraseñas de UNIX contiene N contraseñas cifradas, ¿cuál es el número esperado de adivinaciones hasta encontrar, al menos, una contraseña?
c) ¿Cuanto tardará en llevar a cabo el número esperado de adivinaciones?

☐ ☒ ☐

Ejercicio M8: 35

Necesitamos encriptar la frase SITUACIÓN DE DEUDORES utilizando el algoritmo de palabra clave. La palabra clave a utilizar es NECESIDAD, y el valor de las letras es 0 para el espacio, 1, para la A, y así sucesivamente (considere solo las letras del alfabeto inglés). Se pide que realice el encriptado.

NO SE TOMARÁN EN CUENTA AQUELLOS EJERCICIOS EN DONDE SOLO SE COLOQUE LA FRASE RESULTANTE DEL ENCRIPADO, SINO QUE ES OBLIGATORIO QUE SE COLOQUE LA FRASE ORIGINAL, EL VALOR DE CADA UNA DE LAS LETRAS, ETC., Y SE CONVIERTA EL RESULTADO A LA LETRA QUE CORRESPONDE.

☐ ☒ ☐

Ejercicio M8: 36

EJERCICIOS COMBINADOS

EJERCICIOS RESUELTOS

FINAL UTN del 07/12/2002

1. Sea el siguiente esquema de llegada de trabajos a ser procesados en un sistema monoprocesador:

Proceso	T Llegada	Tiempo de CPU	Comienzo de E/S	Duración de E/S	Prioridad Base	Tamaño
1	0	7	2	2	BAJA	2000 k
2	1	9	3	4	MEDIA	1800 k
			8	3		
3	2	7	5	4	MEDIA	800 k
			10	4		
4	3	5	4	4	ALTA	1000 k
5	3	4	-	-	MEDIA	1700 k

- Orden de prioridad: ALTA > MEDIA > BAJA.
- Tamaño de la memoria disponible: 5.500 k.
- Cantidad de instancias de E/S: 1.
- Administración de Memoria: particiones dinámicas, con realocación cada vez que sea necesario y algoritmo de asignación de peor ajuste.
- Administración de CPU: Round Robin (Slice=3), con prioridades. Las prioridades de los procesos aumentan a medida que pasa el tiempo. El método de aumento de prioridad se realiza cuando un proceso permanece en la cola de listo sin ejecutarse, cada 4 unidades de tiempo pasa a la prioridad siguiente (envejecimiento o ageing) y así sucesivamente hasta que tome la CPU, luego de dejar la CPU vuelven a su prioridad base. Cuando un proceso cambia la prioridad por envejecimiento, entra a la cola de listo último de su nueva prioridad. En caso de simultaneidad de eventos, tiene prioridad el proceso envejecido.

Notas: - Recuerde que para acceder a la *cola de ready*, un job debe estar en memoria real, o sea, antes debe competir por el recurso *memoria*.
 - Considere que cuando un job toma un segmento de memoria, no lo libera hasta su finalización, cualquiera sea su prioridad.

Se pide que realice la traza de ejecución completa de los procesos y el mapa de memoria en los instantes correspondientes.

SOLUCIÓN Parte B

1. Solución a verificar:

a.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33
P5																																		
P4																																		
P3																																		
P2																																		
P1																																		

P1 termina en 16

P4 termina en 28

P5 termina en 29

P2 termina en 31

P3 termina en 32

b. MEMORIA (el número en paréntesis es el espacio libre)

0> P1 (3500)

1> P1, P2 (1700)

2> P1, P2, P3 (900)

16> P2, P3, P4, P5 (200)

28> P2, P3, P5 (1200)

- 19> P2, P3 (2900)
 31> P3 (4700)
 32> (5500)

Ejercicio Final del 04/03/2006

B1. Bono y The Edge se están preparando para su presentación en el estadio de River. Debido a ello, están supervisando el estado de los sistemas que controlan la consola de sonido, los cuáles tienen un sistema operativo que utiliza páginas para la administración de memoria y cuya dirección está compuesta de 32 bits, de los cuales 20 son para el índice y 12 para el desplazamiento. El sistema cuenta con 12 frames de memoria disponibles para procesos y 4 para el sistema operativo. En todo momento, se ejecutan en la consola 2 procesos. El estado es el siguiente:

P1	P2
Read(2151)	Down(B)
Down(A)	Write(13333)
Write(1536)	Up(A)
Up(C)	Write(19000)
Read(20480)	Down(C)
Down(B)	Read(560)
Read(512)	Up(B)
Up(A)	Write(195)
Write(7321)	Down(A)
Down(C)	Read(2560)
Read(10500)	Up(C)
	Read(4096)

Estado inicial de los frames

P1			P2		
Página	Frame	Presente	Página	Frame	Presente
0	1	1	0	7	1
1	4	1	1	2	1
2	10	1	2	6	1
3	5	0	3	4	1
4	6	0	4	4	0

Últimas referencias

P1	P2
(2-0-1)	(2-0-3)

Estado de los semáforos: A=C=0;B=1;Mutex

Con la información de la consola, conteste:

B.1.1) ¿Cuántos fallos de página se producen para cada proceso?:

- Si el módulo utiliza como algoritmo de víctima FIFO – Asignación fija – Alcance Global.
- Si el módulo utiliza LRU – Asignación fija – Alcance Local.

B.1.2) ¿Se puede producir deadlock?

B.1.3) ¿Terminan de ejecutar? ¿Por qué?. Indique la traza de ejecución de los procesos.

B.1.4) ¿Con qué tema empezó el show de U2?

SOLUCIÓN

Datos: direccionamiento 32 bits 20 bits para índice y 12 para offset, tamaño de pagina 4 KBy

Traducciones de direcciones lógicas a físicas

Proceso 1		
Dir física	Page	Offset
2151	0	2151
1536	0	1536
20480	5	0
512	0	512
7321	1	3225
10500	2	2308

Proceso 2		
Dir física	Page	Offset
13333	3	1045
19000	4	2616
560	0	560
195	0	195
2560	0	2560
4096	1	0

FINAL UTN del 29/07/00

El Boeing 777 de United Airlines surcaba sigilosamente los aires del mar Caribe. En uno de los asientos de la clase ejecutiva, el agente Ethan Hunt leía su libro, 'El Angel Negro', con total concentración. Todo parecía indicar que sería un viaje placentero. En ese momento, la azafata le ofrece un trago. 'No, gracias', contestó. -'Tal vez esté interesado en una película.', le dijo la azafata, con una mirada que lo decía todo. Él debía ver esa película. Una vez que la azafata se alejó, Hunt colocó la película en uno de los visores de su asiento. Un flujo de adrenalina recorrió su cuerpo al ver el comienzo de la misma. 'Sr. Hunt, veo que a logrado alcanzar su vuelo.', la voz del misterioso hombre puso nervioso a Ethan. Él ya había pasado por esto antes. 'Pero desafortunadamente, le tengo una mala noticia para usted, Sr. Hunt. Una última misión, ', continuó, 'para que se libre de todo compromiso con Sistemas Operativos.'. 'Bien que más da.', pensaba

internamente Hunt, mientras la tensión en su cuerpo crecía vertiginosamente. 'Agentes Kamtchakenses han robado un secreto militar Inglés. Estos documentos son planos para la construcción de un SODERO (Sistema Operativo Distribuido Emulado Reconfigurable On-Line), lo último en tecnología SW. Este era un secreto altamente clasificado. Suponemos que algún doble agente tuvo acceso a éste, y envió los planos a la central de Inteligencia Kamtchakense. Afortunadamente, nosotros también tenemos nuestros informantes. Sabemos que el archivo está ubicado en esa central. La misma posee como computador principal un Trekkies Wserver, del cual tenemos suficiente información como para llevar a cabo esta misión. Sabemos que este servidor, precisa cada vez que se le solicita un documento, cargarlo totalmente en memoria para poder enviarlo, por lo que posee disponible 20 frames de 4kB cada uno en memoria. La elección de la víctima es FIFO. A su vez, se encuentra funcionando en un S.O. que utiliza i-nodos (de 8 punteros en total cada uno) y direcciones de 43 bits (32 bits como identificador y el resto como offset) en su File System. Este S.O. sólo puede tener 3 i-nodos activos en memoria (Además del Superblock) siendo la política de reemplazo LIFO.

Luego de un amplio monitoreo determinamos que este Trekkies Wserver siempre ejecuta 5 procesos, en el siguiente orden y con los siguientes semáforos:

Semáforos:

A=0 H=0 N=0 P=0 T=1 W=1 X=1 Y=1 Z=1

P1	P2	P3	P4	P5
D(Y)	D(N)	D(X)	D(X)	D(N)
D(Z)	D(Z)	D(T)	D(W)	D(Z)
U(N)	D(Y)	U(X)	U(X)	D(Y)
U(Z)	U(Y)	D(H)	U(H)	U(N)
U(Y)	U(Z)	U(P)	D(P)	U(Y)
	U(N)	D(W)	D(T)	U(Z)
	D(A)	U(W)	D(X)	U(A)
		U(T)	U(T)	
			U(W)	
			U(X)	

La estructura de directorios es la siguiente:

/Starfleet

```

/utopia/P1.secretplan
| /Enterprise/P2.secretplan
| /ds9/P3.secretplan
| P4.secretplan
|
|
|

```

Sabemos también que el archivo que buscamos, tiene como nombre el último proceso en finalizar, si no hay deadlock en el resto. O el nombre del primer proceso en terminar, en el caso de que haya deadlock entre el resto de los procesos.

Además, el servidor dispone de las siguientes herramientas:

- Algoritmo de Hash
- Claves asimétricas
- Claves simétricas.

'Su misión, Sr. Hunt, si decide aceptarla, es:

1. Determinar, si existiese, el nombre del archivo que necesitamos, indicando la secuencia de finalización de los procesos y el estado de aquellos que no finalicen.
2. Suponiendo que todavía no se realizó ningún pedido, indicar cuántos bloques leídos de disco, cuántos escritos y cuántos fallos de página provocará el pedido de ese archivo. (El archivo ocupa 100 kb).
3. Indíquenos como enviaría el resultado desde ese servidor, de haberlo logrado, asegurándonos la autenticidad e integridad de su respuesta, justifique claramente su respuesta, Sr. Hunt.

Bien, conoce las reglas. Si es capturado, negaremos cualquier conexión con usted. Este mensaje se autodestruirá en Ochenta (80) minutos.'

RESOLUCIÓN:

1- TRAZA:

De cualquier forma que elijan para correr siempre termina en el siguiente orden: P1-P5-P2, quedando P3 y P4 en Deadlock. La respuesta entonces sería **P1.secretplan**.

Como ejemplo: Supongamos la traza corriendo en orden secuencial:

A	H	N	P	T	W	X	Y	Z	ESTADO
0	0	0	0	1	1	1	1	1	Inicio
0	0	1	0	1	1	1	1	1	Corre y Finaliza P1
-1	0	1	0	1	1	1	1	1	Corre P2 hasta D(A)
-1	-1	1	0	0	1	1	1	1	Corre P3 hasta D(H)
-1	0	1	-1	0	0	1	1	1	Corre P4 hasta D(P), desbloquea a P3
-1	0	1	0	0	-1	1	1	1	Corre P3 hasta D(W), desbloquea a P4
-1	0	1	0	-1	-1	1	1	1	Corre P4 hasta D(T) → DL entre P3 y P4
0	0	0	0	-1	-1	1	1	1	Corre y Finaliza P5, desbloquea a P2
0	0	0	0	-1	-1	1	1	1	Finaliza P2

				DEADLOCK				
--	--	--	--	----------	--	--	--	--

2- INODOS:

Memoria: 20 frames de 4KB cada una --> Mem Física = 80 KB, con algoritmo de Reemplazo FIFO

Disco:

Inodos de 8 punteros total (5 directos + 3 indirectos)

Direcciones de 43 bits (32 ID + 11 Offset)

→ Bloques de 2 KB

Punteros de 4 B

→ Inodo:	5 Directos de 2KB =	+ 5 x 2KB =	10 KB
	1er Indirecto =	+ (2KB/4B) x 2KB =	1 MB
	2do Indirecto =	+ (2KB/4B) ² x 2KB =	512 MB
	3er Indirecto =	+ (2KB/4B) ³ x 2KB =	256 GB

Total Direcciondo por el Inodo ≈ 256,5 GB

Se puede tener 3 inodos en memoria (reemplazo LIFO)

Se quiere cargar un archivo de 100 KB

a) Leídos:

Leer Inodo de /Starfleet:	1
Leer Inodo de /Starfleet/Utopia:	1
Leer Inodo de /Starfleet/Utopia/P1.secretplan:	1
Leer Bloques Directos (10KB):	5
Leer Bloque Punteros 1er Indirecto:	1
Leer ultimos bloques datos (90KB):	45 (90KB/2KB)

Total Leído: 54 bloques

b) Fallos de Página:

Como es con algoritmo FIFO (lo mismo pasa con LRU y con CLOCK también), se reemplaza siempre la más vieja (por ser secuencial la carga) → los primeros 80 KB no generan fallos de página.

A medida que llegan los últimos 20 KB se producen fallos, siendo el total de 20KB/4KB = **5 Fallos de Página** (son la cantidad de páginas que les faltaría para entrar totalmente en memoria).

c) Bloques Escritos:

Se deben escribir las páginas bajadas a disco por los fallos de página=20KB.

Suponiendo que ya exista un INodo para manejar la memoria virtual de ese proceso.

→ Se lo debe leer (1 leída más), se saca de memoria el inodo /Starfleet/Utopia/P1.secretplan (LIFO), pero como no se lo modifico no hace falta grabarlo de nuevo (de más esta decir que no hace falta leerlo de nuevo pues ya tiene el bloque de puntero en memoria).

La grabación de las páginas utiliza todos los bloques directos y el 1er indirecto del inodo (a grabar > total directo). Supongo que cuando termina actualiza el bloque de punteros y el inodo en disco

→ Bloques Datos Directos:	5 (10KB)
Bloque Datos 1er. Ind.	5 (10KB)
Actualización:	2 (grabar el bloque indirectos y el inodo)

Total Escrito: 12 Bloques

-Entonces para CARGAR el archivo en memoria se precisan leer 54 bloques, escribir 12 y se tienen 5 fallos de página.

d) Lo que no queda muy claro es si el archivo después debe leerse el archivo de memoria, de ser así se provocará más: 50 bloques Leídos (2KB x 50 = 100 KB, por el FIFO se lee todo de nuevo)

25 Fallos de Página (100KB/4KB = 25, es como si tuviera que cargar todo por el FIFO)

42 bloques Escritos (los 80 KB restantes, que bajan a disco por FIFO, + actualización inodo y bloque punteros)

Siendo el total, de cargarlo y leerlo en memoria, de 104 bloques leídos, 54 escritos y 30 fallos de página.

3- ASEGURAR AUTENTICIDAD E INTEGRIDAD:

Lo que se quiere mandar lo llamaremos MSG.

Origen X

A=Hash (MSG)

B=Asim Privada_X (MSG, A)

Se manda B a Y.

Destino Y

Recibe B de X.

(MSG, A) = Asim Pública_Y (B)

C=Hash (MSG)

Si (A=C) entonces OK LA INTEGRIDAD

sino ERROR CON LA INTEGRIDAD

Se sabe que lo mando X porque es el único que puede tener su clave Privada y cada par Pública/Privada son únicas.

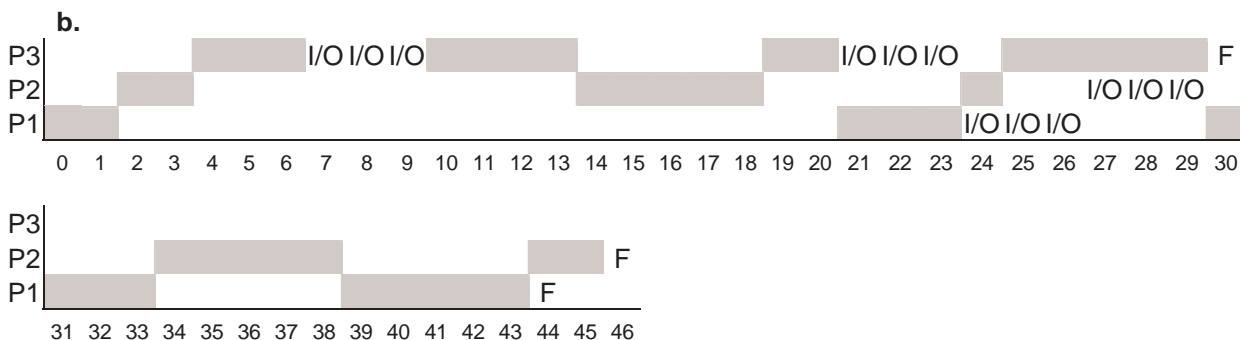
Ejercicio

Sea un sistema operativo que utiliza un planificador con el algoritmo RR de $q=4$ y que dispone de las funciones: leer, escribir y ejecutar, así como también las funciones atómicas: up y down. Conociendo que las colas de los semáforos y la única cola de entrada/salida son independientes y como están inicializados los semáforos, indique la traza de ejecución y el orden de terminación de los siguientes procesos:

Semáforos (valor inicial)	Funciones		
		CPU (al inicio)	I/O (al final)
A=0	Up/Down	2	0
B=0	Leer/Escribir	1	3
C=1	Ejecutar	3	0

Proceso 1	Proceso 2	Proceso 3
Down(A)	Down(B)	Down(C)
Up(C)	Ejecutar(Func 2)	Leer(Archivo A, y)
Leer(Archivo A, y)	Up(A)	Up(B)
Down(A)	Escribir(Archivo A, y)	Ejecutar(Func 2)
Ejecutar(Func2)	Ejecutar(Func 1)	Escribir (Archivo A , x)
Up(B)	Down(B)	Ejecutar (Func 1)
Down(C)	Up(C)	Up(A)

a. Finalizan: P3(30), P1(44), P2(46)



EJERCICIOS NO RESUELTOS

Ejercicio Bombillas

Una fabrica produce bombillas de 25, 40, 75 y 100 WATTS, cada una ocupa 1, 2, 3 y 4 casilleros respectivamente. El galpón de la fábrica posee 2048 casilleros, divididos en estantes de 64 casilleros, y existen dos archivos de entrada (desincronizados), uno para consumir y otro para producir.

La estructura de los archivos será:

NroProductor/NroConsumidor:Cantidad:tipo de bombilla;

“;” es el separador de los n registros existentes

Nro de Productor y Consumidor se deben corresponder en los pedidos y consumiciones.

Existe un capataz que administra los casilleros asignados a cada productor.

Se deberán organizar los pedidos y consumos de acuerdo a los siguientes criterios:

- Se tiene dividido el galpón en 32 estantes. La asignación a cada productor será por estante completo.
- El capataz asignará los casilleros necesarios para cada producción.
- Se cuenta con un depósito adicional de 1024 casilleros que sólo podrá utilizarse en el caso en que el área asignada al productor correspondiente esté completa.

RECORDAR: todas las entregas se realizarán en modo gráfico.

Ejercicio

- Dibuje el árbol de procesos y la salida que genera el siguiente programa:

```
main()
{
    int est,i;
    for(i=0;i<3;i++)
        if (fork()==0){
            // wait(&est);
            print("%d\n",i);
        }
        else wait(&est);
        print("%d\n",i);
}
```

¿Qué ocurriría en el caso de eliminar las barras de comentario (//) en la línea 6?

- Calcular el número **total** de bloques de 1kb que se necesitan para almacenar un archivo de 1Mb en un sistema Unix (los números de bloque utilizan 32 bits).
- Un programa debe leer una tabla de una base de datos, que tiene 20000.- (veinte mil) registros con la siguiente estructura toda a memoria (trabaja con paginación, y el tamaño de página es de 1024).
 - Código (Double)
 - Descripción (60 caracteres)
 - Dirección (35 caracteres)
 - Código postal (8 caracteres)

Se pide:

- Indique la cantidad de páginas necesarias para cargar todos los registros a memoria.
- Indique cual es el porcentaje de desperdicio (si lo hay), de la memoria.
- Que cambios se podrían realizar para disminuir el desperdicio a la mínima sin cambiar el tamaño de página, y recalculé los dos puntos anteriores. (Todas las respuestas deben estar justificadas):

Ejercicio

Dado un archivo grabado sobre un disco de sectores de 240 bytes cuyas características son:

dirección: sector 300

longitud: 30 sectores

bloque: 240 bytes

registro lógico: 80 bytes
 y cuyo contenido en un momento dado es:

A	B	C	D	E	F
---	---	---	---	---	---

sector 300

y en ese mismo momento un programa que lo accede tiene:

READ	A	B	C
programa	Buffer(bloque 0 en el buffer)		

Se pide:

- a. Describa las operaciones para que el programa lea B.
- b. Describa las operaciones para que el programa lea E.

Ejercicio

En un sistema de archivos con catálogos multinivel es frecuentemente muy inconveniente para el usuario referirse a sus archivos por medio de un nombre completamente calificado. En nuestro caso, con un sistema de dos niveles es más conveniente para el programador escribir ARCH que LPR/ARCH. ¿Qué información extra debe registrar el sistema para permitir esa forma de referencia?

En el sistema propuesto, cada archivo posee una Lista de Control de Acceso (LCA). Cada LCA contiene la lista de los usuarios autorizados para acceder a él. Consideremos una implementación en la cual se asigna a cada usuario una Lista de Control de Usuario (LCU). La LCU contiene la lista de todos los archivos a los cuales el usuario puede acceder. Ofrece la LCU ventajas sobre la LCA? ¿Cuál esquema Ud. recomendaría? ¿Por qué?

Ejercicio

Dado el siguiente proceso, conteste las preguntas. (considere que está escrito en pseudo-código).

```

¿Cuál es el objetivo del programa?
¿Puede darse el caso de que un proceso imprima más de 10 veces el mensaje en pantalla? por qué?
¿Dada su función, qué tipo de semáforo es "m"?
¿Cuántos procesos compiten por la CPU? cuál empieza primero?
¿Haber puesto el semáforo "m" me sirve para evitar los continuos context-switch?
int main( )
{
    int x, i, j;
    m = 1;
    for (i=1; i<=3; i++)
        if ( !fork( ) ) break;

    while(True)
    {
        x=mi_random(1, 10); //Genera un numero al azar entre 1 y 10.
        P(m);
        for (j=1; j<=x; j++)
            printf("Soy el proceso %d\n", getpid( ) );
        V(m);
    }
}
  
```

Ejercicio

1. En un sistema que administra la memoria con una segmentación paginada bajo demanda, con asignación estática de frames, y política de reemplazo local (se asignan 6 frames por proceso, asignando uno para el área de código, una para el área de stack –si no se utiliza el área de stack, o

alguna de las páginas de código, las mismas pueden usarse como adicionales para datos-, y el resto para el área de datos). El compilador divide cualquier unidad funcional del programa fuente en un segmento. El loader carga el primer segmento y la página principal del proceso en memoria, y la misma no puede ser descargada. El tamaño de la página es de 32 KBytes. El tipo de dato float ocupa 8 Bytes. El tiempo en realizar un fallo de página sin guardar la víctima es de 20 ms., en cambio si hay que guardar la víctima el tiempo se duplica. El algoritmo para la elección de víctima es un método compuesto por una primera selección segunda oportunidad mejorada, para resolver los conflictos usa LRU, y en caso de que siga habiendo conflictos, usa FIFO.

2.- En el sistema anteriormente descrito se intenta ejecutar el siguiente proceso (debajo se da el programa que lo genera). Se pide que analice la codificación del mismo, e indique la cantidad de fallos de página que se producen entre páginas de datos y páginas de código. También debe calcular el tiempo total empleado por el proceso para cumplir su objetivo (Aclaración: El programa expuesto es solo una parte de la solución total, anterior a éste se ejecuta un proceso que carga la matriz, y posterior a la ejecución del mismo se genera un listado con los resultados obtenidos por el proceso que estamos analizando).

```
#include <stdio.h>
#define CantCol 4096
#define CantFil 20

float CalculaPromedioUnaFila(int);
void CalculaPromedio();

float Matriz[CantFil][CantCol];

int main()
{
    CalculaPromedio();
    return 0;
}

void CalculaPromedio()
{
    int i;
    for (i=0; i<CantFil; i++)
        for (j=0; j<CantCol-1; j++)
            Matriz[i][4095]+=Matriz[i][j];
}
```

En caso de que Uds. crea que existe alguna forma de mejorar el rendimiento del proceso (se puede cambiar la codificación, pero no las condiciones del S.O.), exprese la forma de hacerlo (el código del programa modificado), e indique conceptualmente (no hace falta que haga los cálculos, pero si los hace es mejor), porque mejorará el rendimiento.

Ejercicio

1.- La biblioteca de Universidad necesita reorganizarse de manera tal de optimizar espacio. A su vez, desea que la búsqueda de los libros esté clasificada por carrera, tema, subtema, etc, pudiendo un libro tener más de una clasificación. Los bibliotecarios desean caminar lo menos posible. Los alumnos pueden tener una atención aleatoria, y pedir más de un libro a la vez. Indicar en qué puede ayudar la teoría en sistemas de archivos a los bibliotecarios para optimizar TODO y que sea JUSTO.

2.- Retomemos el caso de la universidad, pero esta vez, en lugar de organizar la biblioteca, organizaremos el departamento de Alumnos. Supongamos que se decide comprar una cierta cantidad de computadoras (que pueden ser PC o workstation baratas) para que los alumnos las usen como autoconsulta, para saber notas, conocer sus estados de cuenta, comunicarse con sus docentes o enviar mensajes a la Tesorería. ¿Qué protección brindaría Ud. a este sistema? Presente la solución en forma ordenada, no en prosa.

Ayuda: Tenga en cuenta los siguientes puntos (son indicativos, no excluyentes, por favor, no se sienta limitado por las indicaciones de la cátedra):

- sistema operativo de la implementación (¿qué materia está cursando?)
- file system de la implementación
- validación de alumnos
- validación de personal del departamento de alumnos
- validación de docentes
- prioridad de los grupos que acceden al sistema
- comunicación docentes-alumnos y alumnos-Tesorería
- validación de acceso a cuentas de alumnos en Tesorería (esto es dinero)

- l) seguridad física (alguien puede querer llevarse las computadoras)
- j) virus

Ejercicio

Se dispone de un entorno de trabajo con las siguientes características:

- Implementa paginación por demanda con direcciones lógicas de 6 bits (3 destinados al nro. de pag).
- Total de memoria física: 2 frames
- Algoritmo de selección de página víctima: FIFO
- Estado actual del sistema:

Memoria física

HBCXI@MZGK1=VM_i5

Archivo de paginación (el archivo contiene las páginas d 0 a 7 en ese orden)

HBCXI@MZWZAM/*CT+D2H\B9S3RXFO#YR8DVLEP(\$GK1=VM_i5Q:6NZ>K0TJ4?4)EL

Tabla de páginas

Pag	Frame	Presente?
0	0	1
1	0	0
2	0	0
3	0	0
4	0	0
5	1	1
6	0	0
7	0	0

Se sabe que para componer una clave trabajan mancomunadamente 6 procesos (P0..P5) sobre un segmento de memoria compartida definido por los datos de manejo de memoria provistos. Cada uno de ellos hace una referencia final a la memoria compartida de la cual toma un carácter componente de la clave; de esta forma la misma resultará de la concatenación de estos caracteres en el orden entregado.

En este instante los 6 procesos (P0..P5) están en la cola de listos en ese orden. Acerca de las características de la ejecución de cada uno de ellos se sabe:

- P0 tendrá una ráfaga de CPU de 6 ut. y referenciará finalmente la dirección 6.
- P1 tendrá una ráfaga de CPU de 5 ut. y referenciará finalmente la dirección 10.
- P2 tendrá una ráfaga de CPU de 4 ut. y referenciará finalmente la dirección 13.
- P3 tendrá una ráfaga de CPU de 3 ut. y referenciará finalmente la dirección 34.
- P4 tendrá una ráfaga de CPU de 2 ut. y referenciará finalmente la dirección 27.
- P5 tendrá una ráfaga de CPU de 1 ut. y referenciará finalmente la dirección 28.

Sabiendo que el algoritmo de scheduling de corto plazo es RR con $q=1$ ut, se pregunta

1. ¿Cuál es la clave generada?
2. ¿Cuántos fallos de página ocurren?
3. Si ahora el algoritmo de scheduling de corto plazo es FIFO, ¿Cuál será la clave?

Ejercicio

Dados los siguientes programas conteste:

- f. ¿Qué caracteres escribe el Proceso A en pantalla?
- g. ¿Qué caracteres escribe el Proceso B en pantalla?
- h. ¿Qué caracteres escribe el Proceso C en pantalla?
- i. ¿Existe algún problema de concurrencia entre los procesos?
- j. ¿Qué proceso debería borrar los recursos semáforos y memoria compartida y bajo qué condiciones?
- k. Indique cuál es la secuencia de ejecución de los procesos
- l. ¿Qué parámetros se deberían pasar a cada proceso (en caso de ser necesarios)?

Proceso A	Proceso B	Proceso C
<pre>#include <operIPC.h> #include <stdio.h> main(int argc, char **argv) { int semid , shmid ; char * dir ; char letra = 'a' ; semid = atoi(argv[1]); shmid = shmget(0xF,0,0); dir = (char *) shmat(shmid,0,0); P(semid,0); *dir = letra; V(semid,3); while(letra <= 'z') { P(semid,0); letra = *dir ; letra ++ ; *dir = letra; V(semid,3); } }</pre>	<pre>#include <operIPC.h> #include <stdio.h> main(int argc, char **argv) { int semid , shmid; char * dir ; char letra = ' ' ; semid = atoi(argv[1]); shmid = shmget(0xF,0,0); dir = (char *) shmat(shmid,0,0); while(letra <= 'z') { P(semid,1); P(semid,3); letra = *dir ; V(semid,2); V(semid,0); printf("%c\n",letra); } }</pre>	<pre>#include <operIPC.h> #include <stdio.h> main(int argc, char **argv) { int semid , shmid; char * dir ; char letra = ' ' ; semid = atoi(argv[1]); shmid = shmget(0xF,0,0); dir = (char *) shmat(shmid,0,0); while(letra <= 'z') { P(semid,2); P(semid,3); letra = *dir ; V(semid,0); V(semid,1); printf("%c\n",letra); } }</pre>

Inicialización de los semáforos (Sem0 = 1 , Sem1 = 1 , Sem2 = 0 , Sem 3 = 0)

La cabecera operIPC.h define las operaciones wait como P y signal como V sobre semáforos.

Ejercicio

Un aeropuerto internacional cuenta con 3 pistas de aterrizaje y 10 hangares para guardar aviones. El trayecto que une los hangares con las pistas solo permite el paso de un avión por vez. Un avión que desea despegar debe verificar que haya una pista disponible, salir del hangar hacia dicha pista y luego despegar. Un avión que desea aterrizar debe verificar que haya una pista libre y aterrizar, luego verificar la disponibilidad de un hangar y desplazarse hacia el. Si un avión aterriza y no hay hangares disponibles debe quedarse ocupando la pista mientras espera que un hangar se libere.

- Se pide que determine los semáforos con sus valores iniciales y las primitivas necesarias para sincronizar el aeropuerto.
- Si las 3 pistas fueran ocupadas por aviones en espera de hangares y todos los hangares estuviesen ocupados, se produciría un deadlock ? Justifique su respuesta.
Resultaría conveniente arbitrar los medios necesarios para evitar que esto ocurra.

Tip: Los procesos a utilizar son:

Despegar()
Aterrizar()
Salir_del_hangar()
Entrar_al_hangar()

Ejercicio

Ejercicio