ΠΡΟΓΡΑΜΜΑΤΙΣΤΙΚΗ ΕΡΓΑΣΙΑ – ΑΝΑΦΟΡΑ

Λειτουργικά Συστήματα 2021-2022

3200234, ΦΡΑΓΓΑΤΖΗΣ ΠΕΤΕΡ
3200056, ΣΤΥΛΙΑΝΟΣ – ΔΙΟΓΕΝΗΣ ΙΕΡΕΜΙΑΔΗΣ

Συνάρτηση main:

Το πρόγραμμα μας ξεκινάει. Ο αριθμός των πελατών που πρόκειται να εξυπηρετηθούν *Ncust* δίνεται μέσω της γραμμής εντολών ώς πρώτη παράμετρος στην εκτέλεση του προγράμματος (*argv*[1]). Ο σπόρος της γεννήτριας τυχαίων αριθμών *seed* δίνεται ώς δεύτερη παράμετρος (*argv*[2]). Για να μπορέσουμε να εξασφαλήσουμε έγκυρες τιμές στις δύο αυτές μεταβλητές, πρίν την ανάθεση των τιμών τους, γίνονται οι απαραίτητοι έλεγχοι και το πρόγραμμα τερματίζει αν δέν τηρείται μία απο τις δύο συνθήκες. Η συνάρτηση *gen_theatre()* φτιάχνει τον δυσδιάστατο πίνακα ακεραίων *theatre*[30][10] ο οποίος θα είναι το θέατρό μας και αρχικοποιεί τα πεδία του σε 0. Πάμε τώρα στο κομμάτι των νημάτων. Η εκφώνηση μας ζήτάει τα νήματα των πελατών να δημιουργόυνται απο ένα αρχικό νήμα. Δεσμεύουμε δυναμικά λοιπόν το αρχικό μας νήμα *t1* στη μνήμη με τη συνάρτηση *malloc()* και καλώντας την *pthread_create()* του δίνουμε να εκτελέσει την συνάρτηση *create_threads()*. Η *create_threads()* δημιουργεί ένα νήμα για τον κάθε πελάτη με δυναμική δέσμευση, με τον ίδιο τρόπο που δημιουργήθηκε και το *t1*. Ύστερα απο αυτό, το μονο που πρέπει να κάνει η main είναι να περιμένει να τερματίσει το *t1* (*pthread_join()*) και να το απελευθερώσει απο την μνήμη (*free(t1)*).

Συνάρτηση create_threads:

Τα νήματα που δημιουργόυνται εδώ είναι ουσιαστικά οι πελάτες μας. Ο κάθε πελάτης εκτελεί την συνάρτηση *krathsh()* Εκτός απο δυναμική δέσμευση των threads των πελατών, η create_threads αναθέτει επίσης στο καθένα ένα ξεχωριστό id (πίνακας *tIDs*[*Ncust*]) και τυπώνει την τελική έξοδο (πλάνο θέσεων θεάτρου και στατιστικά). Ο τυχαίος αριθμός δευτερολέπτων που υπάρχει μεταξύ των πελατών, αφού έχει ξεκινήσει ο πρώτος, υπολογίζεται με την συνάρτηση *gen_random(int a, int b)*, η οποία, με τη βοήθεια της *rand_r()* επιστρέφει έναν τυχαίο αριθμό στο διάστημα [*a, b*]. Το thread *t1* δε δημιουργεί δηλαδή τους πελάτες την ίδια στιγμή, αλλά κάνει *sleep()* ένα τυχαίο χρονικό διάστημα μέχρι να δημιουργήσει τον κάθε επόμενο.

Συνάρτηση krathsh:

Οι πελάτες περνάνε πρώτα απο τη διαδικασία με τους τηλεφωνητές (*thlefwnhma()*) και μετά με τους ταμείες (*tameio()*) για να κλείσουν τα εισητήριά τους. Στο τέλος τυπώνουν στην οθόνη μία σύνοψη της κράτησής τους. Για να εξασφαλίσουμε οτι ανα πάσα στιγμή το πολύ 3 πελάτες εξυπηρετούνται απο

τους τηλεφωνητές (*Ntel* = 3), όποιοι πελάτες καλούν την συνάρτηση *thefwnhma()* μειώνουν το πλήθος των διαθέσιμων τηλεφωνητών (*dia8esimoi_thl*) κατά 1 και το αυξάνουν πάλι κατα 1 όταν τελειώσει η συνάρτηση. Όταν κάποιος πελάτης δεί οτι υπάρχουν 0 διαθέσιμοι, περιμένει μέχρι να αυξηθούν πάλι (*pthread_cond_wait()*), δηλαδή μέχρι να ολοκληρώσει κάποιος άλλος το τηλεφώνημά του. Με τον ίδιο τρόπο γίνεται ο έλεγχος στους ταμείες (αντίστοιχη μεταβλητή για ταμείες *dia8esimoi_tam*). Κάθε πελάτης κρατάει μία μεταβλητή *cust_info* τύπου δομής *INFORMATION* που ορίσαμε στο αρχείο δηλώσεων. Εκεί αποθηκεύονται τα στοιχεία της κράτησής του. Τα στοιχεία αυτά παίρνουν τιμές μέσα απο τις συναρτήσεις *thlefwnhma()* και *tameio()*. Το τι τυπώνει ο κάθε πελάτης στο τέλος εξαρτάται απο την μεταβλητή *cust_info.status*. Παίρνει τιμή -1 αν ο τηλεφωνητής δεν καταφέρει να του βρεί θέσεις και 1 αν στο ταμείο η πληρωμή με κάρτα δεν έγινε αποδεκτή. Ανάλογα με την τιμή της τυπώνεται το ανάλογο μήνυμα.

Συναρτήσεις thlefwnhma και tameio:

Οι συναρτήσεις αυτές σε μεγάλο βαθμό λειτουργούν παρόμοια. Στην thlefwnhma υπολογίζονται ο αριθμός εισητηρίων, η ζώνη, σειρά και θέσεις του πελάτη, όπως επίσης και το κόστος των εισητηρίων υπο την προϋπόθεση οτι βρέθηκαν αρκετές συνεχόμενες θέσεις. Στην tameio γίνεται η προσπάθεια για την πληρωμή και τα χρήματα απο τα εισητήρια προστήθεντε στην μεταβλητή *company_account*. Και στις δύο συναρτήσεις πρίν γίνει κάτι απο τα παραπάνω υπάρχει ένας τυχαίος χρόνος αναμονής (πάλι με χρήση της *sleep()*). Επίπλέον, καταφέρνουμε να αναθέσουμε τιμές στη δομή *cust_info* του πελάτη, δίνοντας και στις δύο συναρτήσεις ως παράμετρο έναν δείκτη τύπου *INFORMATION* (μία θέση μνήμης δηλαδή). Έτσι μπορούμε να διαβάσουμε και να γράψουμε τοπικά τις τιμές του struct και η αλλαγές να εμφανιστούν και εξωτερικά.

Άλλα:

Γενικά, σε όποια σημεία έπρεπε κάποιο νήμα να πειράξει κάποια μεταβλητή που έβλεπαν και όλα τα υπόλοιπα χρησιμοποιήσαμε κλείδωμα με mutexes καλώντας τις pthread_mutex_lock() και pthread_mutex_unlock() όπου χρειαζόταν (π.χ. όταν έπρεπε να μειωθούν/αυξηθούν στην αρχή/τέλος οι διαθέσιμοι τηλεφωνητές/ταμίες, όταν ένας τηλεφωνητής/ταμίας έκλεινε/επέστρεφε θέσεις στο πλάνο του θεάτρου, πρόσθεση χρημάτων στο *company_account*, τύπωμα στην οθόνη για να μην μπέκονται τα μηνύματα μεταξύ τους…).

Οι μεταβλητές του *cust_info* για τη σειρά και τις θέσεις του πελάτη παίρνουν τιμές μέσα απο τη συνάρτηση *check_avail()*.