ΑΝΑΦΟΡΑ ΕΡΓΑΣΙΑΣ
ΣΤΟ ΜΑΘΗΜΑ ΛΕΙΤΟΥΡΓΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

Μαρία Κονταράτου – p3200078
Γεώργιος Κουμουνδούρος – p3200083

ΙΟΥΝΙΟΣ 2022

Μέρος Α: Δομή κώδικα

Σε πρώτο στάδιο, στο αρχείο p3200078-p3200083-res.h γίνονται όλες οι δηλώσεις τόσο των global variables που χρησιμοποιούμε στις συναρτήσεις, όσο και των αρχικοποιήσεων των μεταβλητών που δίνονται από την εκφώνηση. Επιπλέον, κάνουμε δήλωση και αρχικοποίηση όλων των mutexes αλλά και δήλωση των συναρτήσεών μας.

Στο αρχείο p3200078-p3200083-res.c έχουμε τις μεθόδους main(), RSVseats(), cardCalculations() και main_thread().

void * main_thread(void *customer_id):
Εκτός από το cid που θα είναι ο αριθμός πελάτη, δημιουργούμε 5 structs για να μπορέσουμε να υπολογίζουμε το χρόνο αναμονής και εξυπηρέτησης πελατών. Αρχικά κάνουμε lock τους τηλεφωνητές και αν δεν υπάρχει διαθέσιμος αναμένουμε μέχρι να βρεθεί μέσω της pthread_cond_wait(). Όταν βρεθεί, τον δεσμεύουμε και κάνουμε unlock τους τηλεφωνητές. Σε αυτό το στάδιο, χρειάζεται να βρούμε πόσες θέσεις δίνουμε σε κάθε πελάτη, σημείο που θα κάνουμε τη χρήση του σπόρου για να βρούμε τυχαίες θέσεις, χρόνο αναμονής, ζώνη και χρόνο εξυπηρέτηση. Παράλληλα, επιλέγουμε τις τυχαίες θέσεις και ελέγχουμε αν υπάρχει αρκετή μνήμη για αυτές. Αν δεν μπορέσει να γίνει sleep από το νήμα του τηλεφωνητή, τον απελευθερώνουμε και βγαίνουμε εκτός συστήματος.

Ανάλογα τη ζώνη, δημιουργούμε για τις RSVseats() και cardCalculations() μερικές μεταβλητές και mutexes, οι οποίες αρχικοποιούνται στο αρχείο p3200078-p3200083-res.h. Η avail και m_avail χρησιμοποιούνται και ενημερώνουν τις διαθέσιμες θέσεις στην RSVseats(). Ομοίως χρησιμοποιούμε τις availSeatsForCards και m_seats για την cardCalculations() όπως και τα seats και m_seats για τη δημιουργία του πλάνου στη main.

Μέσω της RSVseats() ελέγχουμε εάν οι θέσεις είναι διαθέσιμες και ενημερώνουμε τις αντίστοιχες μεταβλητές. Μετά ελευθερώνομε τον τηλεφωνητή και εκτελούμε την αντίστοιχη διαδικασία για τους ταμίες με τη μόνη διαφορά ότι όταν χρειαστεί να πληρώσει ο χρήστης, χρησιμοποιούμε την cardCalculations(). Τέλος αποδεσμεύουμε τις θέσεις, τον ταμία και με lock και unlock mutexes υπολογίζουμε τους χρόνους αναμονής και εξυπηρέτησης.

void cardCalculations(int zone_id, int rand_seats, int cid, int *seats_id):
Αν η συναλλαγή είναι επιτυχής, τυπώνουμε τις θέσεις του πελάτη καθώς και το κόστος συναλλαγής όπως ζητούνται στην εκφώνηση. Αν είναι αποτυχημένη, ελευθερώνουμε τις θέσεις, ενημερώνουμε τις αντίστοιχες μεταβλητές και τυπώνουμε μήνυμα σφάλματος.

void RSVseats(int zone_id, int rand_seats, int cid, int *seats_id):
Όσο έχουμε διαθέσιμες θέσεις, ελέγχουμε εάν μπορούμε να τις δεσμεύσουμε και αντίστοιχα τις ενημερώνουμε μέσω της μεταβλητής avail. Αν δεν έχουμε θέσεις, ελευθερώνουμε τον τηλεφωνητή, τυπώνουμε μήνυμα σφάλματος, ελευθερώνουμε τις θέσεις και βγαίνουμε εκτός συστήματος.

int main(int argc, char *argv[]):

Ζητάμε από τον χρήστη το πλήθος πελατών προς εξυπηρέτηση Ncust και τον τυχαίο σπόρο seedT. Στη συνέχεια, αρχικοποιούμε τις θέσεις, τους αριθμούς πελατών και τα threads και περιμένουμε να τελειώσουν. Τέλος, τυπώνουμε τα ζητούμενα αποτελέσματα και το πλ'ανο θέσεων.

Μέρος Β: Σχόλια

Δυστυχώς προέκυψε ένα error στον κώδικά μας, στο κομμάτι του RSVseats() και δεν καταφέραμε να δημιουργήσουμε συναλλαγές αποτυχημένες λόγω έλλειψης θέσεων, γι'αυτό και το τελικό ποσοστό είναι 0.

Επιπλέον, στο ίδιο κομμάτι κώδικα φαίνεται στις θέσεις ότι σε κάποια παραδείγματα επικαλύπτονται από πολλούς πελάτες.