1. Cuestiones sobre procesos, y asignación de CPU:

a) ¿Es necesario que lo último que haga todo proceso antes de finalizar sea una
llamada al sistema para finalizar? ¿Sigue siendo esto cierto en sistemas monoprogramados?
Si es necesario ya que sino no se mata al proceso.
Para sistemas monoprogramados también es necesario.

b) Cuando un proceso se bloquea, ¿deberá encargarse él directamente de cambiar el valor de su estado en el descriptor de proceso o PCB?
No. El proceso llama al SO y este lo bloquea y cambia su estado en el PCB gracias al dispatcher

c) ¿Qué debería hacer el planificador a corto plazo cuando es invocado, pero no hay ningún proceso en la cola de ejecutables?
Se introduce lo que se conoce como proceso nulo. Este proceso siempre esta listo para ejecutarse y tiene la prioridad mas baja.

d) ¿Qué algoritmos de planificación quedan descartados para ser utilizados en sistemas de tiempo compartido?
Los de planificación no apropiativa ya que en estos, al proceso actual no se le puede retirar la CPU, lo que quiere decir que no podrían coordinarse entre ellos. También descartaríamos los algoritmos por prioridades porque ocuparía casi toda la CPU y los demás sistemas no podrían utilizar la CPU.

2. La representación gráfica del cociente[(tiempo_en_cola_ejecutables + tiempo_de_CPU) / tiempo_de_CPU] frente a tiempo_de_CPU suele mostrar valores muy altos para ráfagas muy cortas en casi todos los algoritmos de asignación de CPU. ¿Por qué?

La fórmula representa la penalización de un proceso en función de su tiempo de respuesta y su tiempo de ráfaga. Cuanto menor es el tiempo de ráfaga, mayor es la penalización del proceso. Además, los procesos cortos suelen pasar más tiempo en la cola de ejecutables que con la CPU asignada, por ello presentan esos altos valores de penalización.

3. Para cada una de las llamadas al sistema siguientes, especificar y explicar si su procesamiento por el sistema operativo requiere la invocación del planificador a corto plazo:

a) **Crear un proceso.** De eso se encarga el planificador a largo plazo o un proceso interactivo. Sin embargo, si la cola de procesos Listos estuviera vacía y llegase este proceso, el planificador a corto plazo sería el encargado de ponerlo en ejecución. Si la prioridad del nuevo proceso fuera mayor que la del proceso en ejecución y se tratase de una política apropiativa, también intervendría.

b) **Abortar un proceso, es decir, terminarlo forzosamente.** El aborto de un proceso implica que se le quita la CPU, por lo que el planificador a corto plazo entraría en juego.

c) **Suspender o bloquear un proceso.** El planificador a corto plazo quita la CPU al proceso en ejecución. Luego, en el caso de suspensión, el planificador a medio plazo pondría al proceso en ejecución en la cola adecuada. En el caso de bloqueo, el planificador a corto plazo lo pondría en la cola de Listos. En ambos casos, después de lo anterior, el planificador a corto plazo cogería el siguiente proceso de la cola de Listos y lo pondría en ejecución.

d) **Reanudar un proceso (inversa del caso anterior).** Si la política es apropiativa y en caso de que la prioridad del proceso sea mayor del que está en ejecución, el planificador a corto plazo podría bloquear el proceso en ejecución para ejecutar el nuevo proceso. Si no es apropiativa, el planificador a corto plazo no actúa.

e) **Modificar la prioridad de un proceso.** Igual que en el caso anterior.

4. Sea un sistema multiprogramado que utiliza el algoritmo Por Turnos (*Round-Robin*). Sea S el tiempo que tarda el despachador en cada cambio de contexto. ¿Cuál debe ser el valor de quantum Q para que el porcentaje de uso de la CPU por los procesos de usuario sea del 80%?

S = Dispatcher
Q = Quantum
CPU = 80% procesos usuario → $0,8T = Q \rightarrow T = Q / 0,8$

Q?

$T = Q + S$

Sustituir T en la formula anterior:

$Q / 0,8 = Q + S$
$Q / (Q + S) = 0,8$
$Q = 0,8 (Q + S)$
$Q = 0,8Q + 0,8S$
$0,2Q = 0,8S$
$Q = 4S$

7. ¿Puede el procesador manejar una interrupción mientras está ejecutando un proceso si la política de planificación que utilizamos es no apropiativa?

Si que sería posible, ya que el tratamiento de interrupciones es independiente de la política de planificación utilizada. Esto permitiría el progreso finito.

9. En el algoritmo de planificación FCFS, la penalización ((t + tº de espera) / t), ¿es creciente, decreciente o constante respecto a t (tiempo de servicio de CPU requerido por un proceso)? Justifique su respuesta.

Penalización = (t + tº espera) / t

Es decreciente, ya que a medida que aumenta el tiempo de servicio de CPU requerido por un proceso, el cociente disminuye y por tanto la penalización es menor. Por otro lado, a medida que disminuye el tiempo de servicio, la penalización aumenta.

Por eso FCFS penaliza a procesos cortos y beneficia a procesos largos.

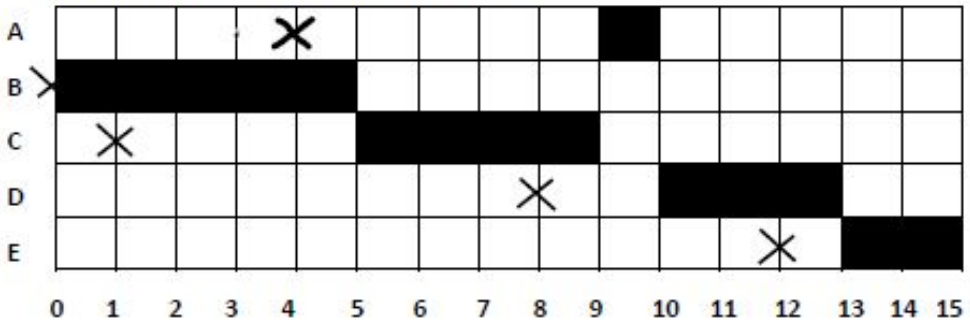
Ejemplo:
tº espera = x
t=1 Penalización = x+1
t=2 Penalización = x+2/2
t=3 Penalización = x+3/3
x+1 < x+2/2 < x+3/3 < …

10. En la tabla siguiente se describen cinco procesos:

Proceso	Tiempo de creación	Tiempo de CPU
A	4	1
B	0	5
C	1	4
D	8	3
E	12	2

Si suponemos que tenemos un algoritmo de planificación que utiliza una política FIFO (primero en llegar, primero en ser servido), calcula:

1º Hacemos el diagrama de ocupación de la CPU:



	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Cola		C	C	C	AC	A	A	A	AD	D			E			

a) **Tiempo medio de respuesta**

ProcesoA = 10 - 4
ProcesoB = 5 – 0
ProcesoC = 9 – 1
ProcesoD = 13 – 8
ProcesoE = 15 – 12

Tº medio de respuesta = 27/5 = 5.4

b) **Tiempo medio de espera**

ProcesoA = 9 - 4
ProcesoB = 0 – 0
ProcesoC = 5 – 1
ProcesoD = 10 – 8
ProcesoE = 13 – 12

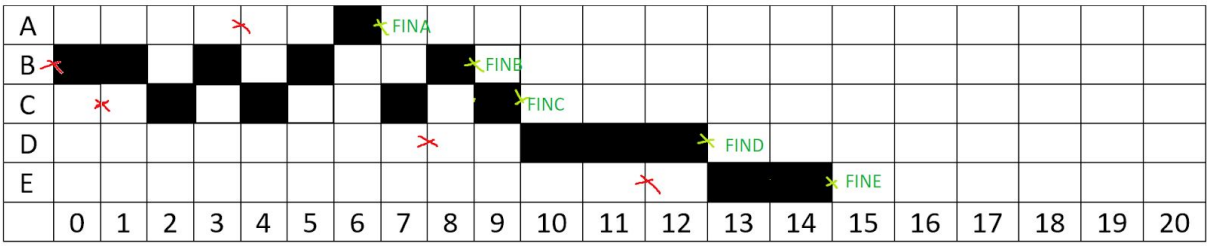
Tº medio de espera = 12/5 = 2.4

11. Utilizando los valores de la tabla del problema anterior, calcula los tiempos medios de espera y respuesta para los siguientes algoritmos:

Especificar que cuando un procesos termina una rafaga si continua teniendo rafagas por realizar, será ese procesos el que primero se ponga a la cola.

Proceso	Tiempo de creación	Tiempo de CPU
A	4	1
B	0	5
C	1	4
D	8	3
E	12	2

a) Por Turnos con quantum q=1



q = 1

Cola	B	BC	CB	BC	CBA	BAC	ACB	CB	BCD	CD	D	D	DE	E	E		

Tiempo espera A = 2
Tiempo espera B = 0
Tiempo espera C = 1
Tiempo espera D = 2
Tiempo espera E = 1
Tiempo medio = 6/5 = 1,2

Tiempo respuesta A = 7- 4 = 3
Tiempo respuesta B = 2 -0 = 2
Tiempo respuesta C = 3 -1 = 2
Tiempo respuesta D = 13 - 8 = 5
Tiempo respuesta E = 15 -12 = 3
Tiempo medio = 3

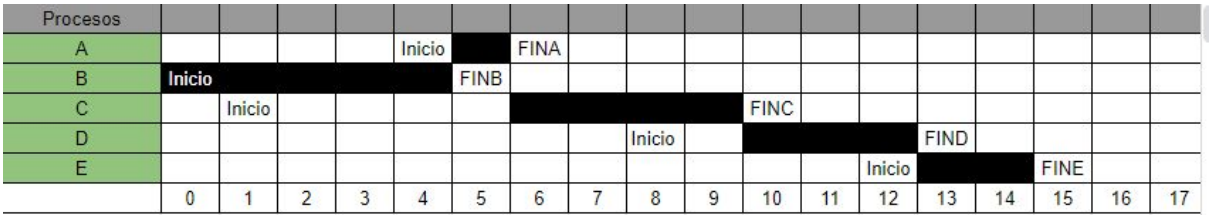
b) Por Turnos con quantum q=4

Procesos																
A					inicio						FIN					
B	inicio									FIN						
C		inicio							FIN							
D									inicio					FIN		
E													inicio			FIN
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Cola de procesos	B	BC	BC	BC	CBA	CBA	CBA	CBA	BAD	AD	D	D	DE	E	E	

Tiempo espera A = 5
Tiempo espera B = 0
Tiempo espera C = 3
Tiempo espera D = 2
Tiempo espera E = 1
Tiempo medio: 11/5

Tiempo respuesta A: 6
Tiempo respuesta B: 4
Tiempo respuesta C: 7
Tiempo respuesta D: 5
Tiempo respuesta E: 3
Tiempo respuesta medio: 25/5 = 5

c) El más corto primero (SJF). Suponga que se estima una ráfaga igual a la real.



Tiempo espera A = 1
Tiempo espera B = 0
Tiempo espera C = 5
Tiempo espera D = 2
Tiempo espera E = 1
Tiempo medio: 9/5

Tiempo respuesta A: 2
Tiempo respuesta B: 5
Tiempo respuesta C: 9
Tiempo respuesta D: 5
Tiempo respuesta E: 3
Tiempo respuesta medio: 24/5

13. Utilizando la tabla del ejercicio anterior, dibuja el diagrama de ocupación de CPU para el caso de un sistema que utiliza un algoritmo de colas múltiples con realimentación con las siguientes colas:

Proceso	Tº Llegada	Tº uso
A	0	3
B	1	1
C	3	12
D	9	5
E	12	5

Cola	Prioridad	Quantum
1	1	1
2	2	2
3	3	4

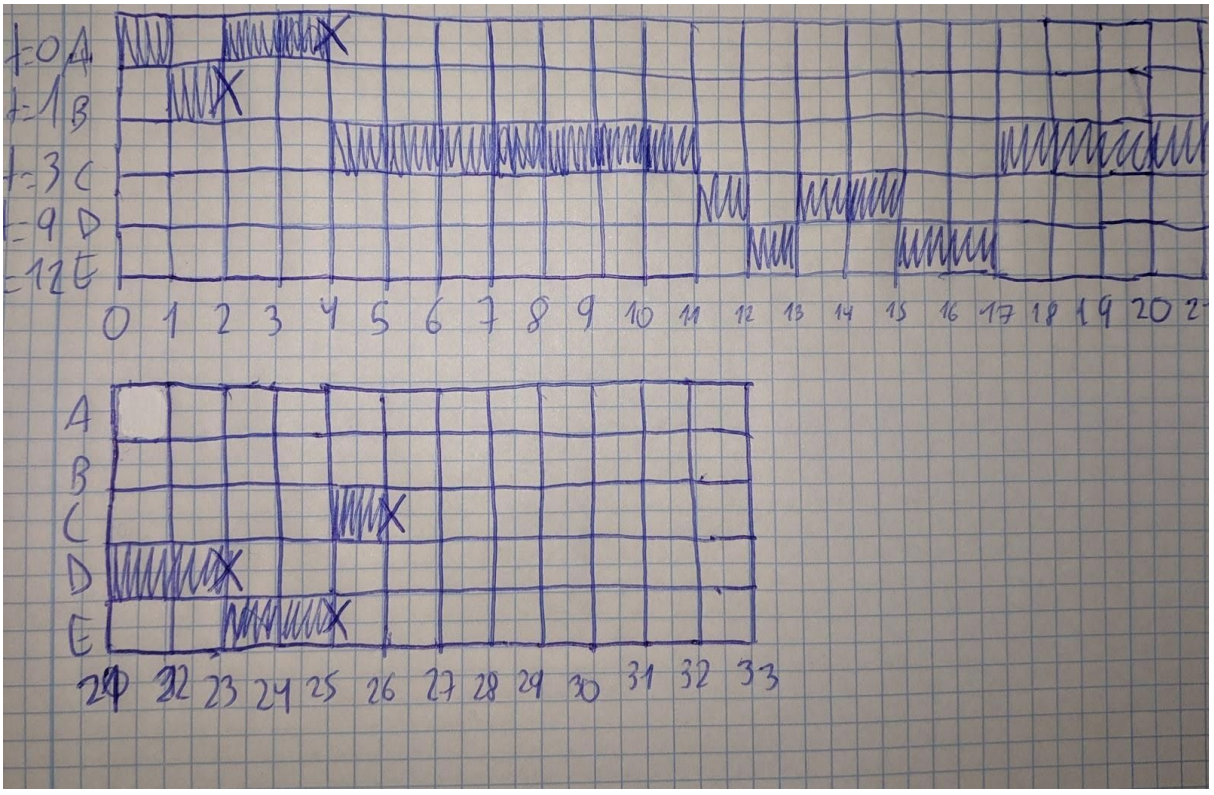
y suponiendo que:

(a) Todos los procesos inicialmente entran en la cola de mayor prioridad (menor valor numérico). Cada cola se gestiona mediante la política Por Turnos.
(b) la política de planificación entre colas es por prioridades no apropiativo.
(c) un proceso en la cola i pasa a la cola i+1 si consume un quantum completo sin bloquearse.
(d) cuando un proceso llega a la cola de menor prioridad, permanece en ella hasta que finalice.

Solución:

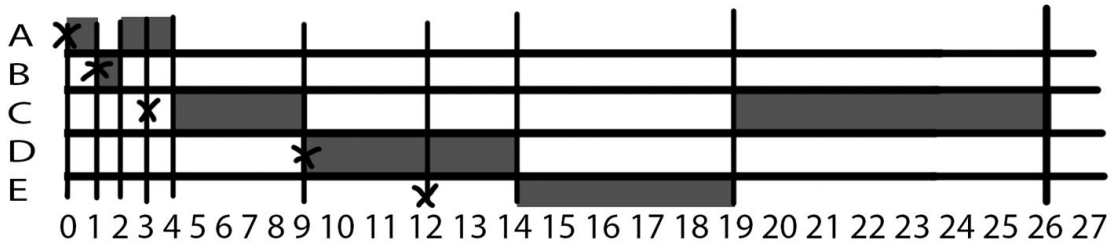
Cola	t=0	t=1	t=2	t=3	t=4	t=5	t=7	t=9	t=11	t=12	t=13
1	A	B	x(B)	C	C			D	D	E	
2		A(2t)			x(A)	C(11)				D(4)	D(4)E(4)
3							C(9)	C(7)	C(5)	C(5)	C(5)

Cola	t=15	t=17	t=21	t=23	t=25	t=26	
1							
2	E(4)						
3	C(5)D(2)	C(5)D(2)E(2)	D(2) E(2) C(1)	x(D) E(2) C(1)	x(E) C(1)	x(C)	



12. Calcula el tiempo de espera medio para los procesos de la tabla utilizando el algoritmo: el primero más corto apropiativo (o primero el de tiempo restante menor, SRTF).

Proceso	Tiempo de creación	Tiempo de CPU
A	0	3
B	1	1
C	3	12
D	9	5
E	12	5



Tº espera medio = Tº espera total/no procesos = 14/5 = 2,8

16. ¿El planificador CFS de Linux favorece a los procesos limitados por E/S (cortos) frente a los procesos limitados por CPU (largos)? Explique cómo lo hace.

Sí, el planificador de Linux favorece a los procesos limitados por E/S.

El planificador CFS *(Completely Fair Scheduler)* guarda la cantidad de tiempo de CPU que se le ha asignado a una determinada tarea *(denominado tiempo de ejecución virtual o virtual runtime en inglés)*. A menor tiempo de ejecución virtual *(es decir, menor tiempo de CPU ha usado este proceso)* más necesidad tiene de usar el procesador. Además, el planificador CFS incluye el concepto de *sleeper fairness* para asegurarse de que procesos que actualmente no son ejecutables *(como por ejemplo, aquellos procesos bloqueados esperando E/S)* reciben también una cantidad justa de tiempo.