ΛΕΙΤΟΥΡΓΙΚΑ ΣΥΣΤΗΜΑΤΑ

Εαρινό Εξάμηνο 2018-2019

1η Προγραμματιστική Εργασία

Δανοπούλου Έμυ [3170033]

Μπαλή Νίκη [3170114]

Χαβιατζή Ελένη [3170172]

Header file

Στο header file p3170033-p3170114-p3170172-res1.h αρχικοποιήσαμε τις σταθερές που ζητούνται, δηλώσαμε τις μεταβλητές balance, cost, waiting_time, total_waiting_time, double total_service_time, seedp και tid, δηλώσαμε 5 mutexes (για τους τηλεφωνητές, τις συναλλαγές με την τράπεζα, τις θέσεις, την οθόνη, και τους χρόνους που θα υπολογίσουμε), δημιουργήσαμε έναν πίνακα Seats που θα περιέχει τις θέσεις του θεάτρου και έναν πίνακα Return 5 στοιχείων (αφού 5 είναι ο μέγιστος αριθμός εισιτηρίων που μπορεί να ζητήσει ο πελάτης) τον οποίο θα χρησιμοποιήσουμε για να επιστρέφουμε τις θέσεις σε περίπτωση ακύρωσης της συναλλαγής. Επίσης δηλώσαμε ένα struct με όνομα timespec το οποίο θα χρησιμοποιηθεί για την συνάρτηση clock_gettime.

Συνάρτηση main

Ξεκινάμε ελέγχοντας πως έχουμε 3 ορίσματα όταν τρέχουμε το πρόγραμμα. Το πρώτο όρισμα είναι προφανώς το αρχείο a.out που θα δημιουργηθεί, το δεύτερο είναι ο αριθμός των πελατών (και άρα το πλήθος των νημάτων που θα φτιάξουμε) και το τρίτο είναι ο σπόρος για τη σειρά ψευδοτυχαίων αριθμών που θα δημιουργηθεί.

Έπειτα καλούμε τη malloc για να διαθέσουμε τον απαραίτητο αριθμό bytes για τα νήματά μας και ελέγχουμε για τυχόν σφάλμα.

Στη συνέχεια αρχικοποιούμε τα mutexes με την pthread_mutex_init και ταυτόχρονα ελέγχουμε για τυχόν σφάλματα.

Μετά τρέχουμε ένα loop για όσους πελάτες έχουμε, και σε κάθε επανάληψη δημιουργούμε το αντίστοιχο νήμα με την pthread_create και ορίσματα (&threads[customerCount], NULL, Client,tid), όπου threads ο πίνακας νημάτων που χρησιμοποιούμε, customerCount ο τρέχον αριθμός του πελάτη, Client η συνάρτηση που θα εκτελείται σε κάθε νήμα και tid ένας pointer που δείχνει στον τρέχον αριθμό του πελάτη. Επίσης ελέγχουμε για τυχόν σφάλμα.

Το κύριο νήμα περιμένει μέχρι τα υπόλοιπα νήματα που δημιουργήθηκαν να τελειώσουν τις ενέργειες τους, με τη συνάρτηση pthread_join.

Εκτυπώνουμε το πλάνο θέσεων, τα συνολικά έσοδα από τις πωλήσεις, το μέσο χρόνο αναμονής των πελατών (από τη στιγμή που εμφανίζεται ο πελάτης, μέχρι να μιλήσει με τον τηλεφωνητή) και το μέσο χρόνο εξυπηρέτησης των πελατών (από τη στιγμή που εμφανίζεται ο πελάτης, μέχρι να ολοκληρωθεί η κράτηση, επιτυχώς ή ανεπιτυχώς).

Στη συνέχεια καταστρέφουμε όλα τα mutexes και τη μεταβλητή συνθήκης, κάνοντας και τους απαραίτητους ελέγχους για πιθανά σφάλματα κατά την καταστροφή.

Τέλος, απελευθερώνουμε τη μνήμη που είχαμε δεσμεύσει και επιστρέφουμε 1.

Συνάρτηση Client

Υπολογίζουμε το τυχαίο ακέραιο πλήθος δευτερολέπτων που θα πρέπει να περιμένει ο πελάτης (waiting_time) στο διάστημα [Nseatlow, Nseathigh], τον τυχαίο ακέραιο αριθμό εισιτηρίων που θα ζητήσει (tickets) στο διάστημα [tseatlow,tseathigh] και το τυχαίο ποσοστό επιτυχίας της συναλλαγής (success) στο διάστημα [1,10] όπου 10 σημαίνει αποτυχία.

Περιμένουμε μέχρι ένας τηλεφωνητής να είναι διαθέσιμος χρησιμοποιώντας την pthread_cond_wait και τη μεταβλητή συνθήκης cond, και μόλις λάβουμε το σήμα ότι είναι διαθέσιμος, κλειδώνουμε το tel_mutex (ελέγχοντας και για σφάλματα) και μειώνουμε τη μεταβλητή tele κατά 1.

Στη συνέχεια το νήμα περιμένει για όσο χρόνο υποδεικνύει το waiting_time και κλειδώνει το time_mutex μέχρι να προσθέσει το τρέχον waiting_time στο συνολικό χρόνο αναμονής των πελατών. Έπειτα ξεκλειδώνει τα mutexes του χρόνου και του τηλεφωνητή, δίνει σήμα μέσω της cond και ξεκινάει τη μέτρηση χρόνου εξυπηρέτησης με την clock_gettime.

Αρχικοποιούμε τον πίνακα Return, ώστε και τα πέντε στοιχεία του να έχουν τιμή -1.

Εάν το θέατρο είναι γεμάτο (δηλαδή seats==0) εκτυπώνουμε το αντίστοιχο μήνυμα στην οθόνη, κλειδώνοντας και ξεκλειδώνοντας το screen_mutex. Με την clock_gettime λήγουμε τη μέτρηση χρόνου εξυπηρέτησης και, αφού κλειδώσουμε το time_mutex προσμετράμε το χρόνο αυτό στο συνολικό χρόνο εξυπηρέτησης. Κάνουμε έξοδο από το νήμα.

Αν το θέατρο δεν είναι γεμάτο, κλειδώνουμε το seats_mutex και ψάχνουμε για διαθέσιμες θέσεις και τις κρατάμε. Ταυτόχρονα αποθηκεύουμε τον αριθμό κάθε θέσης που κρατάμε στον πίνακα Return, σε περίπτωση που πρέπει να τις επιστρέψουμε πριν ολοκληρωθεί η συναλλαγή. Τέλος, ξεκλειδώνουμε το seats_mutex.

Εάν δεν υπάρχουν διαθέσιμες θέσεις, επιστρέφουμε τις θέσεις που κρατήσαμε έως στιγμής με τη βοήθεια του πίνακα Return στον οποίο έχουμε αποθηκεύσει τις θέσεις που δεσμεύσαμε, περιμένουμε μέχρι κάποιος τηλεφωνητής να είναι διαθέσιμος (κλειδώνοντας και ξεκλειδώνοντας το tel_mutex) και εκτυπώνουμε το κατάλληλο μήνυμα (κλειδώνοντας

και ξεκλειδώνοντας το screen_mutex). Με την clock_gettime λήγουμε τη μέτρηση χρόνου εξυπηρέτησης και, αφού κλειδώσουμε το time_mutex προσμετράμε το χρόνο αυτό στο συνολικό χρόνο εξυπηρέτησης. Κάνουμε έξοδο από το νήμα.

Εάν η πληρωμή γίνεται αποδεκτή (δηλαδή αν success<=9), τότε υπολογίζουμε το κόστος των θέσεων, περιμένουμε μέχρι ένας τηλεφωνητής να είναι διαθέσιμος, εκτυπώνουμε τα στοιχεία της συναλλαγής (κλειδώνοντας και ξεκλειδώνοντας το screen_mutex), ενημερώνουμε το υπόλοιπο στην τράπεζα (κλειδώνοντας και ξεκλειδώνοντας το bank_mutex).

Εάν η πληρωμή δεν γίνεται αποδεκτή (δηλαδή αν success=10), τότε περιμένουμε μέχρι ένας τηλεφωνητής να είναι διαθέσιμος, εκτυπώνουμε τα στοιχεία της συναλλαγής (κλειδώνοντας και ξεκλειδώνοντας το screen_mutex), επιστρέφουμε τις θέσεις που κρατήσαμε έως στιγμής, ξεκλειδώνουμε το seats_mutex. Έπειτα με την clock_gettime λήγουμε τη μέτρηση χρόνου εξυπηρέτησης και, αφού κλειδώσουμε το time_mutex προσμετράμε το χρόνο αυτό στο συνολικό χρόνο εξυπηρέτησης. Ξεκλειδώνουμε το time_mutex και κάνουμε έξοδο από το νήμα.

Εάν η συναλλαγή έχει ολοκληρωθεί με επιτυχία, δίνουμε σήμα ότι ένας ακόμη τηλεφωνητής είναι διαθέσιμος, αυξάνουμε τη μεταβλητή tele κατά 1 και ξεκλειδώνουμε το mutex του. Τέλος, με την clock_gettime λήγουμε τη μέτρηση χρόνου εξυπηρέτησης και, αφού κλειδώσουμε το time_mutex προσμετράμε το χρόνο αυτό στο συνολικό χρόνο εξυπηρέτησης. Ξεκλειδώνουμε το time_mutex και κάνουμε έξοδο από το νήμα.