Parcial Sistemes Operatius II

Avaluació parcial 2

14 de gener del 2022

La prova parcial es pot fer en 2h, duració prevista per a l'examen parcial presencial. El parcial té aquestes condicions i restriccions. A partir de la següent pàgina es mostra l'enunciat de la prova.

- La prova s'ha de respondre de forma individual.
- **Les respostes a les preguntes han de ser manuscrites, no s'acceptarà cap entrega escrita amb un editor de text.** Indiqueu el vostre nom a cadascuna de les pàgines que utilitzeu per respondre a les preguntes.
- Es podran utilitzar a tot estirar 4 pàgines per respondre a la prova parcial. En cas que les respostes excedeixin les 4 pàgines només s'avaluaran les 4 primeres pàgines.
- **És molt important comentar i/o raonar les respostes a les preguntes**. És suficient amb una resposta curta (3 o 4 frases) perquè es doni la resposta com a vàlida. No és necessari doncs estendre's en respondre. En avaluar es valorarà el fet que es pugui veure que s'ha entès el que ha donat com a resposta.
- Es permet fer anotacions a l'enunciat per indicar com modificar el codi per respondre a les preguntes que es fan als exercicis.
- L'entrega de la prova es farà a través del campus. **Teniu temps fins avui, 14 de gener, a les 24h per realitzar l'entrega.** Eviteu enviar-la per correu atès que en entregar-la via campus queda constància "oficial" que s'ha realitzat l'entrega.
- Per realitzar l'entrega es poden escanejar o fotografiar les pàgines manuscrites (així com aquelles pàgines de l'enunciat si hi heu fet modificacions al codi). Entregueu, preferentment, un únic document PDF. També podeu entregar fitxers independents escanejats/fotografiats en format PDF, JPG o PNG, comprimits en un ZIP.
- Assegureu-vos que el document escanejat/fotografiat es pot veure bé a la vostra pantalla. Un cop hagueu pujat el document al campus, es recomana que us el baixeu per assegurar-vos que és el que volíeu entregar. Es podrà pujar el document un nombre il·limitat de vegades fins que arribi la data i hora límit d'entrega.
- En cas que es detecti còpia es posarà un zero (0) a totes les persones implicades. Es podrà aplicar la normativa acadèmica general de la Universitat de Barcelona i l'inici d'un procés disciplinari.

Exercici 1 (espera activa, 2.5 punts, 0.5 punts per pregunta) Les funcions de bloqueig es caracteritzen pel fet que, quan hi ha múltiples fils esperant a una clau, no se sap quin dels fils agafarà la clau així que s'allibera. És a dir, no s'assegura (per defecte) que els fils entraran a la secció crítica en l'ordre en què arriben per agafar la clau. Es proposa un codi que ho assegura per a N fils, és a dir, que els fils entraran a la secció crítica en l'ordre en què hi arriben.

a) Observar que hi ha una variable `torn`, que és global, i una variable `torn_meu`, local a cada fil. Per a què serveixen aquestes dues variables? Per què la primera és global i la segona local?

b) Quina és la utilitat de la variable global `torn_actual`?

c) Observa que a la funció `unlock` s'executen les instruccions de les línies 15 i 16. Per a què serveixen? És a dir, què es vol aconseguir?

d) Fa falta que la instrucció de la línia 8 sigui atòmica? Raona la resposta. En cas que hagi de ser atòmica, indica un exemple que faci que el codi no funcioni correctament.

e) L'algorisme ha estat dissenyat per a N fils. Funcionarà correctament per a un nombre de fils inferior a N? I per a un nombre de fils superior a N? Raoneu la resposta.

```
 1 variables globals:
 2     boolean flag[N] = {false, false, ..., false}; flag[0] = true;
 3     int torn_actual;
 4     int torn = 0;
 5
 6 lock:
 7     int torn_meu;
 8     <torn_meu = torn; torn++;>    // instrucció atòmica
 9     torn_meu = torn_meu % N;
10     while (!flag[torn_meu]) {};
11     torn_actual = torn_meu;
12     return;
13
14 unlock:
15     flag[torn_actual] = false;
16     flag[(torn_actual + 1) % N] = true;
17     return;
```

Exercici 2 (semàfors, 3.75 punts, 0.75 punts per pregunta) Un refugi de muntanya té dutxes unisex. Pot ser utilitzat tant per homes i dones però no hi pot haver homes i dones al mateix temps a la dutxa. Es proposa una solució al problema fent servir semàfors binaris (vegeu següent pàgina).

a) Per a què serveixen cadascun dels semàfors `mutexD`, `mutexH` i `dutxa`? Comenteu la vostra resposta per a cadascun dels semàfors en el context d'aquest codi.

b) Creieu necessari fer servir semàfors binaris? Què passaria si `mutexD` o `mutexH` fossin un semàfor general? Què passaria si `dutxa` fos un semàfor general? Raoneu la resposta.

c) El codi assegura que no hi pot haver homes i dones al mateix temps a la dutxa. Suposeu que hi ha un home dutxant-se i que tres dones volen entrar a dutxar-se. A quin dels semàfors s'esperaran cadascuna de les dones? Raoneu la resposta.

d) Què passa si mentre les dones s'esperen arriba un nou home. Podrà entrar a la dutxa? Raoneu la vostra resposta.

e) (Difícil) Suposeu que la dutxa té una capacitat màxima per a quatre persones. Comenteu i indiqueu com modificaríeu el codi perquè a tot estirar hi pugui haver quatre persones (del mateix sexe!) a la dutxa. Podeu indicar les modificacions en el codi de l'enunciat i comenteu per què soluciona aquesta restricció. No està permès fer servir espera activa o monitors per resoldre la pregunta.

```
1 variables globals:
2   int dones = 0, homes = 0;
3   sem_t mutexD (= 1), mutexH (= 1), dutxa (= 1);
4
5 dones:
6   sem_wait(&mutexD);
7   dones = dones + 1;
8   if (dones == 1) sem_wait(&dutxa);
9   sem_post(&mutexD);
10  // entrar a la dutxa i dutxar-se
11  sem_wait(&mutexD);
12  dones = dones - 1;
13  if (dones == 0) sem_post(&dutxa);
14  sem_post(&mutexD);
15
16 homes:
17  sem_wait(&mutexH);
18  homes = homes + 1;
19  if (homes == 1) sem_wait(&dutxa);
20  sem_post(&mutexH);
21  // entrar a la dutxa i dutxar-se
22  sem_wait(&mutexH);
23  homes = homes - 1;
24  if (homes == 0) sem_post(&dutxa);
25  sem_post(&mutexH);
```

Exercici 3 (monitors, 3.75 punts, 0.75 punts per pregunta) El problema de "la barberia" és un dels problemes clàssics de la sincronització de fils en sistemes operatius. Es presenta a continuació una versió simplificada del problema original. La barberia és una botiga en què hi ha una sala d'espera amb infinites cadires. A la botiga hi ha la sala del barber on el client és atès. Les condicions de funcionament de la barberia són les següents: a) Només hi ha un barber. Si no hi ha cap client, el barber se'n va a dormir (línies 12-13 del codi), b) Quan una persona es vol tallar el cabell crida a la funció client per entrar a la botiga. Si el barber està ocupat, el client espera (línies 23-24), c) Se suposa que el barber arriba abans a la botiga que els clients.

Comenteu i raoneu les respostes a les preguntes en el context d'aquest codi:

a) Fa falta protegir amb la mateixa clau `barberia` tant la part del barber com la del client? Es poden protegir aquestes dues parts amb claus diferents? Raoneu la resposta.

b) Observar que a la línia 12 es protegeix el `wait` amb un `if` en comptes d'un `while`. Raoneu per què no fa falta un `while`.

c) De forma similar, observar que a la línia 23 es protegeix el `wait` amb un `while` en comptes d'un `if`. Es pot produir algun problema si es posa un `if`? En cas afirmatiu, comenteu com i per què es produeix el problema

d) Dintre del context d'aquest exercici, quan es fa la crida a la funció `signal` de la línia 14 i la de la línia 25? Quin és el propòsit de cada crida? Hi ha d'haver algú esperant al `wait` corresponent perquè l'algorisme funcioni? (Per respondre la pregunta, feu servir les paraules "barber" i "client" en comptes del terme genèric "fil").

e) (Difícil) Suposeu que a la botiga hi ha múltiples barbers en comptes d'un únic barber. Cada barber podrà atendre, doncs, a un client diferent. Com modificaríeu el codi per tal d'adaptar-ho a aquest cas? Podeu indicar les modificacions en el codi de l'enunciat i comenteu la vostra proposta. No està permès fer servir espera activa o semàfors per respondre la pregunta.

```
1 variables globals:
2     int clients = 0;         // Nombre de clients
3     boolean ocupat = false;  // Permet saber si el barber esta ocupat
4
5     mutex barberia;
6     cond cond_barber, cond_clients;
7
8 barber: // Hi ha nomes un unic barber. El barber obre la botiga.
9     while (true) {
10        lock(&barberia);
11        ocupat = false;
12        if (clients == 0) // Si no hi ha clients, el barber s'adorm
13            wait(&cond_barber, &barberia);
14        signal(&cond_clients);
15        clients--;
16        unlock(&barberia);
17        tallar_cabell_client();
18    }
19
20 client: // Els clients criden la funcio quan es volen tallar el cabell
21    lock(&barberia);
22    clients++;
23    while (ocupat) // Si el barber esta ocupat, el client s'adorm
24        wait(&cond_clients, &barberia);
25    signal(&cond_barber);
26    ocupat = true;
27    unlock(&barberia);
28    anar_a_sala_per_tallar_cabell();
29    return; // El client surt de la barberia
```