ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ
ΕΡΓΑΣΤΗΡΙΟ ΥΠΟΛΟΓΙΣΤΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ
http://www.cslab.ece.ntua.gr

Λειτουργικά Συστήματα

7ο εξάμηνο, Ακαδημαϊκή περίοδος 2009-2010

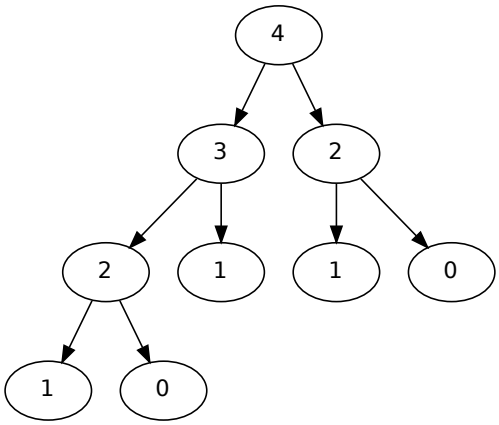
Σεπτέμβριος 2010 – Λύσεις Θεμάτων

Θέμα 1

α.

```
int fibfork(int n){
    int ret1, ret2;

    if (n == 0 || n == 1)
        return n; /* end recursion */

    if (fork() == 0)
        exit(fibfork(n-1)); /* first child */

    if (fork() == 0)
        exit(fibfork(n-2)); /* second child */

    /* parent */
    xwait(&ret1);
    xwait(&ret2);
    return ret1 + ret2;
}
```

β.



γ. Με κριτήριο το μέγιστο χρησιμοποιούμενο χώρο, η τακτική (i) είναι καλύτερη διότι "ανοίγει" το δέντρο διεργασιών "κατά βάθος" και όχι "κατά πλάτος". Με την τακτική (i) ο μέγιστος αριθμός διεργασιών που μπορεί να υπάρξουν είναι ίσος με το ύψος του δέντρου. Αντίθετα, στην περίπτωση (ii) ο μέγιστος αριθμός διεργασιών είναι μεγαλύτερος και κατά συνέπεια χρησιμοποιείται περισσότερος χώρος.

δ. Μπορούν να χρησιμοποιηθούν σωληνώσεις (`pipe()`) ή οποιαδήποτε άλλη μορφή IPC που παρέχει το σύστημα. Πριν τη δημιουργία κάθε παιδιού, θα πρέπει να δημιουργηθεί η σωλήνωση και ο κατάλληλος περιγραφητής αρχείου για το άκρο εγγραφής να περαστεί στο παιδί. Το παιδί θα περάσει το αποτέλεσμα τον γονιό γράφοντας έναν ακέραιο στον περιγραφητή αρχείου.

Θέμα 2

α.

```
barber = Semaphore(0);
client = Sempahore(0);
```

```
void barber(){
    while (1){
        client.wait();
        barber.signal();
        cut_hair();
    }
}
```

```
void customer(){
    client.signal();
    barber.wait();
    get_haircut();
}
```

β.

```
barber  = Semaphore(0);
client  = Sempahore(0);
lock    = Semaphore(1); // mutex for waiting shared variable
waiting = 0;

// barber remains the same

void customer(){
    lock.wait()
    if (waiting <= N){ // increase counter
        waiting += 1;
        lock.signal()
    } else {
        lock.signal(); // release lock
        leave();
    }

    client.signal();
    barber.wait();
    get_haircut();

    lock.wait();
    waiting -= 1;
    lock.signal();
}
```

γ. Η συνάρτηση μπορεί να οδηγήσει σε deadlock. Αν έχουμε δύο λογιαρασμούς Α και Β, τότε ταυτόχρονη κλήση των `money_xfer(A,B)` και `money_xfer(B,A)` μπορεί να οδηγήσει σε κυκλική αναμονή και deadlock.

Θέμα 3

α.

1. Η διεργασία προσπαθεί να προσπελάσει διεύθυνση εικονικής μνήμης που δεν της ανήκει. Σε αυτή την περίπτωση το ΛΣ πρέπει να στείλει κατάλληλο σήμα στη διεργασία για να την τερματίσει.

2. Η διεργασία προσπαθεί να προσπελάσει διεύθυνση εικονικής μνήμης που της ανήκει, αλλά έχει μεταφερθεί στο δίσκο. Σε αυτή την περίπτωση το ΛΣ πρέπει να φέρει την αντίστοιχη σελίδα από το δίσκο και να φτιάξει τις κατάλληλες κατάλληλες απεικονίσεις (mappings), ώστε η σελίδα να είναι διαθέσιμη στην εργασία. Όταν γίνει αυτό η διεργασία τίθεται σε κατάσταση "ΕΤΟΙΜΗ".

β.

i. Ναι. Αν, για παράδειγμα, η διεργασία δεσμεύσει μεγάλο χώρο μνήμης και τον προσπελάσει ακουλουθιακά (όλες τις διευθύνσεις από την αρχή έως το τέλος). Τότε κάθε πρόσβαση σε νέα σελίδα θα δημιουργεί σφάλμα σελίδας.

Ο αριθμός των σφαλμάτων είναι: $\frac{\text{μέγεθος δεσμευμένης μνήμης}}{\text{μέγεθος σελίδας}}$

Άρα, αύξηση του μεγέθους σελίδας οδηγεί σε μείωση των σφαλμάτων σελίδας.

ii. Ναι. Έστω, για παράδειγμα, ότι η διεργασία χρησιμοποιεί χώρο μνήμης ίσο με δύο φορές το μέγεθος της φυσικής μνήμης. Έστω, επίσης, ότι η διεργασία κάνει μεγάλο αριθμό προσβάσεων στη μνήμη και ότι όλες οι προσβάσεις της είναι στην αρχή των σελίδων. Σε αυτή την περίπτωση θα υπάρχουν σφάλματα σελίδας επειδή δεν χωράνε όλα τα δεδομένα στη μνήμη. Αν μειωθεί το μέγεθος της σελίδας στο μισό, τότε οι προσβάσεις θα γίνονται μόνο στις μισές σελίδες. Σε αυτή την περίπτωση όλα τα δεδομένα που χρειάζεται η εφαρμογή χωράνε στη μνήμη και κατά συνέπεια μειώνονται τα σφάλματα σελίδας.

γ. Μπορεί να πραγματοποιηθεί εύκολα ο διαχωρισμός αριθμού σελίδας (page number) και μετατόπισης (offset).

δ. Ένα πλεονέκτημα είναι χρειάζεται μικρότερος χώρος για την διατήρηση των πινάκων σελίδων. Ένα μειονέκτημα είναι ότι αυξάνεται το κόστος διάσχισης των πινάκων σελίδων (π.χ. σε περίπτωση TLB miss).

Θέμα 4

α. Το πρόγραμμα πραγματοποιεί πολλά writes μεγέθους ενός byte. Είναι πιθανό, συνεπώς, το πρόβλημα να οφείλεται στο κόστος κάθε κλήσης συστήματος. Για να μειωθεί αυτό το κόστος, ο χρήστης μπορεί να κάνει εγγραφές μεγαλύτερου μεγέθους (χρησιμοποιώντας π.χ. buffering).

β. Θα μπορούσε το σύστημα κρυφής μνήμης ΣΑ να φέρνει στη μνήμη μεγάλα τμήματα του αρχείου πριν τα ζητήσει ο χρήστης (prefetching).

γ. Ο κωδικός λάθους ELOOP προσπαθεί να αποτρέψει την πιθανότητα βρόχων (loops). Για παράδειγμα:

```
$ ln -s A A # συμβολικό link από το A στον εαυτό του
$ cat A
cat: A: Too many levels of symbolic links
$ strace cat A
...
open("A", O_RDONLY) = -1 ELOOP (Too many levels of symbolic links)
...
```