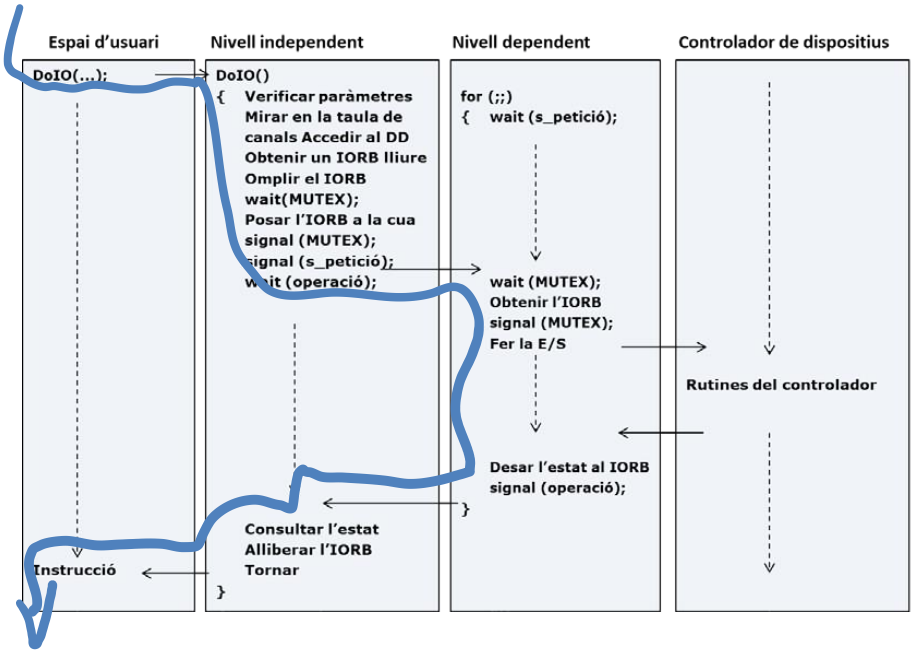
PROJECTE DE SISTEMES OPERATIUS.
Grau en Enginyeria Informàtica 3r Curs
2on parcial, 15 de desembre de 2020, Temes 3, 4 i 5

Cognoms:_____**Nom:** _____

Justifica totes les respostes.

1. **(3,50 punts) Entrada/Sortida.**



a) (0,75 punts) Marca l'opció correcta:

1. El codi anterior és?	**síncron**	~~asíncron~~
2. Permet servir les peticions d'un procés en ordre LIFO?	~~Sí~~	**No**
3. Permet servir les peticions d'un procés en ordre FIFO?	**Sí**	~~No~~
4. Permet servir les peticions de diferents processos en ordre LIFO?	**Sí**	~~No~~
5. Permet servir les peticions de diferents processos en ordre FIFO?	**Sí**	~~No~~

El codi anterior només permet servir les E/S de forma síncrona ja que el procés d'usuari espera la finalització de l'operació d'E/S per continuar la seva execució.

Un procés només pot executar operacions d'E/S en el mateix ordre que les demana ja que són síncrones. Només en pot demanar una i esperar la seva finalització per demanar la següent.

L'ordre en el que es serviran les peticions entre processos depèn de l'ordre en el que el gestor les agafa de la cua de IORB. Per tant pot ser qualsevol ordre LIFO, FIFO, ...

b) (0,50 punts) Quants ***threads*** (fluxos d'instruccions independents) hi ha a l'esquema anterior? Què permet aquest fet?

Si es mira l'esquema amb un únic procés d'usuari, podem dir que hi ha un únic flux ja que només s'executa una instrucció en cada instant. Podem veure el flux en la línia afegida a la figura.

No obstant hi ha 2 processos implicats: el procés d'usuari i el Gestor, a l'esquema estan sincronitzats i cadascun d'ells espera a que l'altre faci la seva feina i l'avisi per continuar la seva execució.

c) (0,25 punts) El codi d'un mòdul de Linux s'executa en espai de SO o d'usuari? Des del codi del mòdul es pot utilitzar el ***printf*** ?

Els mòduls s'han d'executar en mode Sistema ja que han de poder accedir al hardware i a les estructures de dades del SO, per tan s'executen des de l'espai del SO en mode privilegiat.

Des del SO no es pot executar codi d'usuari o codi en l'espai d'usuari (explicat en la resposta següent). Printf és un procediment de la llibreria de C que acaba fent una crida al SO. Des del SO no es pot invocar al propi SO.

d) (0,25 punts) Els mòduls de Linux poden executar codi d'usuari? En cas afirmatiu, com?

Per motius de seguretat, des del codi del SO no és pot invocar cap procediment que estigui en espai d'usuari. EL motiu és de seguretat. No es pot executar codi d'un usuari amb els privilegis del mode privilegiat. Aquest fet podria donar accés a tot el sistema i a totes les dades de tots els usuaris deixant desprotegit al SO.

e) (0,25 punts) Els mòduls de Linux contenen un fitxer executable o un fitxer objecte?

Els mòduls de Linux són un fitxer objecte, en concret un Kernel Object (ko). No tenen un programa principal. Són un conjunt de procediments que al executar la comanda insmod es carreguen a memòria del SO i s'enllacen amb la resta del SO per poder-se cridar entre ells (SO i mòdul).

f) (0,75 punts) En relació a la comanda: ***mknod nom c 27 2***

1. Per què s'utilitza?

   La comanda mknod crea un nom per a un dispositiu real a l'arbre de directoris. Al fer-ho guarda a l'inode relacionat el tipus de dispositiu: el major (27 en aquest cas) i el minor (2 en aquest cas). Aquests dos valors serviran perquè, quan un procés executi un "open" d'aquest nom, el SO sàpiga identificar el dispositiu, i per tant el driver (part dependent) que haurà d'utilitzar.

2. La comanda ***mknod*** quina relació té amb els mòduls de Linux?

   El mòdul quan es fa el insmod, en l'execució del procediment inicial, registra el dispositiu en el SO amb un major.

   El major és el nombre que identifica el DD (fileops) del dispositiu i permet que la capa independent del SO pugui executar el procediments de la part dependent del dispositiu.

3. S'ha d'utilitzar abans o després de carregar el mòdul?

   La comanda mknod es pot executar abans o després d'instal·lar el driver. Però abans de fer el open del dispositiu el mòdul ha d'estar instal·lat i el nom creat.

4. Quin significat té el ***major*** (27) i el ***minor*** (2) ?

   El major indica el tipus de dispositiu real i el minor la instància d'aquest.

g) (0,50 punts) Els mòduls implementen la part independent dels dispositius?

Els mòduls de Linux implementen la part dependent del dispositiu que es cridada per la part independent. La part independent implementa les operacions uniformes com read o write.

h) (0,25 punts) En els mòduls de Linux com es fa per a que des de les operacions independents és puguin cridar les operacions dependents del dispositiu?

El mòdul quan es fa el insmod, en l'execució del procediment inicial, registra el dispositiu en el SO amb un major.

El major és el nombre que identifica el DD (fileops) del dispositiu i permet que la capa independent del SO pugui executar el procediments de la part dependent del dispositiu.

El procediment independent, un cop identificat el canal pot accedir a la taula de fitxers obert o taula d'inodes. En aquesta taula hi ha la còpia de l'inode del dispositiu que conté el major i el minor del dispositiu. Ara amb el major ja es pot accedir al DD (fileops) i localitzar els procediments de la part dependent que implementa el mòdul.

2. (3,00 punts) Sistemes de Fitxers.

Es disposa d'un Sistema de Fitxers a on les dades associades a un fitxer s'organitzen de forma contigua. El **punter al primer bloc d'un fitxer és de 8 bits** i la **longitud** en bytes del fitxer es guarda en un enter positiu de **8 bits**. Els blocs del dispositiu són de **2 KBytes**. L'**estructura d'un fitxer** ocupa **16 bytes** i es dediquen **2 blocs de disc a guardar totes les estructures de fitxers del SF.**

a) (0,25 punts) De què depèn el **nombre màxim de fitxers** que pot contenir aquest SF?

#Blocs = 2^7 bytes/Bloc = 2^11 bytes/inode = 2^4

Mínim entre el nombre d'estructures (inodes) de fitxers del SF, i entre el nombre de blocs lliures

d'Estructures de fitxer = (2 * 2^11 bytes/Bloc)/ (2^4 bytes/inode) = 2^8 inodes

de blocs lliures = **2^7 – (1 SuperBloc + 1 Bloc del dir root + 2 blocs d'inodes). blocs**

b) (0,25 punts) Quin és el **nombre màxim de bytes d'un fitxer**? Quant blocs de dades són necessaris?

Mínim entre els bytes adreçables des del camp longitud, i entre el nombre de bytes dels blocs lliures

Bytes adreçables pel camp longitud = **2^8 bytes**

de bytes dels blocs lliures = (2^7 -4) bloc * 2^11 bytes/Bloc

Cada fitxer ocupa com a molt un bloc.

c) (0,25 punts) Quin és el **nombre de bytes màxim del disc**? De què depèn?

Bytes de dics = 2^7 blocs * 2^11 bytes/blocs = **2^18 bytes**

d) (0,50 punts) Com es pot gestionar l'**espai lliure d'aquest SF**? Amb la solució que has proposat quant d'espai del SF s'hi ha de dedicar?

1) Si s'utilitza un **mapa de bits** es necessita un bit per bloc ,per tant 2^7 bits

El mapa de bits ocupa en bytes = (2^7 bits)/(2^3 bits /byte) = 2^4 bytes = **16 bytes**

Es pot suposar que el mapa de bits es guardarà en el Súper bloc.

2) Un alternativa seria un **cadena de blocs lliures**. Per aquesta solució es necessiten 2 punters de 8 bits (cadena doblement encadenada) i podrien ser en el Súper bloc.

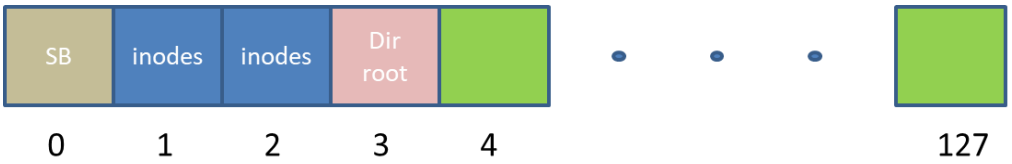
3) Seria menys adequada una **llistat de blocs lliures**, ja que requereix 2^7 punters d'un byte per tant ocuparia **128 bytes** i també podrien ser en el Súper bloc.

e) (0,25 punts) indica un **inconvenient** d'aquest SF?

El principal inconvenient és el camp longitud que no permet tenir fitxers més grans de 256 bytes. És perd molt d'espai per fragmentació interna (cada bloc és de 2 Kbytes).

Si aquest inconvenient no hi fos, el problema podria ser l'estructura contigua que no és apta per un SF dinàmic amb creació i eliminació de fitxers, ni amb fitxers que poden augmentar de contingut. El motiu és que una estructura contigua genera fragmentació externa i obliga a fer una gestió complexa de l'espai.

f) (0,50 punts) Fes un e**squema d'aquest SF** a on es vegi a que es dedica cada bloc.

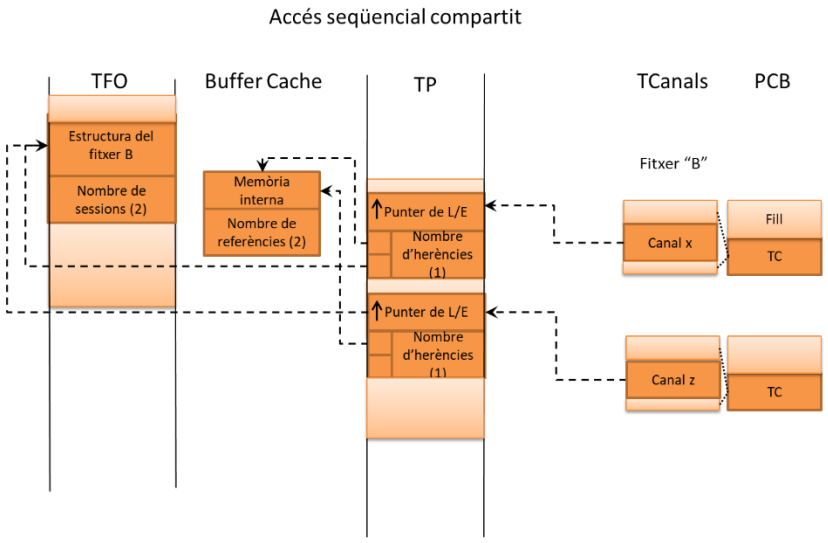


g) (0,25 punts) En aquest SF es vol obrir el fitxer "**/A/B/C**" a on tots els fitxer implicats són directoris menys el fitxer "**C**" que és un fitxer ordinari. Quants accessos a disc farà falta fer si no es disposa de cap informació prèviament carregada a memòria del SO?

1r accés inode arrel
2n accés dades del directori arrel
3r accés inode de A
4t accés dades del directori A
5e accés inode de B
6è accés dades del directori B
7è accés inode de C

No fa falta accedir a les dades del fitxer C ja que l'operació que s'està fent és obrir el fitxer.

h) (0,75 punts) En el cas de la pregunta anterior, un cop obert, quines **estructures de dades del SO** estan involucrades amb aquesta obertura i què contenen si aquest sistema és d'**accés seqüencial compartit**.

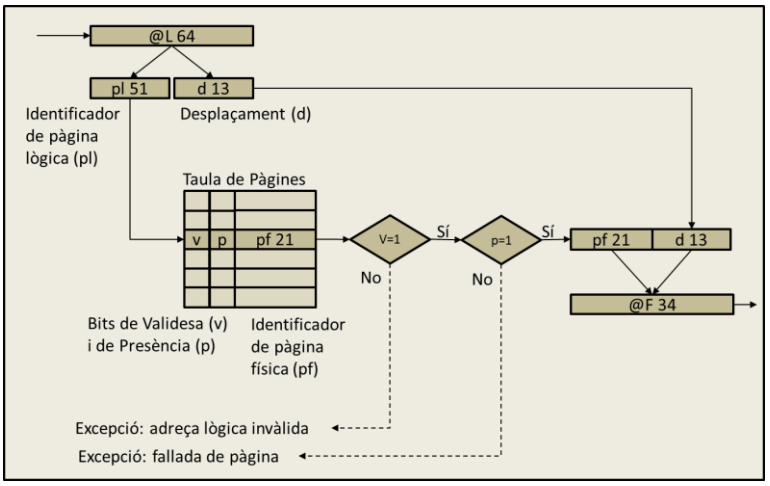


3. (3,50 punts) Memòria Virtual.

Tenim un sistema amb memòria **Virtual Paginada sota demanda**. Tenim que un procés fa la següent seqüència de referències a pàgina.

1	4	3	4	1	1	3	1	2	4	3	1	1	4	1	5	7	5	6	8	5	1	6	8	5	6	7	8	6	7

a) (0,50 punts) Si **l'adreça lògica és de 64 bits**, la memòria física és de **16 GBytes** i es té una MMU **paginada** amb pàgines de **8 kBytes,** com és la taula de pàgines? Indica: nombre d'entrades i els camps de cada entrada.



El nombre d'entrades de la Taula de pàgines és 2^51 i cada entrada té 23 bits, arrodonits a 3 bytes

b) (0,25 punts) Quines característiques ha de tenir una MMU per permetre Memòria Virtual?

Com a mínim ha de ser una MMU paginada o amb pàgines (segmentació paginada, paginació amb multinivells)
Ha de tenir el bit de P de presència
Ha de generar interrupcions en el cas de fallada de pàgina.

c) (0,25 punts) Què ha de fer el SO quan es produeix una fallada de pàgina. Indica totes les accions que podria haver de fer.

Suposem un algoritme de MV sota demanda amb assignació local.
Quan es produeix una fallada de pàgina la MMU genera una interrupció. Associada a aquesta interrupció s'executa el procediment que identificarà la pàgina que es necessita a memòria i la seva localització a l'àrea de swap.

Es determinarà en quina pàgina física ha d'anar. Per fer-ho pot haver d'aplicar la política d'assignació com el Workingset i la de reemplaçament. En el cas d'haver de reemplaçar una pàgina, haurà de determinar si ha estat modificada en memòria i, si és el cas l'haurà de guardar a l'àrea de swap.

Ara ja podrà llegir la pàgina de disc, portar-la a memòria i actualitzar les taules de pàgines.

Finalment podrà deixar que el procés que ha tingut la fallada de pàgina pugui continuar la seva execució.

d) (0,25 punts) El procés que ha provocat la fallada de pàgina, per quins estats creus que passarà (Run, Ready, Wait) des de que es produeix la fallada de pàgina fins a que pot tornar a executar-se?

El procés passarà de RUN a WAIT a l'espera que el SO resolgui la fallada de pàgina (veure pregunta anterior).

Un cop resolt passarà a READY i quan la política de scheduling ho determini podrà tornar a RUN.

e) (0,25 punts) Per quins motius un Sistema amb molta memòria física podria entrar en **Thrashing** i com es podria evitar?

Un SO és en thrashing quan executa majoritàriament instruccions de SO en comptes d'instruccions d'usuari.

En aquest cas podria ser que cada procés se li donés molta menys memòria física de la que necessita per un mal algoritme d'assignació o que hi hagués un grau de multiprogramació massa gran que no permetés donar un mínim de memòria a cada procés. Es podria evitar, en el primer cas, millorant la política d'assignació o, en el segon cas, reduint el grau de multiprogramació i/o augmentant la memòria física.

f) (1,00 punts) Quantes fallades de pàgina es produiran si es fa una assignació dinàmica de trames amb una finestra de **WorkingSet de 4** trames i l'algoritme de reemplaçament és **Óptim**? Utilitza la taula adjunta.

g) (1,00 punts) Quantes fallades de pàgina es produiran si es fa una **assignació estàtica** de trames de **3 trames** i l'algoritme de reemplaçament és **FIFO amb segona oportunitat**? Utilitza la taula adjunta.

ref	**1**	**4**	**3**	**4**	**1**	**1**	**3**	**1**	**2**	**4**	**3**	**1**	**1**	**4**	**1**	**5**	**7**	**5**	**6**	**8**	**5**	**1**	**6**	**8**	**5**	**6**	**7**	**8**	**6**	**7**
Opt	1	4	3	3	3	3	3	4	2	2	3	3	2	2	4	5	7	7	6	8	6	6	6	8	8	8	7	7	7	7
		1	4	4	4	4	4	1	4	4	2	2	4	4	1	4	5	5	5	6	5	5	5	6	6	6	8	8	6	6
			1	1	1	1	1		1	1	4	4	1	1		1	4	1	1	5	1	1	1	5	5	5	6	6	5	5
										?	1	1					1			1		?	?	1	1		5	5		
F	**F**	**F**	**F**						**F**		**F**					**F**	**F**		**F**	**F**				**F**			**F**			
WS 4	**1**	**2**	**3**	**3**	**3**	**3**	**3**	**2**	**3**	**4**	**4**	**4**	**3**	**3**	**2**	**3**	**4**	**3**	**3**	**4**	**3**	**4**	**4**	**4**	**4**	**3**	**4**	**4**	**3**	**3**
	1	4	3	4	1	1	3	1	2	4	3	1	1	4	1	5	7	5	6	8	5	1	6	8	5	6	7	8	6	7
		1	4	3	4	1	1	3	1	2	4	3	1	1	4	1	5	7	5	6	8	5	1	6	8	5	6	7	8	6
			1	4	3	4	1	1	3	1	2	4	3	1	1	4	1	5	7	5	6	8	5	1	6	8	5	6	7	8
				1	4	3	4	1	1	3	1	2	4	3	1	1	4	1	5	7	5	6	8	5	1	6	8	5	6	7
ref	**1**	**4**	**3**	**4**	**1**	**1**	**3**	**1**	**2**	**4**	**3**	**1**	**1**	**4**	**1**	**5**	**7**	**5**	**6**	**8**	**5**	**1**	**6**	**8**	**5**	**6**	**7**	**8**	**6**	**7**
FIFO 2	1	4	3	3	3	3	3	3	2	2	2	1	1	4	4	5	7	7	6	8	5	1	6	8	5	5	7	7	6	6
		1	4	4	4	4	4	4	3	3	3	2	2	1	1	4	5	5	7	6	8	5	1	6	8	8	5	5	8	8
			1	1	1	1	1	1	4	4	4	3	3	2	2	1	4	4	5	7	6	8	5	1	6	6	8	8	7	7
F	**F**	**F**	**F**						**F**			**F**		**F**		**F**	**F**		**F**	**F**	**F**	**F**	**F**	**F**	**F**		**F**		**F**	

El quadrats blancs tenen el bit R a 1.

Els quadrats grocs tenen el bit R a 0.