ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΛΕΙΤΟΥΡΓΙΚΑ ΣΥΣΤΗΜΑΤΑ

Υποχρεωτικό Μάθημα 4ου εξαμήνου

Εαρινό Εξάμηνο 2018-2019

1η Προγραμματιστική Εργασία

Σκοπός: Στις εργασίες σας θα υλοποιήσετε ένα σύστημα αγοράς θέσεων σε θεατρικές παραστάσεις με χρήση του πακέτου νημάτων POSIX threads (pthreads). Στο σύστημα αυτό κάθε πελάτης αρχικά κλείνει τις θέσεις που επιθυμεί, στη συνέχεια τις πληρώνει με πιστωτική κάρτα και, τέλος, τα χρήματα μεταφέρονται στο λογαριασμό της εταιρείας. Στην πρώτη εργασία θα κατασκευάσετε ένα απλουστευμένο σύστημα, το οποίο στη δεύτερη εργασία θα γίνει πιο ρεαλιστικό. Στα συστήματα αυτά έχουμε έναν μεγάλο αριθμό πελατών οι οποίο εξυπηρετούνται από έναν περιορισμένο αριθμό σημείων εξυπηρέτησης, συνεπώς το πρόγραμμά σας πρέπει να υλοποιεί αμοιβαίο αποκλεισμό (με mutexes) και συγχρονισμό (με μεταβλητές συνθήκης). Ο κώδικάς σας θα πρέπει να λειτουργεί σωστά στην εικονική μηχανή που διατίθεται στα CSLAB και στο διαδίκτυο (βλ. ανακοίνωση στο eclass). Η εργασία είναι ομαδική και είναι σχεδιασμένη για ομάδες τριών (3) ατόμων, μπορείτε όμως να την υλοποιήσετε και σε μικρότερες ομάδες. Ο βαθμός της 1ης εργασίας θα είναι το 20% του τελικού βαθμού (δηλαδή, 2 μονάδες στις 10).

Αντικείμενο: Το σύστημα αγοράς θέσεων διαθέτει έναν λογαριασμό στην τράπεζα, έναν μετρητή συναλλαγών, ένα πλάνο με τις N_{seat} θέσεις του θεάτρου κι ένα τηλεφωνικό κέντρο με N_{tel} τηλεφωνητές οι οποίοι εξυπηρετούν τους πελάτες (βρίσκουν θέσεις και χρεώνουν τις πιστωτικές κάρτες). Όταν όλοι οι τηλεφωνητές είναι απασχολημένοι, οι πελάτες περιμένουν τον επόμενο διαθέσιμο τηλεφωνητή. Όταν ένας πελάτης συνδεθεί με κάποιον τηλεφωνητή, επιλέγει έναν τυχαίο ακέραιο αριθμό εισιτηρίων στο διάστημα [$N_{seatlow}$, $N_{seathigh}$] και ο τηλεφωνητής χρειάζεται ένα τυχαίο ακέραιο πλήθος δευτερολέπτων στο διάστημα [$t_{seatlow}$,$t_{seathigh}$] για να δει αν υπάρχουν αρκετές θέσεις. Αν δεν υπάρχουν θέσεις, ο πελάτης λαμβάνει ένα μήνυμα λάθους και ολοκληρώνει την κλήση. Αν υπάρχουν θέσεις, δεσμεύονται στο πλάνο του θεάτρου και προχωράμε στην πληρωμή με πιστωτική κάρτα. Με πιθανότητα $P_{cardsucces}$ η πληρωμή γίνεται αποδεκτή, ο πελάτης χρεώνεται C_{seat} ευρώ ανά θέση, τα χρήματα μεταφέρονται στο λογαριασμό της εταιρείας και ο πελάτης ενημερώνεται για το συνολικό κόστος, τις θέσεις του και τον αριθμό συναλλαγής. Αν η πληρωμή αποτύχει, οι θέσεις επιστρέφονται στο πλάνο του θεάτρου και ο πελάτης λαμβάνει ένα μήνυμα λάθους. Όταν επεξεργαστεί N_{cust} πελάτες, το σύστημα τυπώνει τα αποτελέσματά του.

Είσοδος και δεδομένα: Το αρχικό ποσό στο λογαριασμό της εταιρείας είναι 0 ευρώ και ο αρχικός αριθμός συναλλαγής είναι 0. Οι ακόλουθες σταθερές θα ορίζονται σε ένα αρχείο δηλώσεων:

- N_{seat}=250 θέσεις
- N_{tel}=8 τηλεφωνητές
- $N_{seatlow}$=1 θέση
- $N_{seathigh}$=5 θέσεις
- $t_{seatlow}$=5 sec
- $t_{seathigh}$=10 sec
- $P_{cardsucces}$=90%
- C_{seat}=20 ευρώ

Το πρόγραμμά σας θα δέχεται δύο (ακριβώς) παραμέτρους με το πλήθος των πελατών προς εξυπηρέτηση, N_{cust}, και τον τυχαίο σπόρο για τη γεννήτρια των τυχαίων αριθμών.

Έξοδος εργασίας: Για κάθε πελάτη θα τυπώνεται στην οθόνη ένα από τα παρακάτω μηνύματα, ανάλογα με το πώς κατέληξε η κράτησή του, το οποίο θα ξεκινά με το μοναδικό αναγνωριστικό του πελάτη:

- Η κράτηση ολοκληρώθηκε επιτυχώς. Ο αριθμός συναλλαγής είναι <tid>, οι θέσεις σας είναι οι <a, b, …> και το κόστος της συναλλαγής είναι <X> ευρώ.
- Η κράτηση ματαιώθηκε γιατί η συναλλαγή με την πιστωτική κάρτα δεν έγινε αποδεκτή.
- Η κράτηση ματαιώθηκε γιατί δεν υπάρχουν αρκετές διαθέσιμες θέσεις.
- Η κράτηση ματαιώθηκε γιατί το θέατρο είναι γεμάτο.

Η σειρά των γραμμών θα είναι τυχαία, αλλά οι γραμμές δεν πρέπει να μπλέκονται μεταξύ τους. Στο τέλος της εκτέλεσης, το σύστημα θα τυπώνει τα ακόλουθα:

- Το πλάνο των θέσεων, π.χ. Θέση 1 / Πελάτης 3, Θέση 2 / Πελάτης 4, κ.λπ.
- 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).
- Το μέσο χρόνο εξυπηρέτησης των πελατών (από τη στιγμή που εμφανίζεται ο πελάτης, μέχρι να ολοκληρωθεί η κράτηση, επιτυχώς ή ανεπιτυχώς).

Δομή κώδικα: Το αρχικό νήμα του προγράμματός σας θα δημιουργεί ένα νήμα ανά πελάτη (συνολικά N_{cust} νήματα) στα οποία θα περνάτε έναν αριθμό νήματος (από 1 έως N_{cust}) ώστε να τα διακρίνετε. Κάθε νήμα στη συνέχεια θα εκτελεί τα παραπάνω βήματα μέχρι να ολοκληρωθεί η παραγγελία του και θα τυπώνει την κατάλληλη έξοδο. Τέλος, το αρχικό νήμα θα τυπώνει την τελική έξοδο. Θα χρειαστείτε τουλάχιστον τα ακόλουθα:

- Μία ακέραιη μεταβλητή και ένα mutex για να μετράτε το πλήθος των διαθέσιμων τηλεφωνητών και μία μεταβλητή συνθήκης για να συγχρονίσετε τους τηλεφωνητές με τους πελάτες, έτσι ώστε όταν δεν υπάρχουν διαθέσιμοι τηλεφωνητές να μπλοκάρονται οι πελάτες.
- Ακέραιες μεταβλητές και τα σχετικά mutex για τον τραπεζικό λογαριασμό και τον αριθμό συναλλαγής.
- Πραγματικές μεταβλητές και τα σχετικά mutex για τους μέσους χρόνους αναμονής και εξυπηρέτησης.
- Έναν πίνακα με το πλάνο των θέσεων κι ένα mutex για να τον κλειδώνετε (μην ξεχνάτε ότι χρειάζεται κλείδωμα και στη δέσμευση και στην αποδέσμευση θέσεων). Σκεφτείτε πώς μπορείτε να επιταχύνετε τη διαδικασία.
- Ένα mutex για κλείδωμα της οθόνης όταν τυπώνετε την έξοδο.

Προσοχή στο να τερματίζετε σωστά τα νήματα, να τα περιμένετε όπου χρειάζεται και (κυρίως!) να απελευθερώνετε σωστά όση μνήμη δεσμεύετε.

Υποδείξεις:

- Κατά τη μεταγλώττιση πρέπει να δίνετε την επιλογή `-pthread` για να χρησιμοποιηθεί η βιβλιοθήκη POSIX threads.
- Για να προσομοιώσετε το χρόνο που διαρκεί η αναζήτηση θέσεων θα χρησιμοποιήσετε την `unsigned int sleep(unsigned int seconds)`.
- Για να παράγετε μία σειρά ψευδοτυχαίων αριθμών, θα χρησιμοποιήσετε την `int rand_r(unsigned int *seedp)`. Δίνοντας διαφορετικούς αριθμούς στο σπόρο, θα έχετε διαφορετικές ακολουθίες εκτέλεσης. Χρησιμοποιήστε τον τελεστή % για να περιορίσετε το εύρος τιμών των τυχαίων αριθμών.
- Για τον υπολογισμό του χρόνου αναμονής και εξυπηρέτησης χρησιμοποιείστε την `int clock_gettime(clockid_t clk_id, const struct timespec *tp)` στην αρχή και τέλος κάθε πράξης, με πρώτη παράμετρο τη σταθερά `CLOCK_REALTIME`.
- Χρησιμοποιήστε έναν βρόχο με `while` για να ελέγχετε τη συνθήκη αναμονής και να κάνετε `pthread_cond_wait` για όσες φορές χρειάζεται.

Παραδοτέα: Ο κώδικάς σας πρέπει να αποτελείται από ένα αρχείο με δηλώσεις (συμπεριλαμβανομένων των σταθερών) και ένα αρχείο κώδικα C για το πρόγραμμα. Τα αρχεία αυτά πρέπει να έχουν ονόματα της μορφής `p3x-p3y-p3z-res1.h` για τις δηλώσεις, `p3x-p3y-p3z-res1.c` για τον κώδικα C , όπου και `p3x-p3y-p3z` είναι οι αριθμοί μητρώου σας. Εκτός από τον κώδικα, θα πρέπει να γράψετε μία αναφορά η οποία να περιγράφει τη δομή του κώδικά σας και να αναφέρει τυχόν περιορισμούς ή πρόσθετα χαρακτηριστικά που έχετε υλοποιήσει. Η αναφορά πρέπει να είναι ένα αρχείο σε μορφή PDF με όνομα της μορφής `p3x-p3y-p3z-res1.pdf`. Τέλος, θα πρέπει να συμπεριλάβετε ένα αρχείο με όνομα `test-res1.sh` το οποίο θα μεταγλωττίζει και θα εκτελεί το πρόγραμμά σας με παραμέτρους 100 πελάτες και αρχικό σπόρο 1000. Αυτά τα τέσσερα αρχεία (και τίποτα άλλο) θα πρέπει να συμπιεστούν σε ένα αρχείο σε μορφή 7zip με όνομα της μορφής `p3x-p3y-p3z-res1.7z` και να υποβληθούν από ένα μόνο μέλος της ομάδας μέσω της υποβολής εργασιών του eclass.

Προθεσμία υποβολής: Τα συμπιεσμένα αρχεία με τις εργασίες σας θα πρέπει να παραδοθούν μέσω του eclass μέχρι την Δευτέρα 15/4/2019 και ώρα 23:59.

Βαθμολόγηση και εξέταση: Αρχεία που δεν θα έχουν την *ακριβή* ονοματολογία που ζητείται παραπάνω *θα μηδενιστούν*. Εργασίες με ομοιότητες που υποδεικνύουν αντιγραφή *θα μηδενιστούν όλες* (θα γίνει έλεγχος με ειδικό πρόγραμμα). Δειγματοληπτικά, θα κληθούν ορισμένες ομάδες για προφορική εξέταση στα CSLAB. Μέλη ομάδων, ή και ολόκληρες ομάδες, που δεν θα προσέλθουν στην προφορική εξέταση, *θα μηδενιστούν*.