PRÀCTICA

1. (**2.5 p**) **Script Bash**. Heu de fer un script (guió), de nom `calc_bash.sh`, que executarà una simple calculadora amb les operacions bàsiques (`+,-,x,/`), amb els següents requeriments:

   - La crida a aquest script haurà de rebre zero paràmetres o únicament "`-h`", opció que farà que es mostri una petita ajuda sobre l'aplicació. Per mostrar informació sobre l'ús correcte de l'script i la informació d'ajuda s'han d'implementar les funcions us i ajuda, que caldrà invocar dins l'script on convingui. A la funció d'ajuda no cal que fiqueu el text que es mostra als exemples, fiqueu simplement "text d'ajuda".
   - Un cop s'executi l'script es demanarà que s'entri per teclat un operand, l'operador i un altre operand: `operand1 {+|-|x|X|/} operador2`. El producte es pot indicar amb 'x' o 'X'.
   - Cal controlar que l'operador sigui correcte. NO cal controlar que els operands siguin nombres.
   - Si l'operador és correcte, es mostrarà el resultat; en cas contrari es mostrarà un missatge indicant que s'ha introduït un operador incorrecte.
   - Tot seguit es demanarà a l'usuari si vol sortir, o si vol continuar fent operacions. Cal demanar l'opció a l'usuari i controlar que l'opció entrada sigui 's' o 'n', i sortir o no del guió segons l'opció triada.
   - Tot el que no estigui indicat, podeu prendre les vostres decisions.

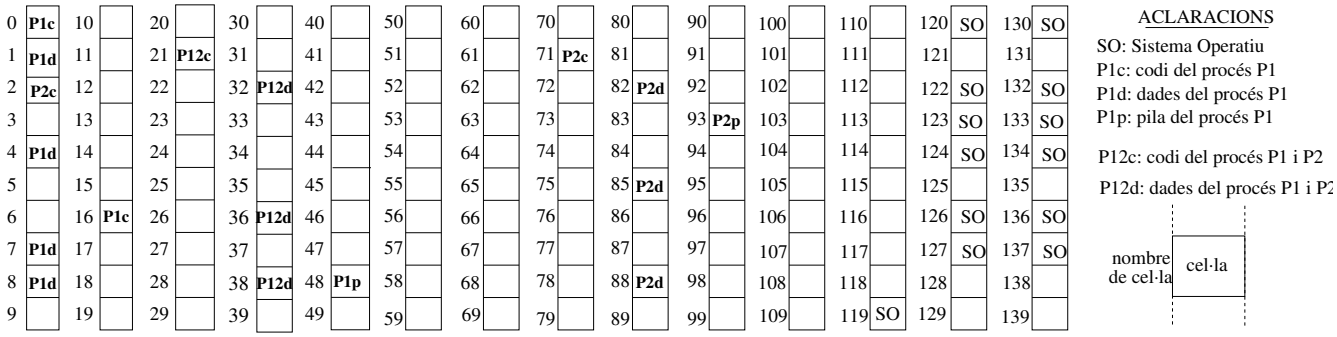
   La crida a l'script és: `# calc_bash.sh [-h]`

   Exemples d'execució (en **negreta** es mostra la informació que subministra l'usuari):

```
# bash_calc.sh -h
 bash_calc.sh es una calculadora d'operacions basiques (+,-,x,/):
   - No admet parametres a excepcio d'aquesta ajuda.
   - Els operands i operador es demanen dins el programa en execucio.
   - S'executa en bucle fins que decideixis sortir.
# bash_calc.sh 23
Us: ./bash_calc.sh [-h]
# bash_calc.sh 23 67 1
Us: ./bash_calc.sh [-h]
# bash_calc.sh h
Us: ./bash_calc.sh [-h]
# bash_calc.sh
Introdueix operand1 {+|-|x|X|/} operand2 i ENTER: 1 + 2
1 + 2 = 3
Vols continuar?  [s/n] t
Vols continuar?  [s/n] u
Vols continuar?  [s/n] s
Introdueix operand1 {+|-|x|X|/} operand2 i ENTER: 1 A 3
ERROR - Operador 'A' no valid
Vols continuar?  [s/n] s
Introdueix operand1 {+|-|x|X|/} operand2 i ENTER: 3 x 4
3 x 4 = 12
Vols continuar?  [s/n] n
#
```

TEORIA

1. **(3 p) Gestió de Memòria**. Disposem d'un sistema de gestió de la memòria del tipus segmentació paginada. La mida d'una pàgina (cel·la) és de 1.024 paraules. Un segment conté com a molt 3 pàgines. Un procés consta de com a molt 8 segments. La mida d'una paraula és igual a l Byte. En la següent figura es pot veure el contingut de MP:



   (a) **(2 p)** Doneu un exemple de la informació que ha de guardar el sistema operatiu per gestionar la memòria del procés P_1 i P_2. Suposeu que totes les pàgines del procés P_1 estàn plenes. Tota aquesta informació s'ha de posar a la cel·la 9 i ha d'ocupar el mínim espai possible. Digueu què és cada dada que emmagatzemeu, així com la seva adreça física de MP i la seva mida en Bytes.

   (b) **(1 p)** Doneu l'esquema de traducció d'adreces en el cas que la taula de segments s'implementi en registres i les taules de pàgines en Memòria Principal.

2. **(2.5 p) Memòria Virtual.** Disposem d'un sistema de gestió de la memòria del tipus **"`paginació`"** i Mem. Virtual del tipus **"`paginació sota demanda`"**. Assignació local i igualitària de 5 cel·les per procés. La mida d'una pàgina i d'una cel·la és de 16 Bytes. Mida d'una instrucció = Mida d'una paraula = Mida d'un enter = 1 Byte. Donat el programa en Pseudo-assembler següent:

@log	**codi**
017h	Reg0 ← 1
018h	Comparar Reg0, 35
019h	Branch_equal to @024h
01Ah	Reg1 ← Reg0 mod 16
01Bh	Reg2 ← Reg0 - 1
01Ch	Reg3 ← Reg0 + 1
01Dh	Load Reg4 ← Mem(@500h + Reg0)
01Eh	Load Reg5 ← Mem(@600h + Reg1)
01Fh	Load Reg6 ← Mem(@700h + Reg2)
020h	Reg7 ← Reg4 + Reg5 + Reg6
021h	Store Mem(@500h + Reg3) ← Reg7
022h	Reg0 ← Reg0 + 1
023h	Branch to @018h

   (a) **(1.25 p)** Quantes fallades de pàgina es produirà en l'execució del programa suposant que l'algorisme de reemplaç és LRU? Feu l'esquema que ens mostri com es van omplint les cel·les a mesura que es van produint les fallades de pàgina.

   (b) **(1.25 p)** Quantes fallades de pàgina es produirà en l'execució del programa suposant ara que l'algorisme de reemplaç és FIFO?

3. (**1 p**) Memòria Virtual. Considereu que tenim un sistema informàtic amb memòria virtual (del tipus paginació sota demanda). El rendiment (nombre de processos executats per unitat de temps) és molt baix i el disc de paginació està treballant al 90%. Què està passant? Doneu 3 solucions, relacionades amb Memòria Virtual, que probablement augmentarien el rendiment del sistema.

4. (**1 p**) **Interbloqueig.**

(a) **(0.5 p) Evitació**. Tenim 5 processos $\{p_0, p_1, p_2, p_3, p_4\}$ i 3 tipus de recursos $\{r_0 = 10, r_1 = 8, r_2 = 4\}$. Segons l'Algorisme del Banquer i de Seguretat es serviría la sol·licitut $Sol{\cdot}licitut_3 = (1, 1, 1)$ donat l'Estat General següent?

	Assignat			Necessitat		
	r_0	r_1	r_2	r_0	r_1	r_2
p_0	3	2	1	1	1	3
p_1	2	1	1	7	7	3
p_2	0	0	0	5	0	1
p_3	1	3	0	4	1	1
p_4	3	0	1	0	1	0

Disponible		
r_0	r_1	r_2
1	2	1

(b) **(0.5 p) Detecció** 1 **instància de cada tipus de recurs**. Donat el graf d'espera de la figura, doneu tots els processos que estan intebloquejats.