Espai grapa

Examen 2017/18-1

Assignatura	Codi	Data	Hora inici
Sistemes operatius	05.566	10/01/2018	12:00

05.566 10 01 18 EX

Enganxeu en aquest espai una etiqueta identificativa amb el vostre codi personal
Examen

Fitxa tècnica de l'examen

- **Comprova que el codi i el nom de l'assignatura corresponen a l'assignatura en la qual estàs matriculat.**
- **Només has d'enganxar una etiqueta d'estudiant a l'espai corresponent d'aquest full.**
- **No es poden adjuntar fulls addicionals.**
- **No es pot realitzar la prova en llapis ni en retolador gruixut.**
- **Temps total:** 2 **h.**
- **En cas que els estudiants puguin consultar algun material durant l'examen, quin o quins materials poden consultar?**
  Un full mida foli/DIN A-4 amb anotacions per les dues cares
- **Valor de cada pregunta:** 2,5 punts
- **En cas que hi hagi preguntes tipus test: Descompten les respostes errònies?** NO **Quant?**

- **Indicacions específiques per a la realització d'aquest examen:**
  Poden portar calculador per realitzar l'examen.

Enunciats

Examen 2017/18-1

Assignatura	Codi	Data	Hora inici
Sistemes operatius	05.566	10/01/2018	12:00

1. Teoria [2.5 punts]

Contesteu **justificadament** les següents preguntes:

a) Donat el següent estat inicial:

```
int x, y;
sem_t semA;
x = 1;
y = 3;
sem_init(semA, 0, 1);  // S'inicialitza el semàfor SemA a 1 (una instància)
```

Enumerar els possibles valors de x per la següent execució, després que els dos fils (Thread1 i Thread2) hagin acabat o es bloquegin. Indicar també si algun dels fils es bloqueja indefinidament i perquè.

```
Thread1:
   x = x+1;
```

```
Thread2:
   x = x+2;
```

En aquest codi tenim dos fils que accedeixen a la variable compartida x. En no implementar seccions crítiques es pot produir condicions de carrera en la modificació de la variable x.

En aquest cas els possibles valors que podem obtenir de la variable x serien:

- x = 4, si la modificació de x del fil 1 es realitza abans o després que s'hagi completat l'operació del fil 2.
- x = 3, si l'operació del fil 1 es realitza enmig de l'operació del fil 2.
- x = 2, si l'operació del fil 2 es realitza enmig de l'operació del fil 1.

Cap fil es bloqueja indefinidament.

b) ¿Indiqueu quants processos, pipes i redireccions necessita l'intèrpret de comandes per executar la següent comanda?

```
$ ps –edaf | sort > dades.txt
```

Es necessiten dos processos, un per executar la comanda ps i un altre per executar la comanda sort. També es necessita un pipe per connectar la stdout del procés ps a la stdin del procés sort. Finalment, es necessita realitzar 3 redireccions: redirigir la stdout del procés ps a l'entrada del pipe, redirigir la sortida del pipe a la stdin del procés sort i redirigir la stdout del procés sort al fitxer dades.txt

Examen 2017/18-1

Assignatura	Codi	Data	Hora inici
Sistemes operatius	05.566	10/01/2018	12:00

c) La reubicació dinàmica, ¿exigeix que tot el procés estigui carregat a memòria?

Fals. Entenem per reubicació el procés de traducció entre les adreces físiques i les lògiques. La reubicació dinàmica és aquella que es fa en temps d'execució. Tècniques com la memòria virtual permeten que un procés estigui carregat només parcialment a la memòria física. L'ús de reubicació estàtica en un sistema de memòria virtual el que ens diu és que cada partició (pàgina o segment) del procés sap en temps de compilació o càrrega inicial en quina posició de memòria física s'ha de carregar; i per tant tota nova càrrega d'una mateixa partició haurà d'anar en la mateixa posició.

d) ¿L'ús d'una operació d'E/S no instantània (per exemple, una lectura de disc) i no bloquejadora, exigeix que el perifèric involucrat enviï una interrupció un cop ha acabat l'operació d'E/S que se li ha demanat?

Cert. Un procés fa una crida de sistema per iniciar una operació d'E/S no bloquejadora. La rutina de servei pertinent inicia l'operació en el perifèric, però retorna el control al procés iniciador. Així aquest pot continuar la seva tasca, però no sap en quin moment pot considerar l'operació d'E/S finalitzada (per exemple, en quin moment les dades que es volen llegir ja estan disponibles). Per solucionar el problema, quan el perifèric acaba l'operació d'E/S envia una interrupció de finalització; la rutina de servei pertinent el que fa és enviar un senyal al procés que ha fet la petició de l'operació d'E/S, per tal d'indicar-li que l'operació d'E/S ha finalitzat.

Examen 2017/18-1

Assignatura	Codi	Data	Hora inici
Sistemes operatius	05.566	10/01/2018	12:00

2. Processos [2.5 punts]

Contesteu les següents preguntes:

a) Indiqueu un motiu que pugui provocar que un procés passi d'estat Run a estat Ready.

El motiu típic és l'expiració del quantum. Si el procés que està a Run expira el quantum que té assignat, el planificador fa que aquest procés passi a l'estat Ready i que un del processos a Ready passi a Run.

b) Indiqueu quin serà el resultat d'executar el següent programa (jerarquia de processos creada, informació mostrada per la sortida estàndar). Podeu assumir que cap crida al sistema retornarà error.

```
main()
{ int a,b;
  char s[80];

  if (fork() == 0) a=0; else a=1;
  if (fork() == 0) b=0; else b=1;

  sprintf(s, "a=%d b=%d\n", a, b);
  write(1, s, strlen(s));
}
```

La primera sentència crea un procés fill; el procés original (Or) tindrà a=1 i el fill creat (F1) a=0.
La segona sentència és executada pels dos processos amb el que cadascun crearà un fill (F2 i N) respectivament, on el procés N serà net del procés original.

Tots quatre processos imprimeixen els seus valors d'a i b:
Or: a=1 b=1
F1: a=0 b=1
F2. a=1 b=0
N: a=0 b=0

No podem saber en quin ordre apareixeran aquestes línies, dependrà del planificador de la cpu.

c) Escriviu un programa que cada cop que l'usuari premi un salt de línia, el programa indiqui quants processos estan a l'estat Ready al sistema mitjançant la comanda "ps –aux | grep –c S". No és necessari que implementeu el tractament d'errors a les crides al sistema. Cal implementar el programa utilitzant crides al sistema Unix, no és permès utilitzar rutines de biblioteca com ara system().

```
void printPsGrep()
{
  int fd[2];

  if (pipe(fd) < 0) error("pipe");

  switch (fork ())
    {
    case -1:
      error ("fork");
```

Examen 2017/18-1

Assignatura	Codi	Data	Hora inici
Sistemes operatius	05.566	10/01/2018	12:00

```
    case 0:
      close(1); dup(fd[1]); close(fd[0]); close(fd[1]);
      execlp ("ps", "ps", “-aux”, NULL);
      error ("execl");
    }

  switch (fork ())
    {
    case -1:
      error ("fork");

    case 0:
      close(0); dup(fd[0]); close(fd[0]); close(fd[1]);
      execlp ("grep", "grep", “-c”, “S”, NULL);
      error ("execl");
    }

  close(fd[0]); close(fd[1]);

  wait(NULL); wait(NULL);
}

int main (int argc, char *argv[])
{
  char c;
  int r;

  while ((r = read(0, &c, 1)) > 0)
    {
      if (c=='\n')
        printPsGrep();
    }

  if (r < 0)
    error("read");

  return 0;
}
```

Assignatura	Codi	Data	Hora inici
Sistemes operatius	05.566	10/01/2018	12:00

d) Escriviu un programa que creï un procés zombie/defunct.

```
main() {
 switch(fork()) { /* Crea un procés fill */
   case 0: exit(0); /* El fill mor */
    default:
             /* Mentre no invoquem wait(), el fill creat estarà zombie */
             ...
 }
}
```

Assignatura	Codi	Data	Hora inici
Sistemes operatius	05.566	10/01/2018	12:00

3. Memòria [2.5 punts]

Sigui un sistema de gestió de memòria basat en paginació sota demanda on les pàgines tenen una mida de 64KBytes, les adreces lògiques són de 20 bits i l'espai físic és de 512 KBytes.

Sobre aquest sistema es creen dos processos.

- Procés 1: el seu fitxer executable determina que el codi ocuparà dues pàgines, que les dades inicialitzades n'ocupen una, les no inicialitzades dues i la pila també dues.
- Procés 2: el seu fitxer executable determina que les àrees de codi i les dades inicialitzades ocuparan una pàgina cadascuna, que no existeixen dades no inicialitzades i que la pila ocuparà dues pàgines.

Es demana:

a) Estimeu la mida del fitxer executable corresponent al procés 1.

El fitxer executable d'un procés conté el codi i el valor inicial de les dades inicialitzades. En aquest cas, tenim dos pàgines de codi i una de dades inicialitzades per tant, aproximadament, l'executable ocuparà la mida corresponent a tres pàgines, és a dir, uns 192KB.

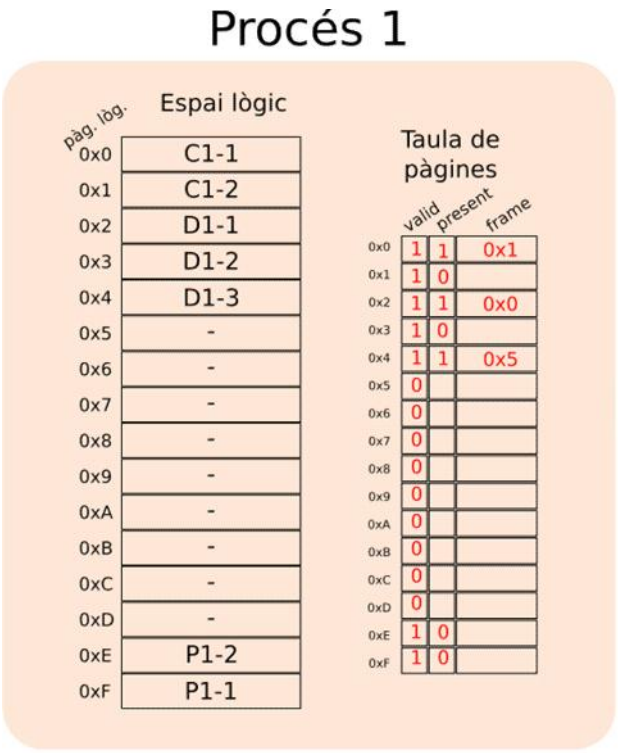
Per a una estimació més precisa caldria afegir la mida de les capçaleres de l'executable i caldria descomptar la fragmentació interna que pugui existir a la darrera pàgina de codi i de dades inicialitzades.

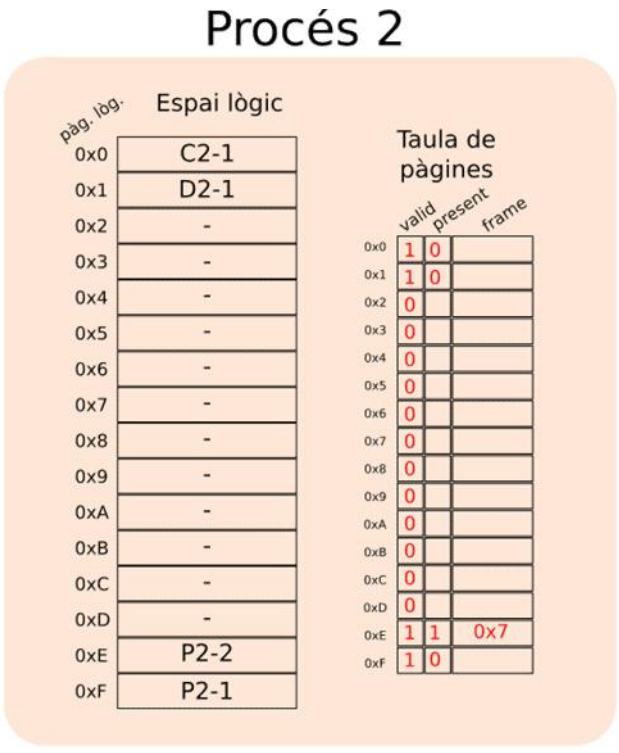
Assignatura	Codi	Data	Hora inici
Sistemes operatius	05.566	10/01/2018	12:00

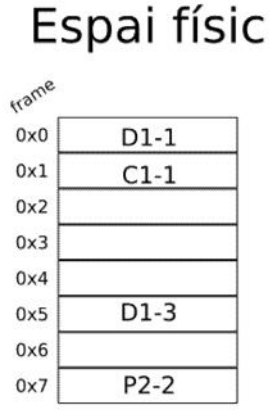
b) Suposant que les pàgines es carreguen a memòria física tal i com indica el diagrama següent, indiqueu quin serà el contingut de les taules de pàgines de tots dos processos (podeu contestar sobre el diagrama de l'enunciat).

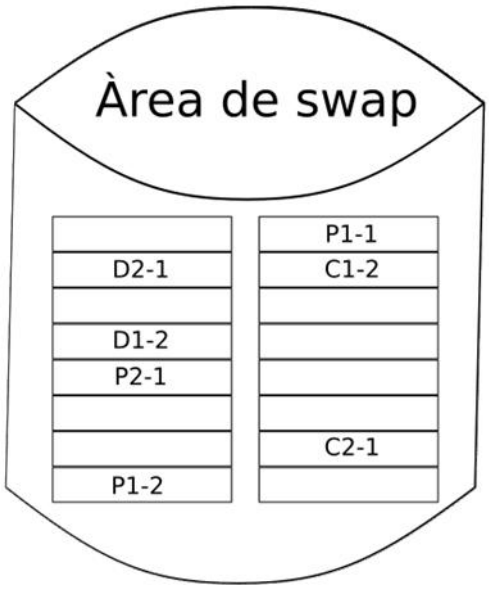
Examen 2017/18-1

Assignatura	Codi	Data	Hora inici
Sistemes operatius	05.566	10/01/2018	12:00









Assignatura	Codi	Data	Hora inici
Sistemes operatius	05.566	10/01/2018	12:00

c) Suposant que el procés en execució és el procés 1, indiqueu quines seran les adreces físiques corresponents a les següents adreces lògiques: 0x0A123 i 0x4B342. Variaria la resposta si el procés en execució fos el procés 2? En cas afirmatiu, indiqueu el motiu i com canviaria.

Primer cal descomposar les adreces lògiques en identificador de pàgina i desplaçament.

Com la mida de pàgina és de 64KB (2^16 bytes), calen 16 bits per a codificar el desplaçament dins de la pàgina, és a dir, quatre dígits hexadecimals. La resta de bits (4) seran l'identificador de pàgina. Per tant, el primer dígit hexadecimal ens indica l'identificador de pàgina lògica i la resta de dígits indiquen el desplaçament dins la pàgina.

Les traduccions serien:

@L	@F Procés 1	@F Procés 2
0x0A123	0x1A123	Fallada de pàgina
0x4B342	0x5B342	Excepció: Adreça invàlida

Són diferents a cada procés perquè cada procés té una taula de pàgines pròpia.

Examen 2017/18-1

Assignatura	Codi	Data	Hora inici
Sistemes operatius	05.566	10/01/2018	12:00

4. Concurrència [2.5 punts]

La PAC2 de l'assignatura versava sobre el problema de com controlar l'accés a un pont estret per part de conjunt de cotxes, simulats mitjançant fils d'execució, que volien creuar-ho en ambdues direccions. En tot moment, només es permet creuar cotxes en un sentit, havent d'esperar els cotxes de l'altra direcció fins que se'ls concedeixi el torn. L'espera s'implementava mitjançant semàfors amb l'objecte de no consumir recursos de CPU (espera passiva).

a) En la primera solució que ens proposaven, només podria creuar un cotxe al mateix temps. Aquesta solució es basava en la utilització de semàfors i l'espera passiva:

```
sem_t Bridge;
sem_init(Bridge, 0, 1);
```

```
CarCrossEast()
{
   sem_wait(Bridge);

   CrossingBridge();

   sem_signal(Bridge);
}
```

```
CarCrossWest()
{
   sem_wait(Bridge);

   CrossingBridge();

   sem_signal(Bridge);
}
```

Codi 1. Solució que només permet creuar un cotxe simultàniament.

Proposeu una solució basada en l'espera activa i que no utilitzi semàfors.
Indicar quins serien els inconvenients d'aquesta solució.

```
Shared int CarsBridge=0;
```

```
CarCrossEast()
{
   while(CarsBridge>0);
   CarsBridge++;

   CrossingBridge();

   CarsBridge--;
}
```

```
CarCrossWest()
{
   while(CarsBridge>0);
   CarsBridge++;

   CrossingBridge();

   CarsBridge--;
}
```

Codi 2. Solució que només permet creuar un cotxe simultàniament amb espera activa.

Els principals inconvenients d'aquesta solució són dues: l'espera activa i les condicions de carrera. La utilització d'espera activa implica que es desaprofiti la CPU a comprovar de forma contínua si una determinada condició es compleix o no.

En aquest cas, les condicions de carrera que es poden produir en modificar la variable compartida *CarsBridge*. En fer aquesta modificació fora d'una secció crítica, pot ocórrer que alguna de les modificacions es perdi, la qual cosa pot comportar un interbloqueig de tots els fils d'execució.

Assignatura	Codi	Data	Hora inici
Sistemes operatius	05.566	10/01/2018	12:00

b) En una de les variants, ens proposaven una solució que permet creuar múltiples cotxes, sempre que ho facin en el mateix sentit:

```
Shared int CarsEast=0, CarsWest=0;
sem_t Bridge, MutexEast, MutexWest;

sem_init(Bridge, 0, 1);
sem_init(MutexEast, 1);
sem_init(MutexWest, 1);
```

```
CarCrossEast()
{
   sem_wait(MutexEast);
   CarsEast++;
   if (CarsEast==1)
      sem_wait(Bridge);
   sem_signal(MutexEast);

   CrossingBridge();

   sem_wait(MutexEast);
   CarsEast--;
   if (CarsEast==0)
      sem_signal(Bridge);
   sem_signal(MutexEast);
}
```

```
CarCrossWest()
{
   sem_wait(MutexWest);
   CarsWest++;
   if (CarsWest==1)
      sem_wait(Bridge);
   sem_signal(MutexWest);

   CrossingBridge();

   sem_wait(MutexWest);
   CarsWest--;
   if (CarsWest==0)
      sem_signal(Bridge);
   sem_signal(MutexWest);
}
```

Codi 3. Solució que permet creua múltiples cotxes en el mateix sentit.

Després d'una inspecció del pont s'ha constatat que la seva integritat estructural pot perillar si ha de suportar el pes de més de 5 vehicles a la vegada. Per això, ens han demanat que proposem una solució amb el nostre simulador, que permeti controlar mitjançant semàfors que com a màxim només pot haver-hi 5 cotxes simultàniament al pont.

La solució es basa en utilitzar un semàfor (MaxCars) per limitar en nombre de cotxes que es permet passar el Pont. Aquest semàfor s'inicialitza al nombre de cotxes que poden passar simultàniament (5) i cada vegada que un cotxe vol creuar es decrementa el semàfor. Quan ja no queda instàncies disponibles el cotxe es bloqueja, esperant que surti un cotxe del pont.

```
Shared int CarsEast=0, CarsWest=0;
sem_t Bridge, MutexEast, MutexWest, MaxCars;

sem_init(Bridge, 0, 1);
sem_init(MutexEast, 0, 1);
sem_init(MutexWest, 0, 1);
sem_init(MaxCars, 0, 5);
```

Examen 2017/18-1

Assignatura	Codi	Data	Hora inici
Sistemes operatius	05.566	10/01/2018	12:00

```
CarCrossEast()
{
   sem_wait(MutexEast);
   CarsEast++;
   if (CarsEast==1)
      sem_wait(Bridge);
   sem_signal(MutexEast);
   sem_wait(MaxCars);

   CrossingBridge();

   sem_wait(MutexEast);
   CarsEast--;
   sem_signal(MaxCars);
   if (CarsEast==0)
      sem_signal(Bridge);
   sem_signal(MutexEast);
}
```

```
CarCrossWest()
{
   sem_wait(MutexWest);
   CarsWest++;
   if (CarsWest==1)
      sem_wait(Bridge);
   sem_signal(MutexWest);
   sem_wait(MaxCars);

   CrossingBridge();

   sem_wait(MutexWest);
   CarsWest--;
   sem_signal(MaxCars);
   if (CarsWest==0)
      sem_signal(Bridge);
   sem_signal(MutexWest);
}
```

Codi 4. Solució, mitjançant un semàfor, per controlar el nombre màxim de cotxes al pont.

Examen 2017/18-1

Assignatura	Codi	Data	Hora inici
Sistemes operatius	05.566	10/01/2018	12:00

Examen 2017/18-1

Assignatura	Codi	Data	Hora inici
Sistemes operatius	05.566	10/01/2018	12:00

Examen 2017/18-1

Assignatura	Codi	Data	Hora inici
Sistemes operatius	05.566	10/01/2018	12:00