Η/Υ 445: Λειτουργικά Συστήματα
Άνοιξη 2002 - Παναγιώτα Φατούρου
2ο Σετ Προγραμματιστικών Ασκήσεων

Ημερομηνία Παράδοσης: Πέμπτη, 11/4/02

Σχόλια: Θα πρέπει να στείλετε με e-mail την άσκησή σας στη διεύθυνση cs445@cs.uoi.gr μέχρι τις 12:59 της Πέμπτης 11/4. Επίσης, θα πρέπει να κρατήσετε την άσκησή σας σε κάποιο directory στην περιοχή σας χωρίς να αλλάξετε την ημερομηνία της μέχρι να γίνει η εξέτασή της. Η ημερομηνία τελευταίας εγγραφής της άσκησής σας θα πρέπει να είναι είτε κάποια προηγούμενη ή η αναγραφόμενη ημερομηνία και ώρα παράδοσης. Μην ετοιμάζετε την άσκησή σας την τελευταία στιγμή. Η ημερομηνία και ώρα παράδοσης προβλέπει πιθανά προβλήματα στο σύστημα και η άσκησή σας θα πρέπει να παραδοθεί μέχρι αυτή την ημερομηνία ανεξάρτητα από τέτοιου είδους προβλήματα. Σας υπενθυμίζω πως ο κώδικάς σας θα πρέπει να είναι γραμμένος σε C, να τρέχει στα μηχανήματα του εργαστηρίου του Τμήματος, να είναι ευανάγνωστος και να έχει σχολιαστεί κατάλληλα.

Ο σκοπός της άσκησης είναι να εξοικειωθείτε με κάποιους τρόπους ανταλλαγής δεδομένων μεταξύ διεργασιών στο UNIX, υλοποιώντας μια παραλλαγή του κλασικού προβλήματος διαδιεργασιακής επικοινωνίας παραγωγών-καταναλωτών.

Το πρόγραμμά σας θα πρέπει να εκκινεί NP = 2 διεργασίες παραγωγούς και NC = 3 διεργασίες καταναλωτές. Κάθε παραγωγός και καταναλωτής θα είναι ξεχωριστή ατέρμονη διεργασία (process) στο UNIX. Επίσης, θα πρέπει να υπάρχει μια ακόμη διεργασία συντονιστής. Όλες οι διεργασίες θα δημιουργούνται χρησιμοποιώντας την fork(2).

Στην άσκηση αυτή οι διεργασίες επιτρέπεται να επικοινωνούν μόνο με ανταλλαγή μηνυμάτων μέσω σωληνώσεων (pipe(2)).

Ο συντονιστής περιμένει ανά πάσα στιγμή δεδομένα από κάποιο παραγωγό (δες select(2)). Αν υπάρχουν διαθέσιμα δεδομένα μπορεί να τα παρέχει σε κάποιον από τους καταναλωτές. Ο ρόλος του συντονιστή είναι επομένως να μεσολαβεί ώστε δεδομένα οποιουδήποτε παραγωγού να φθάνουν σε οποιονδήποτε καταναλωτή. Ο συντονιστής χρησιμοποιεί εσωτερικό buffer, μεγέθους 8192 bytes, ενιαίο για όλους τους παραγωγούς και καταναλωτές. Αν ο buffer γεμίσει, ο συντονιστής θα πρέπει να αποτρέπει τους παραγωγούς από το να παράγουν επιπλέον δεδομένα έως ότου κάποια από τα δεδομένα στον buffer διατεθούν στους καταναλωτές (ώστε να δημιουργηθεί ο απαραίτητος χώρος στον buffer). Αντίστοιχα, αν ο buffer είναι άδειος, ο συντονιστής θα πρέπει να αποτρέπει τους παραγωγούς από το να ζητούν δεδομένα για κατανάλωση (χρειάζεστε επομένως και μερικά pipes που θα λειτουργούν σαν σημαφόροι).

Ο συντονιστής θα πρέπει να εξετάζει πρώτα τις απαιτήσεις των παραγωγών (με τυχαία σειρά) και αν είναι δυνατό να τις ικανοποιεί. Στη συνέχεια θα πρέπει να εξετάζει τις απαιτήσεις των καταναλωτών, ξεκινώντας κάθε φορά από κάποιον τυχαίο από αυτούς (μπορείτε να παράγεται τυχαίους αριθμούς είτε χρησιμοποιώντας τα tv_usec που παρέχει η gettimeofday(3C), ή χρησιμοποιώντας τις rand(3C), srand(3C)).

Οι παραγωγοί και οι καταναλωτές δουλεύουν ανεξάρτητα μεταξύ τους και χρησιμοποιούν μικρές (~0.001 sec) τυχαίες καθυστερήσεις (usleep(3)), πριν παράγουν ή ζητήσουν για να καταναλώσουν δεδομένα, κάποιο τυχαίο αριθμό από bytes (μεταξύ 1 και 10). Όλες οι ανταλλαγές bytes γίνονται μόνο με τον συντονιστή, και όχι απευθείας μεταξύ παραγωγών και καταναλωτών.

Το πρόγραμμα σας θα πρέπει να ζητάει από τον χρήστη να δίνει τον αριθμό των sec που θα τρέξει. Ο συντονιστής θα πρέπει επομένως επαναληπτικά να καλεί την gettimeofday (δες gettimeofday(3C)) και να ελέγχει αν το πρόγραμμα έχει τρέξει περισσότερο από τον ζητούμενο χρόνο, οπότε και θα πρέπει να τερματίζει. Όταν ο συντονιστής τερματίσει θα πρέπει να φροντίζει και για τον τερματισμό όλων των άλλων διεργασιών (χρησιμοποιώντας τις signal(3) και kill(2)).

Κάθε διεργασία που τερματίζει θα πρέπει να αναφέρει το checksum (το άθροισμα των bytes) των δεδομένων (και όχι τυχόν μηνυμάτων ελέγχου) που αντάλλαξε με οποιαδήποτε άλλη διεργασία (οι παραγωγοί και οι καταναλωτές με τον συντονιστή και ο συντονιστής με τους παραγωγούς και καταναλωτές) και αυτά θα πρέπει να συμφωνούν).

Υποδείξεις

Χρησιμοποιείστε τα manuals του UNIX για να βρείτε πως θα υλοποιήσετε δύσκολα σημεία της άσκησής σας. Θα πρέπει να είστε σε θέση να δικαιολογήσετε τυχόν αυθαίρετες αποφάσεις που θα πάρετε και να περιγράψετε τον τρόπο λύσης που θα ακολουθήσετε.

Όσο το πρόγραμμα δεν είναι έτοιμο ώστε να τερματίζει, μην παραλείπετε να “σκοτώνετε” τις διεργασίες που δημιουργεί με την εντολή kill, αφού πρώτα δείτε ποιες είναι με την εντολή ps, γιατί διαφορετικά γρήγορα θα γεμίσετε το σύστημα με διεργασίες zombie.

```
ERROR: syntaxerror
OFFENDING COMMAND: --%ztokenexec_continue--

STACK:

false
```