Tema 4. Gestió de processos

Què aprendràs

- Diferenciar un programa d'un procés, i un procés d'un fil.
- Explicar què guarda el sistema operatiu de cada procés i per a què li serveix.
- Dibuixar el diagrama d'estats d'un procés i explicar cada canvi.
- Calcular els temps de resposta, de servei i d'espera d'un conjunt de processos.
- Resoldre el gràfic d'execució (diagrama de Gantt) amb FIFO, Round Robin, SJF i SRT, amb la cua.
- Veure i manejar els processos del teu ordinador des del terminal.

Currículum del mòdul 0222 (RD 1691/2007): RA1 c) · RA4 c).

La teva xuleta per a tot el tema

Aquestes set columnes surten a tots els exercicis del tema. Copia-les cada vegada.

Columna	Què és	Com es calcula
Arribada	Cicle en què arriba el procés	te la donen
CPU	Cicles de CPU que necessita	te'ls donen
Inici	Cicle en què s'executa per primera vegada	la treus de la graella
Fi	Últim cicle en què s'executa: quan acaba	la treus de la graella
Resposta	Cicles des que arriba fins que entra per primer cop a la CPU	**Resposta = Inici − Arribada**
Servei	Cicles des que arriba fins que acaba	**Servei = Fi − Arribada + 1**
Espera	Cicles que passa a la cua de preparats, sense CPU	**Espera = Servei − CPU**

Fixa't en l'ordre: **primer omples la graella, d'allà treus l'Inici i la Fi, i amb això ja surten la resposta, el servei i l'espera.** Mai al revés.

Per què el +1 del servei? Perquè compten el cicle d'arribada i el de fi. Si un procés arriba al cicle 4 i acaba al 5, ha estat al sistema els cicles 4 i 5: són **2**, i 5 − 4 + 1 = 2.

Si hi ha entrada/sortida, a l'espera també li restes els cicles d'E/S: **Espera = Servei − CPU − E/S**. Ho veuràs a l'apartat 10.1.

---

1. Un programa no és un procés

Un **programa** és un fitxer guardat al disc. Està quiet. No fa res.

Un **procés** és aquest programa quan l'obres i es posa en marxa: es carrega a la RAM i la CPU comença a executar les seves instruccions una darrere l'altra.

El programa és la recepta. El procés és cuinar.

Dues coses que sorprenen i convé tenir clares:

- **Un programa pot donar diversos processos.** Obre dues finestres del Bloc de notes: és un sol programa, però el sistema té dos processos independents.
- **Un procés pot estirar de diversos programes.** Un navegador pot cridar un lector de PDF o un reproductor de vídeo mentre treballa.

Practica 4.1

a) Tens obertes tres finestres del Chrome. Quants programes hi ha? Quants processos?

b) Explica amb les teves paraules la diferència entre programa i procés.

2. Procés i fil

Dins d'un procés hi pot haver diversos **fils** (o *threads*). Un fil és una línia d'execució dins del procés.

	Procés	Fil
Memòria	La seva pròpia, aïllada	Comparteix la del procés
Crear-lo	Costa més	És ràpid i lleuger
Si falla	No afecta els altres processos	Pot tombar tot el procés
Exemple	Obrir el navegador	Cada pestanya carregant alhora

Els fils d'un mateix procés **comparteixen** el codi, les dades, el monticle i els fitxers oberts. Cada fil té **només per a ell** la seva pròpia pila i el seu comptador de programa.

> **Per què importa.** Que els processos tinguin la memòria separada és el que impedeix que un programa llegeixi o espatlli la memòria d'un altre. És la base de la seguretat i de l'estabilitat del sistema. Si un programa es penja, els altres segueixen funcionant.

---

3. La memòria d'un procés

Quan un procés es carrega a la RAM, el seu espai es reparteix en quatre zones:

Zona	Què guarda
Codi	Les instruccions del programa. Només lectura: no es poden canviar mentre s'executa.
Dades	Les variables globals, les que existeixen durant tot el programa.
Pila *(stack)*	Les variables locals i les crides a funcions. Creix i decreix sola.
Monticle *(heap)*	La memòria que el programa demana sobre la marxa. La gestiona el programador.

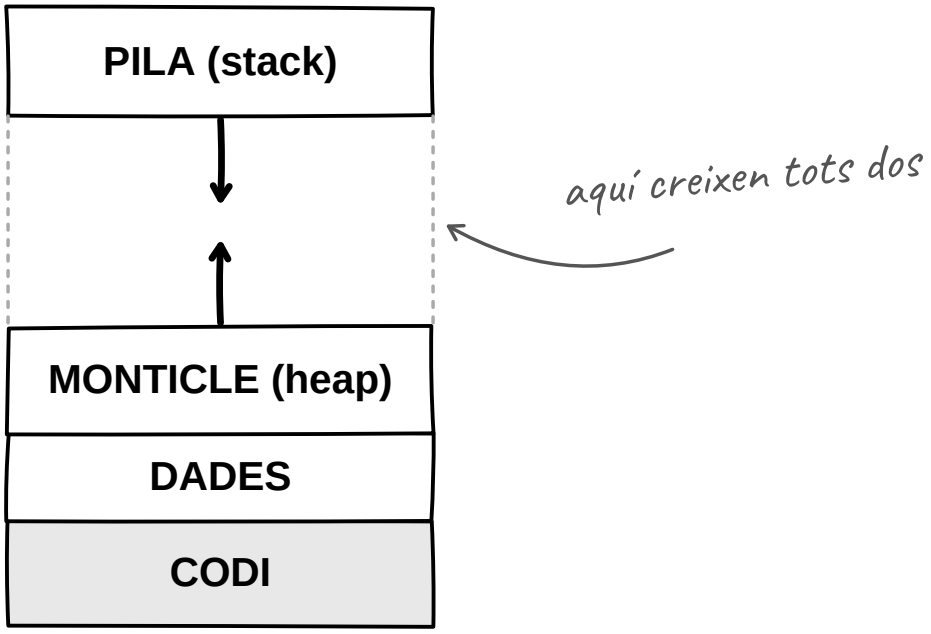

FIG 4.1 Les quatre zones de memòria d'un procés.

Practica 4.2

Digues en quina zona va cada cosa.

Element	Zona
Les instruccions del programa	
Una variable que es crea dins d'una funció	
Una variable global	
Memòria que el programa reserva mentre corre	

4. El PCB: la fitxa de cada procés

Quan el sistema operatiu crea un procés, li obre una fitxa. Es diu **PCB**, *Process Control Block*. Allà guarda tot el que necessita saber-ne:

- **PID**: el número que l'identifica. No n'hi ha dos d'iguals.
- **Estat**: en quin dels cinc estats està ara mateix.
- **Comptador de programa**: l'adreça de la instrucció següent que li toca.
- **Registres de la CPU**: els valors que tenia la CPU quan se li va llevar el torn.
- **Informació de planificació**: la seva prioritat i el temps de CPU que porta gastat.
- **Estadístiques**: quan va començar i quant porta.

El PCB és el que fa possible que un procés s'aturi i després segueixi **exactament on era**.

---

5. El canvi de context

La CPU només pot atendre un procés alhora per cada nucli. Quan li toca el torn a un altre, el sistema ha de fer dues coses:

1. **Guardar** al PCB del procés que surt tot el que tenia: registres i comptador de programa.
2. **Carregar** des del PCB del procés que entra tot el seu.

Això és un **canvi de context** (*context switch*).

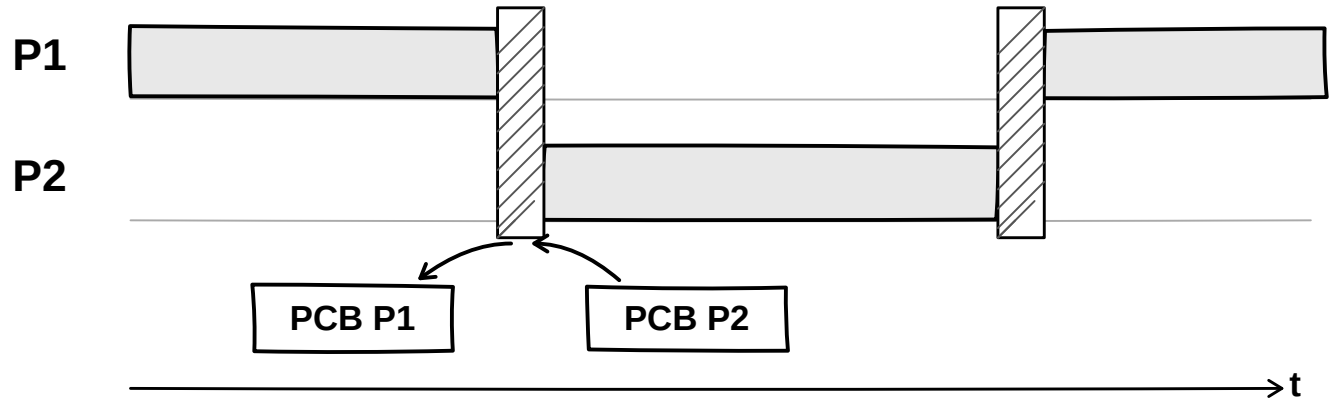

FIG 4.2 Canvi de context entre dos processos. La franja ratllada és temps perdut.

El que importa. El canvi de context **costa temps** i durant aquesta estona la CPU no fa feina útil. Per això no interessa canviar de procés cada dos per tres. Recorda-ho quan arribem al Round Robin.

---

6. Els estats d'un procés

Un procés no s'està executant sempre. Va passant per cinc estats:

Estat	Què significa
Nou *(new)*	S'acaba de crear. Encara no està preparat.
Preparat *(ready)*	A punt i esperant torn de CPU.
En execució *(running)*	Té la CPU. **És l'únic estat on el procés avança.**
Bloquejat *(blocked)*	Està esperant alguna cosa: el disc, el teclat, la xarxa. No vol CPU.
Acabat *(terminated)*	Ha acabat. El sistema allibera els seus recursos.

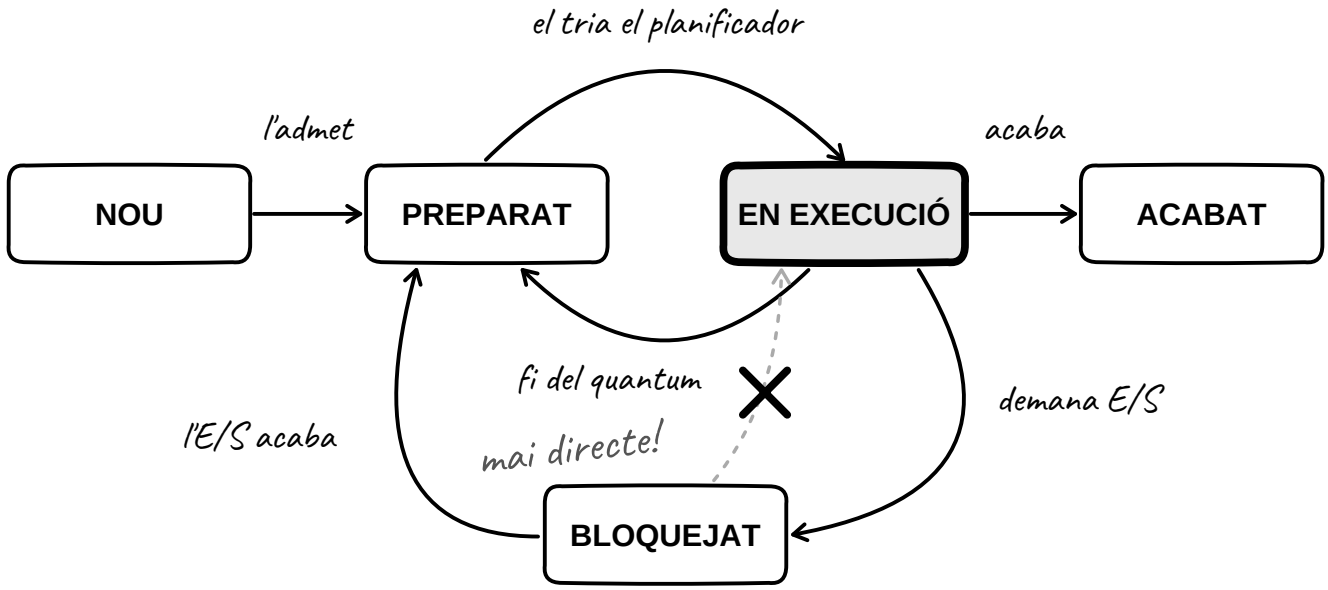

FIG 4.3 Diagrama d'estats d'un procés.

Els canvis entre estats no són lliures. Aquests són els únics que existeixen:

De	A	Per què
Nou	Preparat	El sistema l'admet
Preparat	En execució	El planificador el tria
En execució	Preparat	Se li ha acabat el quantum
En execució	Bloquejat	Demana una operació d'entrada/sortida
Bloquejat	Preparat	L'operació ha acabat
En execució	Acabat	El procés acaba

Atenció al que més es falla. De **Bloquejat** no es passa mai directament a **En execució**. Quan acaba allò que estava esperant, el procés torna a **Preparat** i fa cua una altra vegada, com tothom. Aquesta fletxa mal posada és l'error clàssic de l'examen.

Guiat 4.3

Completa quin estat correspon a cada situació.

Situació	Estat
El procés està esperant que l'usuari premi una tecla	
El procés està fent servir la CPU en aquest moment	En execució
Ha esgotat el seu quantum i torna a la cua	
Acaba de crear-se i encara no és a la cua	
Ha acabat la seva última instrucció	

Practica 4.4

Dibuixa de memòria el diagrama d'estats complet, amb les sis fletxes i les seves causes.

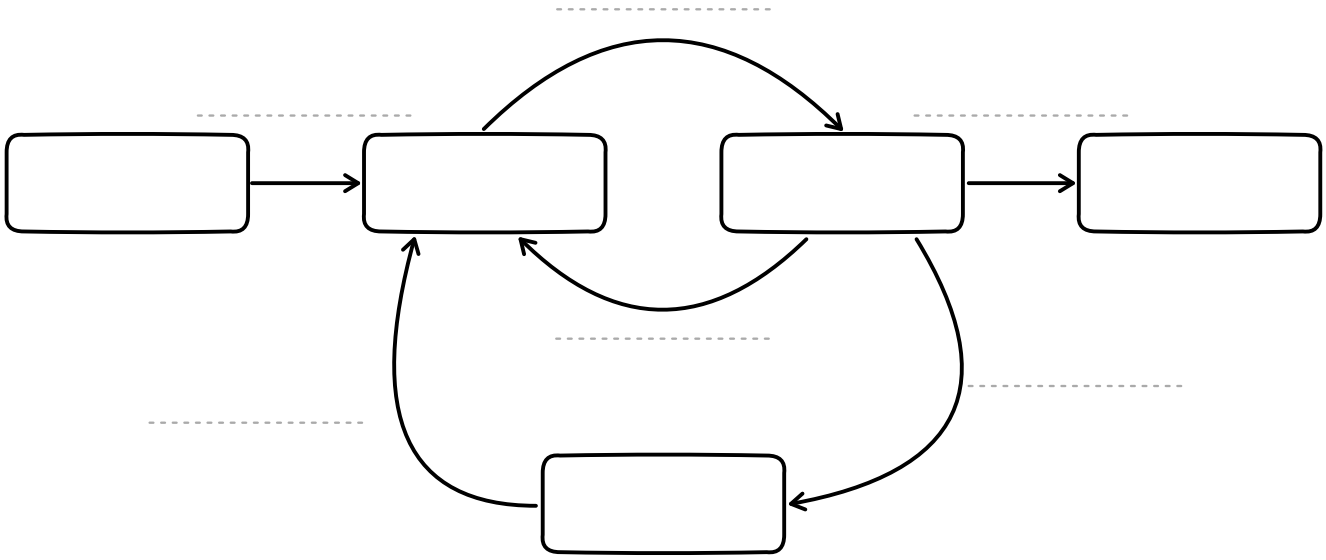

FIG 4.3m Completa el diagrama: escriu els estats i el motiu de cada fletxa.

7. El planificador

Quan hi ha molts processos a la **cua de preparats** (*ready queue*), algú ha de decidir quin entra a la CPU. Aquest algú és el **planificador** (*scheduler*), i decideix seguint un **algorisme de planificació**.

Un cop triat, qui fa el canvi de context i li dona la CPU al procés és una altra peça petita del nucli, el **distribuïdor** (*dispatcher*). El planificador decideix; el distribuïdor executa.

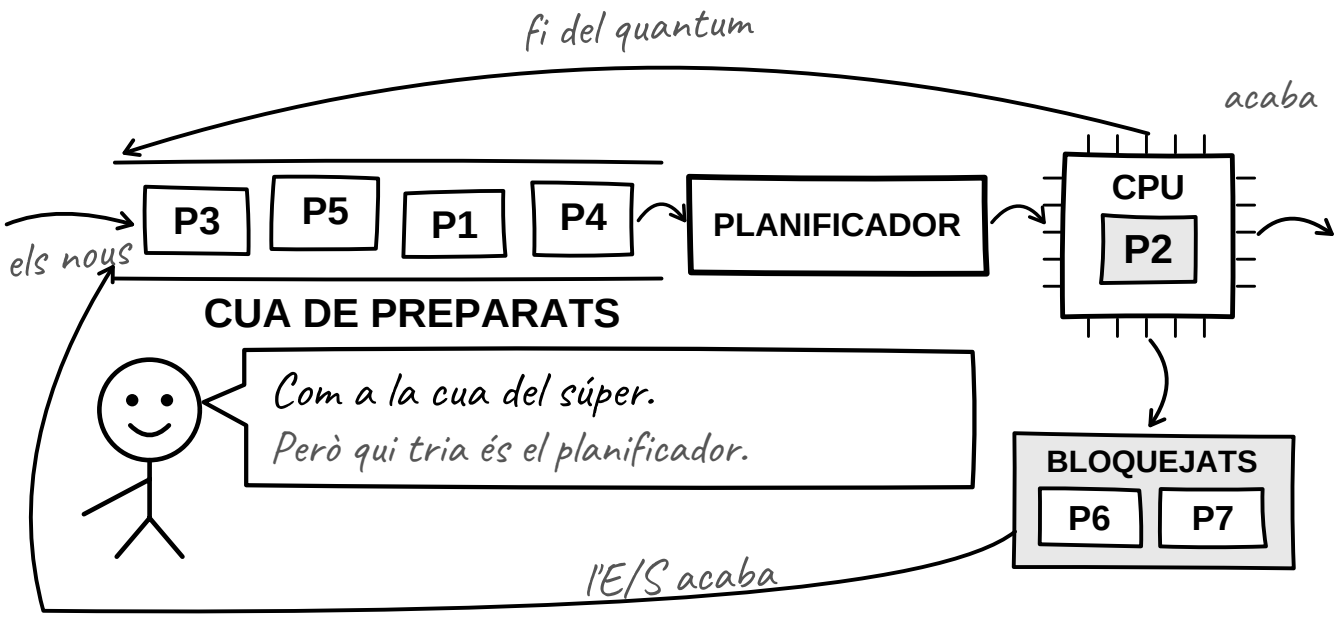

FIG 4.4 La cua de preparats, el planificador i la CPU.

Hi ha una paraula que necessites abans de seguir:

- **No expulsiu** (*non-preemptive*): un cop un procés agafa la CPU, la té fins que acaba o fins que es bloqueja. Ningú no l'hi lleva.
- **Expulsiu** (*preemptive*): el sistema li pot llevar la CPU encara que no hagi acabat.

8. Com es mesura si un algorisme és bo

Aquí és on entra la xuleta del principi. Anem amb un exemple complet perquè vegis el procediment sencer.

Aquests cinc processos els farem servir amb els quatre algorismes, per poder comparar-los:

Procés	Arribada	CPU
A	1	3
B	4	2
C	5	3
D	7	4
E	8	2

El temps es compta en **cicles de rellotge**, i el primer és el cicle **1**. Un procés que arriba al cicle 4 ja es pot executar al cicle 4.

---

9. FIFO — el primer que arriba, el primer que s'atén

FIFO (*First In, First Out*) és la cua del supermercat. S'atén per ordre d'arribada, i **el que entra no deixa anar la CPU fins que acaba**. És **no expulsiu**.

Exemple resolt 1 — FIFO

Pas 1. La graella, el diagrama de Gantt (*Gantt chart*). A classe en diem el **gràfic d'execució**. Hi ha una fila per a cada procés i una columna per a cada cicle:

- **X**: aquell cicle, el procés és a la CPU.
- **F**: l'últim cicle de CPU. El procés acaba.
- **Casella buida**: el procés és a la cua, o encara no ha arribat.

A sota hi ha la **cua de preparats** de cada cicle. A dalt, el primer que entrarà.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	X	X	F											
B				X	F									
C						X	X	F						
D									X	X	X	F		
E													X	F
Cua					C		D	D	E	E	E	E		
								E						

Pas 2. Traiem l'Inici i la Fi de la graella i calculem.

Procés	Arribada	CPU	Inici	Fi	Resposta	Servei	Espera
A	1	3	1	3	0	3	0
B	4	2	4	5	0	2	0
C	5	3	6	8	1	4	1
D	7	4	9	12	2	6	2
E	8	2	13	14	5	7	5
				Mitjana	**1,6**	**4,4**	**1,6**

Mira la D: arriba al cicle 7, entra al 9 i acaba al 12. Resposta 9 − 7 = **2**. Servei 12 − 7 + 1 = **6**. Espera 6 − 4 = **2**.

Avantatge: és simplíssim i no dona feina al sistema.

Inconvenient: l'**efecte comboi** (*convoy effect*). Si un procés molt llarg enxampa la CPU primer, tots els curts que vénen darrere esperen un munt. Mira la E: només necessita 2 cicles i n'espera 5.

> **Comprova sempre.** A la fila de cada procés, compta les caselles buides des que arriba fins que acaba: han de sortir igual que la seva espera. I si el primer arriba al cicle 1 i la CPU no s'atura mai, la Fi de l'últim és la suma de tots els cicles de CPU. Aquí: 3 + 2 + 3 + 4 + 2 = 14. Coincideix.

Guiat 4.5 — FIFO

Amb aquests processos, completa la graella, la cua i la taula. El P1 ja et ve fet.

Procés	Arribada	CPU	Inici	Fi	Resposta	Servei	Espera
P1	1	4	1	4	0	4	0
P2	2	3					
P3	3	2					
P4	6	5					
				Mitjana			

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
P1	X	X	X	F										
P2														
P3														
P4														
Cua														

---

10. Round Robin — tots per torns

Round Robin reparteix la CPU en torns iguals. Cada procés rep un **quantum** (q): un nombre fix de cicles. Quan se li acaba, **encara que no hagi acabat**, va al final de la cua i li toca al següent. És **expulsiu**.

Un procés deixa anar la CPU per tres motius:

1. Se li ha acabat el quantum.
2. Es bloqueja esperant una entrada/sortida.
3. Ha acabat abans d'esgotar el quantum.

Si dos processos han d'entrar alhora a la cua, passa primer **el que fa més temps que no s'executa**. Un que **arriba nou** passa davant del que **ha esgotat el quantum**: el nou no s'ha executat mai.

Exemple resolt 2 — Round Robin amb q = 2

Els mateixos cinc processos:

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	X	X	F											
B				X	F									
C						X	X					F		
D								X	X				X	F
E										X	F			
Cua					C		D	E	E	C	C	D		
								C	C	D	D			

Procés	Arribada	CPU	Inici	Fi	Resposta	Servei	Espera
A	1	3	1	3	0	3	0
B	4	2	4	5	0	2	0
C	5	3	6	12	1	8	5
D	7	4	8	14	1	8	4
E	8	2	10	11	2	4	2
				Mitjana	**0,8**	**5**	**2,2**

Mira el cicle 8: arriba E i C ha esgotat el quantum. Han d'entrar alhora a la cua: **primer E, que és nou**, i després C.

Fixa't en una cosa que no esperaves: amb aquestes dades Round Robin surt **pitjor** que FIFO en servei. La mitjana passa de 4,4 a 5.

Llavors per què es fa servir? Mira la **resposta**: amb Round Robin la mitjana és **0,8**, i amb FIFO, **1,6**. Tots entren aviat a la CPU, encara que sigui una estoneta. En un ordinador amb el qual estàs treballant, això és el que fa que no se't quedi congelat.

> **La regla del quantum.** Si el quantum és **molt gran**, Round Robin es comporta igual que FIFO. Si és **molt petit**, es perd moltíssim temps en canvis de context. Cal buscar el punt mitjà.

Practica 4.6 — Round Robin

Repeteix l'exemple anterior però amb **q = 3**. Completa la graella, la cua i la taula.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A														
B														
C														
D														
E														
Cua														

Procés	Arribada	CPU	Inici	Fi	Resposta	Servei	Espera
A	1	3					
B	4	2					
C	5	3					
D	7	4					
E	8	2					
				Mitjana			

Surt millor o pitjor que amb q = 2? Per què t'ho esperaves?

10.1. Quan un procés demana entrada/sortida

Fins ara els processos només feien servir la CPU. A la realitat, gairebé tots s'aturen a mig camí per llegir el disc, esperar una tecla o rebre dades de la xarxa. Mentre esperen **no volen CPU**: estan **bloquejats**.

Als exercicis t'ho donaran per **ràfegues**, així:

A: CPU 2 · E/S 2 · CPU 2

Vol dir que A fa servir la CPU 2 cicles, després s'està 2 cicles esperant l'entrada/sortida, i després torna a necessitar la CPU 2 cicles més. A classe ho veuràs escrit només amb números: **2 2 2**.

A la graella, els cicles d'E/S es marquen amb una **W** (de *wait*, esperar).

Tres regles noves:

1. **Quan un procés demana E/S, deixa anar la CPU a l'acte**, encara que no hagi esgotat el quantum. Entra el següent de la cua.
2. **Mentre fa E/S, no és a la cua.** L'E/S no ocupa la CPU: passa alhora que un altre procés s'executa.
3. **Quan acaba l'E/S, torna al final de la cua de preparats.** Mai directament a la CPU. És la fletxa de Bloquejat a Preparat del diagrama d'estats.

I els empats a la cua es resolen igual: passa primer **el que fa més temps que no s'executa**.

- Si un **arriba nou**, passa davant: no s'ha executat mai.
- Si un **torna d'E/S** i l'altre **ha esgotat el quantum**, primer el que torna d'E/S.
- Si **tornen dos d'E/S**, primer el que fa més temps que hi és.

> **Si a la cua no hi ha ningú**, la CPU queda aturada: en aquella columna de la graella no hi ha cap X.

I la fórmula de l'espera canvia una mica. Els cicles que un procés passa fent E/S no els ha passat a la cua, així que també s'han de restar:

Espera = Servei − CPU − E/S

on **CPU** és la suma de totes les ràfegues de CPU i **E/S** la suma de totes les d'entrada/sortida.

Exemple resolt 3 — Round Robin amb E/S, q = 2

Procés	Arribada	Ràfegues
A	1	CPU 3 · E/S 2 · CPU 1
B	2	CPU 2 · E/S 2 · CPU 2
C	3	CPU 2 · E/S 2 · CPU 2

Pas 1. Seguim la cua cicle a cicle. Aquest és el pas que no et pots saltar.

- **Cicle 1.** Només hi ha A. Entra a la CPU.
- **Cicle 2.** Arriba B. Va a la cua.
- **Cicle 3.** Arriba C i A ha esgotat el quantum, amb 1 cicle encara per fer: han d'entrar alhora a la cua. **Primer C, que és nou, després A.** Entra B. A la cua queden C i A.
- **Cicle 5.** B ha acabat la primera ràfega i **se'n va a fer E/S** els cicles 5 i 6. Entra C. A la cua, A.
- **Cicle 7.** C se'n va a fer E/S els cicles 7 i 8. B torna de l'E/S i va a la cua, darrere A. Entra A.
- **Cicle 8.** A ha acabat la primera ràfega i se'n va a fer E/S els cicles 8 i 9. Entra B.
- **Cicle 9.** C torna de l'E/S i va a la cua.
- **Cicle 10.** B ha acabat. A torna de l'E/S i va a la cua, darrere C. Entra C.
- **Cicle 12.** C ha acabat. Entra A, amb la segona ràfega, i acaba en aquest mateix cicle.

Pas 2. La graella.

	1	2	3	4	5	6	7	8	9	10	11	12
A	X	X					X	W	W			F
B			X	X	W	W		X	F			
C					X	X	W	W		X	F	
Cua		B	C	C	A	A	B		C	A	A	
			A	A								

Fixa't a la columna 8: A i C fan E/S alhora mentre B fa servir la CPU.

Pas 3. La taula, ara amb les ràfegues d'E/S:

Procés	Arribada	CPU i E/S	Inici	Fi	Resposta	Servei	Espera
A	1	3 · 2 · 1	1	12	0	12	6
B	2	2 · 2 · 2	3	9	1	8	2
C	3	2 · 2 · 2	5	11	2	9	3
				Mitjana	**1**	**9,66**	**3,66**

Fixa't en la A: fa servir la CPU **4** cicles, les dues ràfegues sumades (3 + 1), i fa **2** cicles d'E/S. Servei 12 − 1 + 1 = 12. Espera 12 − 4 − 2 = **6**.

> **Comprova sempre.** A la fila de la A, compta les caselles buides des que arriba fins que acaba: la 3, 4, 5, 6, 10 i 11. En són **6**, com l'espera. Si no coincideix, és que no has restat l'E/S.

Practica 4.7 — Round Robin amb E/S

Amb q = 2, completa la graella, la cua i la taula.

Procés	Arribada	Ràfegues
P1	1	CPU 2 · E/S 2 · CPU 1
P2	3	CPU 3 · E/S 2 · CPU 1
P3	4	CPU 3 · E/S 2 · CPU 1

	1	2	3	4	5	6	7	8	9	10	11	12
P1												
P2												
P3												
Cua												

Procés	Arribada	CPU i E/S	Inici	Fi	Resposta	Servei	Espera
P1	1	2 · 2 · 1					
P2	3	3 · 2 · 1					
P3	4	3 · 2 · 1					
				Mitjana			

a) Al cicle 5 dos processos han d'entrar alhora a la cua. Quins són, i en quin ordre hi entren?

b) A la columna 10 no hi ha cap X. Per què?

11. SJF — primer el més curt

SJF (*Shortest Job First*) mira els processos que ja han arribat i **tria el que menys cicles de CPU necessita**. És **no expulsiu**: un cop triat, el deixa acabar.

Exemple resolt 4 — SJF

Amb els mateixos cinc processos:

- **Cicle 1:** només hi ha A. S'executa dels cicles 1 a 3.
- **Cicle 4:** només hi ha B. Cicles 4 i 5.
- **Cicle 6:** només hi ha C (D arriba al 7 i E, al 8). Cicles 6 a 8.
- **Cicle 9:** hi ha D (necessita 4) i E (necessita 2). **Guanya E.** Cicles 9 i 10.

- **Cicle 11:** queda D. Cicles 11 a 14.

A la cua, ara, posem a dalt el més curt, que és el que entrarà.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	X	X	F											
B				X	F									
C						X	X	F						
D											X	X	X	F
E									X	F				
Cua					C		D	E	D	D				
								D						

Procés	Arribada	CPU	Inici	Fi	Resposta	Servei	Espera
A	1	3	1	3	0	3	0
B	4	2	4	5	0	2	0
C	5	3	6	8	1	4	1
D	7	4	11	14	4	8	4
E	8	2	9	10	1	3	1
				Mitjana	**1,2**	**4**	**1,2**

SJF guanya en servei: 4 de mitjana, davant de 4,4 de FIFO i 5 de Round Robin.

> **El problema de l'SJF.** Si no paren d'arribar processos curts, un de llarg pot quedar-se esperant per sempre. Això es diu **inanició** (*starvation*).

I hi ha un altre problema, més pràctic: **el sistema no sap per endavant quant trigarà un procés.** L'ha d'estimar. Per això l'SJF pur es fa servir poc fora dels exercicis.

---

12. SRT — el més curt, i a més amb expulsió

SRT (*Shortest Remaining Time*) és SJF però **expulsiu**. Cada vegada que arriba un procés nou, el sistema compara:

El que necessita el nouvingut és menys que el que li queda al que és a la CPU?

Si la resposta és sí, li lleva la CPU al que hi era i l'hi dona al nou.

Amb els cinc processos de sempre, SRT dona el mateix que SJF

Comprova-ho tu: cada vegada que n'arriba un de nou, al que és a la CPU sempre li queda menys. Mai no es compleix la condició d'expulsar. Per això el resultat és idèntic.

SRT només es diferencia d'SJF quan arriba un procés més curt que el que li falta al que s'està executant. Anem a veure-ho amb un cas que sí que ho provoca.

Exemple resolt 5 — SRT davant d'SJF

Procés	Arribada	CPU
P1	1	7
P2	3	3

Amb SJF (no expulsiu): P1 agafa la CPU al cicle 1 i no la deixa anar. P1 dels cicles 1 a 7, P2 del 8 al 10.

	1	2	3	4	5	6	7	8	9	10
P1	X	X	X	X	X	X	F			
P2								X	X	F
Cua			P2	P2	P2	P2	P2			

Procés	Arribada	CPU	Inici	Fi	Resposta	Servei	Espera
P1	1	7	1	7	0	7	0
P2	3	3	8	10	5	8	5
				Mitjana	**2,5**	**7,5**	**2,5**

Amb SRT (expulsiu): al cicle 3 arriba P2 i necessita 3. A P1 li'n queden 5. Com que 3 < 5, **P2 expulsa P1**. P2 va dels cicles 3 a 5, i P1 recupera la CPU del 6 al 10.

	1	2	3	4	5	6	7	8	9	10
P1	X	X				X	X	X	X	F
P2			X	X	F					
Cua			P1	P1	P1					

Procés	Arribada	CPU	Inici	Fi	Resposta	Servei	Espera
P1	1	7	1	10	0	10	3
P2	3	3	3	5	0	3	0
				Mitjana	**0**	**6,5**	**1,5**

SRT millora la mitjana. A canvi, P1 acaba més tard i hi ha hagut un canvi de context de més.

---

13. Els quatre, un al costat de l'altre

Algorisme	Expulsiu?	Just?	Espera mitjana	Risc d'inanició	Es fa servir en
FIFO	No	Sí	Alta	No	Sistemes simples, cues d'impressió
Round Robin	Sí	Sí	Mitjana	No	Sistemes interactius
SJF	No	No	Baixa	**Sí**	Processos per lots amb durada coneguda
SRT	Sí	No	**Molt baixa**	**Sí**	Sistemes de temps real

I amb els nostres cinc processos:

	FIFO	RR (q=2)	SJF	SRT
Resposta mitjana	1,6	0,8	1,2	1,2
Servei mitjà	4,4	5	4	4
Espera mitjana	1,6	2,2	1,2	1,2

La conclusió que cal treure. No existeix el millor algorisme. Existeix el que va millor **per al que vols**. SJF dona el millor servei i la millor espera, però mata de gana els processos llargs. Round Robin dona pitjor servei, però és el que respon més de pressa: per això l'ordinador et respon mentre treballes.

Practica 4.8 — Els quatre amb les mateixes dades

Procés	Arribada	CPU
P1	1	5
P2	2	2
P3	3	4
P4	5	1

Resol els quatre algorismes (Round Robin amb q = 2) i compara les mitjanes.

FIFO

	1	2	3	4	5	6	7	8	9	10	11	12
P1												
P2												
P3												
P4												
Cua												

Round Robin, q = 2

	1	2	3	4	5	6	7	8	9	10	11	12
P1												
P2												
P3												
P4												
Cua												

SJF

	1	2	3	4	5	6	7	8	9	10	11	12
P1												
P2												
P3												
P4												
Cua												

SRT

	1	2	3	4	5	6	7	8	9	10	11	12
P1												
P2												
P3												
P4												
Cua												

	FIFO	RR (q=2)	SJF	SRT
Resposta mitjana				
Servei mitjà				
Espera mitjana				

Quin dona la millor mitjana d'espera? Quin triaries per a un ordinador de sobretaula i per què?

14. Els processos del teu ordinador

Tot això no és teoria llunyana: ho pots veure ara mateix.

14.1. A Linux, des del terminal

Ordre	Què fa
top	Mostra els processos actius en temps real, amb la seva CPU i memòria
ps -aux	Llista tots els processos del sistema amb el seu PID
kill -9 PID	Força el tancament immediat d'un procés
kill -STOP PID	Suspèn un procés, el deixa adormit
kill -CONT PID	El reprèn
Ctrl+Z	Envia el procés actual a segon pla
bg / **fg**	El passa a segon pla / el porta al primer

Així es veu un **ps -aux**:

```
USER       PID %CPU %MEM COMMAND
root         1  0.0  0.1 /sbin/init
alumne    1234  2.3  1.5 firefox
alumne    1235  0.1  0.3 bash
```

La primera columna és qui el va llançar i la segona és el **PID**, el número que necessites per a qualsevol ordre **kill**.

14.2. A Windows

L'**Administrador de tasques** (Ctrl+Shift+Esc) ensenya el mateix amb finestres. A la pestanya *Detalls* hi veuràs el PID de cada procés.

Practica 4.9 — A l'ordinador

a) Obre un terminal i executa **ps -aux | grep firefox**. Apunta el PID que et surti.

b) Executa **top**. Quants processos hi ha en total? Quants estan realment en execució?

c) Si a **top** hi ha 120 processos i només 1 en execució, en quin estat estan els altres 119?

d) Obre l'Administrador de tasques de Windows i busca un procés que estigui al 0 % de CPU.
Quin estat del diagrama li correspon?

El que has de saber sí o sí

- **Programa** és el fitxer al disc. **Procés** és el programa executant-se.
- Els **fils** d'un procés comparteixen memòria; els **processos** entre si, no.
- El **PCB** és la fitxa del procés: PID, estat, comptador de programa i registres.
- El **canvi de context** guarda un procés i en carrega un altre. Costa temps.
- Cinc estats: **Nou, Preparat, En execució, Bloquejat, Acabat**.
- De **Bloquejat** es passa a **Preparat**, mai directament a En execució.
- **Resposta = Inici − Arribada · Servei = Fi − Arribada + 1 · Espera = Servei − CPU**
- Primer la graella, després la taula. Mai al revés.
- **FIFO** per ordre d'arribada, no expulsiu, pateix efecte comboi.
- **Round Robin** per torns de quantum, expulsiu, el més just.
- Amb **E/S**, el procés deixa la CPU a l'acte, no fa cua mentre espera i, quan acaba, torna al **final** de la cua. **Espera = Servei − CPU − E/S**.
- **Empats a la cua:** passa primer el que fa més temps que no s'executa.
- **SJF** el més curt primer, no expulsiu, millors mitjanes però pot causar inanició.
- **SRT** com SJF però expulsiu: expulsa si n'arriba un de més curt que el que queda.

Paraules noves

Paraula	Què és	Les meves notes
Procés	Programa en execució, amb la seva memòria i els seus recursos.	
Fil	Línia d'execució dins d'un procés. Comparteix la seva memòria.	
PID	Número únic que identifica un procés.	
PCB	Fitxa on el SO guarda tot el del procés.	
Canvi de context	Guardar un procés i carregar-ne un altre a la CPU.	
Planificador *(scheduler)*	La part del SO que decideix qui entra a la CPU.	
Distribuïdor *(dispatcher)*	La part del nucli que dona la CPU al procés triat.	
Cicle	Unitat de temps del rellotge. A la graella, cada columna.	
Temps de resposta	Cicles que passen des que arriba un procés fins que entra per primer cop a la CPU.	
Quantum	Nombre fix de cicles de CPU que rep cada torn en Round Robin.	
Ràfega	Tros seguit de temps de CPU, o d'E/S, d'un procés.	
Expulsiu	El sistema pot llevar la CPU a un procés sense que acabi.	
Efecte comboi	Processos curts encallats darrere d'un de llarg.	

Paraula	Què és	Les meves notes
Inanició	Un procés que no arriba mai a executar-se.	

Repàs del tema

A. Vertader o fals

Frase	V o F	Per què
Un programa i un procés són el mateix		
Un procés bloquejat passa directament a execució		
Round Robin és expulsiu		
SJF pot provocar inanició		
Els fils d'un procés comparteixen la pila		
El canvi de context no costa temps		

B. Preguntes curtes

1. Què guarda el sistema al PCB abans d'un canvi de context?

2. Per què un quantum molt petit és mala idea?

3. Quina diferència hi ha entre SJF i SRT?

4. Explica l'efecte comboi amb un exemple teu.

C. Problemes

Problema 1. Amb aquests processos, resol FIFO: la graella, la cua i les tres mitjanes.

Procés	Arribada	CPU	Inici	Fi	Resposta	Servei	Espera
A	1	6					
B	2	3					
C	4	1					
D	6	4					
				Mitjana			

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A														
B														
C														
D														
Cua														

Problema 2. Els mateixos processos amb Round Robin i q = 2.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A														
B														
C														
D														
Cua														

Procés	Arribada	CPU	Inici	Fi	Resposta	Servei	Espera
A	1	6					
B	2	3					
C	4	1					
D	6	4					
				Mitjana			

Problema 3. Els mateixos amb SJF. Quin dels tres dona la menor espera mitjana?

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A														
B														
C														
D														
Cua														

Procés	Arribada	CPU	Inici	Fi	Resposta	Servei	Espera
A	1	6					
B	2	3					
C	4	1					
D	6	4					
				Mitjana			

Problema 4. Un procés està esperant que el disc li torni un fitxer. Mentre espera, un altre procés fa servir la CPU. Explica en quin estat està cadascun i què passarà quan el disc contesti.

Les meves notes