Λειτουργικά Συστήματα (Εργασία)

Αρχικά στο πρόγραμμά μας δηλώνουμε τα mutexes, conditions, διάφορες μεταβλητές (για στατιστικά, λογαριασμό εταιρίας κλπ.), δυο πίνακες τύπου int για τις θέσεις arrayZoneA και arrayZoneB (ένας πίνακας για κάθε ζώνη) και δυο πίνακες τυπου int με σκοπό να μετράμε πόσες θέσεις έχουν κρατηθεί σε κάθε σειρά countZoneA (NzoneA θεσεων) και countZoneB (NzoneB θεσεων). Αρχικοποιούμε και τους τέσσερις πίνακες με μηδενικά.

Main:

Στην main αρχικοποιούμε τα mutexes και τα conditions με την init. Μετά, με ένα for, δημιουργούμε ένα νήμα για κάθε πελάτη, με την create, και αναθέτουμε στα νήματα την συνάρτηση με όνομα service. Έπειτα, με ένα for, καλούμε την join για όλα τα νήματα. Τέλος καλούμε την printInfo, αποδεσμεύουμε την μνήμη που δεσμεύσαμε για τον πίνακα με τους αριθμούς πελατών και κάνουμε return -1.

service:

Η συνάρτηση service είναι η ρουτίνα που εκτελεί το νήμα πελάτη. Αρχικά κλειδώνουμε το firstMutex (βλ. «Διάφορα») και ελέγχουμε αν είναι ο πρώτος πελάτης που θα τηλεφωνήσει. Αν είναι ο πρώτος, κάνουμε την μεταβλητή first = 1. Αν δεν είναι ο πρώτος, παίρνουμε ένα τυχαίο αριθμό χ από treslow μέχρι treshigh, με τη βοήθεια της get_rand, και περιμένουμε για χ δευτερόλεπτα. Μετά με τη βοήθεια της clock_gettime δίνουμε τη χρονική στιγμή (που ξεκινούμε να ψάχνουμε για τηλεφωνητή) στις μεταβλητές start και startService. Συνεχίζουμε κλειδώνοντας το telMutex (βλ. «Διάφορα») και ελέγχουμε αν υπάρχει διαθέσιμος τηλεφωνητής. Αν δεν υπάρχει κάνουμε cond_wait. Μόλις βρούμε τηλεφωνητή, τον δεσμεύουμε αφαιρώντας τον μετρητή τηλεφωνητών κατά 1 και ξεκλειδώνουμε το telMutex. Δίνουμε την χρονική στιγμή που τελειώνει η αναμονή στην μεταβλητή stop και υπολογίζουμε τον χρόνο αναμονής του πελάτη. Ακολούθως παίρνουμε τυχαίους αριθμούς για να αποφασίσουμε τη ζώνη που επιθυμεί ο πελάτης και τον αριθμό θέσεων. Για να αποφασίσουμε τη ζώνη που επιθυμεί ο πελάτης παίρνουμε τυχαίο αριθμό από το 0 μέχρι το 9 και ελέγχουμε κατά πόσο είναι μεγαλύτερος ή μικρότερος του PzoneA*10. Έπειτα κλειδώνουμε το κατάλληλο mutex με βάση το ποια ζώνη θέλουμε (zoneAmutex ή zoneBmutex) και καλούμε την checkZone για να μας βρει θέσεις. Αν επιστρέψει -1 (που σημαίνει ότι δεν βρήκε) αποδεσμεύουμε τον τηλεφωνητή προσθέτοντας τον μετρητή κατά 1, ενημερώνουμε τα στατιστικά και τερματίζουμε με τα κατάλληλα μηνύματα. Αν βρούμε θέσεις, πάμε στον πίνακα θέσεων της ζώνης που θέλουμε και βάζουμε στις θέσεις μας τον αριθμό του πελάτη. Αποδεσμεύουμε τον τηλεφωνητή συνεχίζουμε ψάχνοντας για διαθέσιμο ταμία ακριβώς όπως κάναμε και με τον τηλεφωνητή (μετράμε και τον χρόνο αναμονής). Μετά με τη βοήθεια τυχαίων αριθμών περιμένουμε για ένα χρονικό διάστημα από tchashlow μέχρι tchashhigh και υπολογίζουμε αν πέτυχε η πληρωμή. Αν πέτυχει, ενημερώνουμε τα στατιστικά, προσθέτουμε το απαιτούμενο ποσό στον τραπεζικό λογαριασμό της εταιρίας και εμφανίζουμε τα κατάλληλα μηνύματα. Αν δεν πετύχει, πάμε στον πίνακα θέσεων και βάζουμε τον αριθμό 0 σε όποιες θέσεις κρατήσαμε (στον κατάλληλο πίνακα arrayZone), μειώνουμε τον μετρητή για την σειρά θέσεων που κρατήσαμε, ενημερώνουμε τα στατιστικά και τυπώνουμε τα απαιτούμενα μηνύματα. Τέλος, αποδεσμεύουμε τον ταμία, ενημερώνουμε τα υπόλοιπα νήματα με την cond_signal, υπολογίζουμε τον χρόνο εξυπηρέτησης και τον προσθέτουμε στον συνολικό και τερματίζουμε τον πελάτη.

checkZone:

Η checkZone χρησιμοποιείται για να ελέγξουμε αν υπάρχουν διαθέσιμες θέσεις. Παίρνει ως παραμέτρους τον αριθμό θέσεων που ψάχνουμε και μια boolean που δηλώνει σε ποια ζώνη να ψάξουμε. Αρχικά περιμένει για ένα χρονικό διάστημα από tseatlow μέχρι tseathigh. Μετά ελέγχει την διαθεσιμότητα και αν υπάρχουν θέσεις επιστρέψει το index της πρώτης από αυτές που βρήκε και αν δεν υπάρχουν επιστρέφει -1.

Για να ελέγξει αν υπάρχουν θέσεις, βλέπει τον κατάλληλο πίνακα countZone (A ή B ανάλογα σε ποια ζώνη ψάχνουμε) και συγκρίνει το ποσό τον θέσεων που ψάχνουμε με το πόσες θέσεις έχει η κάθε σειρά (πχ. Αν το countZoneA[0] = 10, σημαίνει ότι στην πρώτη σειρά στην ζώνη Α έχουν κρατηθεί 10 θέσεις) και αν υπάρχουν αρκετές επιστρέφουμε i*Nseat + countZone(A ή B)[i]. i*Nseat για να βρούμε τη σωστή σειρά συν countZone[i] (αν πχ. countZone[i] = 4 σημαίνει έχουν κρατηθεί 4ις θέσεις και επιστρέφω την 5^{η}).

printInfo:

Η printInfo παίρνει ως παράμετρο τον αριθμό τον πελατών (Ncust) και εμφανίζει πληροφορίες για την κάθε θέση που έχει κρατηθεί, τα έσοδα, το ποσοστό επιτυχημένων συναλλαγών, το ποσοστό των αποτυχημένων λόγω μη διαθεσιμότητας, το ποσοστό των αποτυχημένων λόγω αποτυχίας πληρωμής και τους μέσους χρόνους αναμονής και εξυπηρέτησης.

get_rand:

Η get_rand παίρνει ως όρισμα τον σπόρο και έναν αριθμό x και επιστρέφει ένα τυχαίο αριθμό από το 0 μέχρι το x-1. Την καλούμε κάθε φορά που θέλουμε να πάρουμε τυχαίο αριθμό.

Διάφορα:

- firstMutex: mutex για έλεγχο κατά πόσο το νήμα είναι το πρώτο που τρέχει.
- telMutex, cashMutex: mutexes για τηλεφωνητές και ταμίες.
- Bool first: Είναι 0 και ο πρώτος πελάτης την μετατρέπει σε 1.
- success, noSeats, cardFailure: μετρητές για τις επιτυχημένες και αποτυχημένες κρατήσεις.
- wait_time, service_time: συνολικοί χρόνοι αναμονής και εξυπηρέτησης.
- clientWait: χρόνος αναμονής του πελάτη.
- arrayZoneA, arrayZoneB: μονοδιάστατοι πίνακες θέσεων ζώνης Α και Β. (ενδεικτικά η θέση 1 σειρά 1 βρίσκεται στην θέση 0 του πίνακα και η θέσει 2 σειρά 2 βρίσκεται στην θέση 11).
- countZoneA, countZoneB: μονοδιάστατοι πίνακες που μετρούν τις κρατημένες θέσεις σε κάθε σειρά. (άρα στο countZoneA[0] είναι το ποσό των κρατημένων θέσεων στην ζώνη Α σειρά 1). Παράλληλα είναι και το index της επόμενης διαθέσιμης θέσης (πχ. αν countZoneB[3] = 5 σημαίνει ότι κρατήθηκαν ήδη 5 θέσεις σε αυτή τη σειρά και το index της επόμενης διαθέσιμης είναι το 5 της 4ης σειράς άρα υπολογίζεται με το i*Nseat + 5).
- Για την επιλογή ζώνης και την επιτυχία πληρωμής που λειτουργούμε με πιθανότητες, παίρνουμε ένα τυχαίο αριθμό από το 0 μέχρι το 9 και τον συγκρίνουμε με την πιθανότητα*10.

Περιορισμός

Τα δύο αρχεία .h και .c πρέπει να είναι στον ίδιο φάκελο για να τρέξουν σωστά.