Εργασία 3 – Σύστημα κράτησης εισιτηρίων

Υλοποιήστε ένα σύστημα κράτησης αεροπορικών εισιτηρίων το οποίο εκτελείται τοπικά σε έναν υπολογιστή. Ένας server (εξυπηρετητής) δέχεται αιτήματα από πράκτορες για πρόσβαση σε κοινόχρηστη μνήμη που περιλαμβάνει πληροφορίες πτήσεων και διαθέσιμων θέσεων. Επίσης δέχεται ενημερώσεις για τις κρατήσεις που κάνουν οι διάφοροι πράκτορες. Κάθε πράκτορας, εφόσον το αίτημά του γίνει δεκτό, αποκτά πρόσβαση στην κοινόχρηστη μνήμη την οποία μπορεί να διαβάσει για να ενημερωθεί για διαθέσιμες πτήσεις ή να γράψει σε περίπτωση κράτησης εισιτηρίων. Μπορεί να υπάρχουν ταυτόχρονα πολλοί ενεργοί πράκτορες αλλά η ανάγνωση από / εγγραφή στην κοινόχρηστη μνήμη πρέπει να γίνεται υπό αμοιβαίο αποκλεισμό με την χρήση σηματοφόρων.

Server

Το πρόγραμμα παίρνει ως ορίσματα το μέγιστο αριθμό πρακτόρων που μπορεί να εξυπηρετήσει, και το όνομα ενός αρχείου που περιέχει πληροφορίες πτήσεων. Δημιουργεί την κοινόχρηστη μνήμη και αποθηκεύει σε αυτή (σε δυαδική μορφή) τις πληροφορίες που διαβάζει από το αρχείο. Δημιουργεί επίσης έναν ή περισσότερους σηματοφόρους για τον συγχρονισμό ανάγνωσης/εγγραφής στην κοινόχρηστη μνήμη.

Ο server διατηρεί ένα file descriptor από τον οποίο διαβάζει τα αιτήματα των πρακτόρων (κάθε αίτημα περιέχει το process id του πράκτορα). Εφόσον δεν είναι συνδεδεμένος ο μέγιστος αριθμός πρακτόρων, το αίτημα γίνεται δεκτό, ο server στέλνει στον πράκτορα ότι πληροφορία χρειάζεται ώστε να μπορεί να προσπελάσει την κοινόχρηστη μνήμη, και εκτυπώνει μήνυμα με το process id του πράκτορα. Επιπλέον, για κάθε πράκτορα που γίνεται δεκτός, ο server δημιουργεί/διατηρεί ξεχωριστό file descriptor από τον οποίο διαβάζει πληροφορία για κάθε κράτηση που πραγματοποιείται, και εκτυπώνει μήνυμα με το process id του πράκτορα και τον αριθμό θέσεων της κράτησης.

Παράλληλα, ο server παρακολουθεί τη συμβατική του είσοδο. Αν διαβάσει το μήνυμα “Q” ή ανιχνεύσει end-of-input, εκτυπώνει στην οθόνη το συνολικό αριθμό θέσεων που κρατήθηκαν από κάθε πράκτορα που είναι ακόμη ενεργός, ανανεώνει τα περιεχόμενα του αρχείου (ανάλογα με τις κρατήσεις που έχουν πραγματοποιηθεί), κλείνει όλους τους file descriptors και μόνιμες οντότητες IPC που δημιούργησε, και τερματίζει.

Το πρόγραμμα πρέπει να παρακολουθεί όλους τους file descriptors χωρίς ενεργή αναμονή (μέσω select ή poll).

Πράκτορας

Κάθε διεργασία-πράκτορας εκτελεί το ίδιο πρόγραμμα το οποίο δέχεται ως όρισμα τα στοιχεία επικοινωνίας του server. Κατά την εκκίνηση στέλνει ένα αίτημα στο server με το process id του. Εάν τον αίτημα δε γίνει δεκτό, τερματίζει. Διαφορετικά, δέχεται επαναληπτικά από το πληκτρολόγιο αιτήσεις από το χρήστη, οι οποίες μπορεί να είναι για εύρεση διαθέσιμων πτήσεων (FIND), για κράτηση εισιτηρίων (RESERVE) ή για τερματισμό (EXIT).

Μια αίτηση για εύρεση πτήσεων είναι της μορφής FIND SRC DEST NUM όπου SRC και DEST τα αεροδρόμια αναχώρησης και άφιξης αντίστοιχα και NUM το πλήθος εισιτηρίων που ζητούνται. Το πρόγραμμα αναζητά στην κοινόχρηστη μνήμη πτήσεις που ικανοποιούν τις απαιτήσεις και τις εμφανίζει στην οθόνη.

Μια αίτηση για κράτηση είναι της μορφής RESERVE SRC DEST AIRLINE NUM όπου AIRLINE είναι η επιθυμητή αεροπορική εταιρία και τα υπόλοιπα όπως περιγράφεται παραπάνω. Το πρόγραμμα αναζητά τη συγκεκριμένη πτήση, κι εφόσον υπάρχουν ακόμη αρκετές θέσεις, αφαιρεί το NUM από το πλήθος διαθέσιμων θέσεων, στέλνει το NUM στον server, και εκτυπώνει μήνυμα επιτυχίας στην οθόνη. Αν δε βρεθεί η ζητούμενη πτήση ή βρεθεί αλλά δεν υπάρχουν αρκετές θέσεις, εκτυπώνει μήνυμα αποτυχίας.

Μορφή αρχείου με πληροφορίες πτήσεων

Το αρχείο δίνεται ως κείμενο ASCII όπου κάθε γραμμή τα στοιχεία μιας πτήσης, και για την ακρίβεια τις παρακάτω πληροφορίες χωρισμένες με ένα κενό: κωδικός αεροπορικής εταιρίας (μέχρι 3 χαρακτήρες), κωδικός αεροδρομίου αναχώρησης (μέχρι 4 χαρακτήρες), κωδικός αεροδρομίου άφιξης (μέχρι 4 χαρακτήρες), πλήθος ενδιάμεσων στάσεων (ακέραιος), πλήθος θέσεων στο αεροσκάφος (ακέραιος).

Γενικές παρατηρήσεις και απαιτήσεις

Οι file descriptors που παρακολουθεί ο server μπορεί να είναι named pipes ή unix sockets (τύπου datagram ή stream). Η επιλογή είναι δική σας.

Κάθε μήνυμα που εκτυπώνουν τα προγράμματα στο stdout ή stderr πρέπει να συνοδεύεται από χαρακτήρα αλλαγής γραμμής. Τα διαγνωστικά μηνύματα λάθους πρέπει να εκτυπώνονται αποκλειστικά στο stderr.

Με τα αρχεία σας πρέπει να συμπεριλάβετε κατάλληλο Makefile. Τα ονόματα των εκτελέσιμων που θα παραχθούν πρέπει υποχρεωτικά να είναι server και agent για τον εξυπηρετητή και πράκτορα αντίστοιχα.

Μετά το τέλος της εκτέλεσης δεν πρέπει να παραμένουν στο σύστημα μόνιμες οντότητες πέραν του αρχείου με πληροφορίες πτήσης.

Όπως πάντα, απαγορεύεται η χρήση καθολικών μεταβλητών, goto, gets. Δώστε περιγραφικά ονόματα σε συναρτήσεις και μεταβλητές, σχολιάστε τις συναρτήσεις και όσα τμήματα του κώδικα δεν είναι ξεκάθαρα, και γράψτε ευανάγνωστα με σωστή στοίχιση. Κώδικας που «δεν διαβάζεται» θα απορριφθεί, χωρίς εξέταση.

Συζήτηση/επεξήγηση εργασίας: **Πέμπτη 19/4/2018**, στην ώρα του μαθήματος Προθεσμία παράδοσης εργασίας: **Κυριακή 6/5/2018, 23:59** Οδηγίες παράδοσης εργασίας: στη σελίδα του μαθήματος.

Βασικά στάδια ανάπτυξης του κώδικα

Στάδιο 0

Αποφασίστε αν ο server και οι πράκτορες θα επικοινωνούν μέσω pipes ή sockets (τύπου datagram ή stream). Γράψτε δύο μικρά προγράμματα, ένα για ενδεικτικό server κι ένα για ενδεικτικό πράκτορα που επικοινωνούν μεταξύ τους με τον τρόπο που αποφασίσατε. Ο πράκτορας θα πρέπει να στέλνει στο server το pid του, θα λαμβάνει από το server ένα αριθμό και μετά να στέλνει στο server μια ακολουθία αριθμών επαναληπτικά, με καθυστέρηση ενός δευτερολέπτου ανάμεσα σε αποστολές. Ο server, αφού λάβει το pid ενός πράκτορα, θα στέλνει στον πράκτορα ένα αριθμό και μετά θα λαμβάνει από αυτόν μια ακολουθία αριθμών επαναληπτικά. Προχωρήστε στο επόμενο στάδιο ΜΟΝΟ εφόσον λειτουργεί σωστά η επικοινωνία.

Στάδιο 1

Ξεκινήστε με το πρόγραμμα που θα εκτελέσει ο server. Δημιουργήστε την κοινόχρηστη μνήμη και αποθηκεύστε σε αυτή τα περιεχόμενα του αρχείου. Βεβαιωθείτε ότι τα δεδομένα είναι γραμμένα σωστά. Επίσης γράψτε τον κώδικα που αποθηκεύει τις αλλαγές που έγιναν στα δεδομένα στην μνήμη πίσω στο αρχείο, και κάντε απλές δοκιμές αλλάζοντας τα δεδομένα μέσα από το πρόγραμμα του server.

Στάδιο 2

Ακολουθώντας τις ιδέες σας από το στάδιο 0, προσθέστε κώδικα στο πρόγραμμα του server που να υλοποιεί την επικοινωνία με πολλαπλούς πράκτορες, ανιχνεύοντας πότε κάποιος από αυτούς έχει στείλει πληροφορία στο server.

Γράψτε το πρόγραμμα που θα εκτελέσει κάθε πράκτορας. Προς το παρόν μην ασχοληθείτε με σηματοφόρους και μην γράφετε στην κοινόχρηστη μνήμη (μόνο να διαβάζετε).

Βεβαιωθείτε ότι το polling λειτουργεί σωστά κι ότι ο server μπορεί να λαμβάνει πληροφορίες από τους πράκτορες. Βεβαιωθείτε ότι πολλαπλοί πράκτορες μπορούν να διαβάζουν από την κοινόχρηστη μνήμη.

Στάδιο 3

Υλοποιήστε αμοιβαίο αποκλεισμό των διεργασιών που προσπελάζουν την κοινόχρηστη μνήμη με χρήση ενός ή περισσότερων σηματοφόρων.

Στάδιο 4

Υλοποιήστε την καταγραφή του συνολικού πλήθους κρατήσεων για κάθε πράκτορα.