ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
Τμήμα Πληροφορικής και Τηλεπικοινωνιών
3η Εργασία - Τμήμα: Αρτίων Αριθμών Μητρώου
K22: Λειτουργικά Συστήματα – Χειμερινό Εξάμηνο '16
Ημερομηνία Ανακοίνωσης: 23/11
Ημερομηνία Υποβολής: 18/12 και Ώρα 23:59

Εισαγωγή στην Εργασία:

Ο στόχος της εργασίας είναι να δημιουργήσετε ανεξάρτητα προγράμματα τα οποία μπορούν να τρέχουν ταυτόχρονα και να προσομοιώνουν την λειτουργία ενός εστιατορίου. Καλείστε να υλοποιήσετε 3 τύπους διεργασιών πού η κάθε μία έχει διαφορετικό ρόλο. Οι τρεις τύποι διεργασιών είναι: *1*) η ομάδα πελατών ή γκρουπ, *2*) ο σερβιτόρος και *3*) ο υπεύθυνος υποδοχής. Οι παραπάνω διεργασίες είναι ανεξάρτητα προγράμματα τα οποία μπορούν να τρέχουν *ταυτόχρονα* και να *υλοποιούν συνεργατικά* το τι συμβαίνει στο εστιατόριο

Ο ρόλος του γκρουπ που αποτελείται από ένα αριθμό πελατών (customer group) είναι ότι η συγκεκριμένη παρέα έρχεται στο εστιατόριο για φαγητό και να συζητήσει. Όταν έχει την ευκαιρία, ένας σερβιτόρος (waiter) εξυπηρετεί αποκλειστικά το ίδιο γκρουπ πελατών. Τέλος, ο υπεύθυνος υποδοχής (doorman) αναλαμβάνει να ελέγχει τη διαθεσιμότητα των τραπεζιών, οδηγεί τις ομάδες των πελατών είτε στο μπαρ (εάν δεν υπάρχει διαθέσιμο τραπέζι) είτε στο τραπέζι τους, είτε τους ενημερώνει (πιο σπάνια) ότι το εστιατόριο είναι πλήρες.

Για δική σας διευκόλυνση, καλείστε να δημιουργήσετε και ένα πρόγραμμα (restaurant) που αρχικοποιεί τα δεδομένα που απαιτούνται, δημιουργεί τις απαραίτητες διεργασίες με την βοήθεια κλήσεων `fork()` και `exec*()` και τελικά κάνει απολογισμό και «κλείνει» το εστιατόριο. Το πρόγραμμα restaurant μπορεί επίσης να δημιουργεί τους semaphores, το shared segment, και οτιδήποτε άλλο χρειάζεται. Στο shared segment θα πρέπει να είναι αποθηκευμένα όλα τα στοιχεία λειτουργίας του εστιατορίου και τα αντίστοιχα λογιστικά/στατιστικά δεδομένα. Εναλλακτικά για την προτεινόμενη διεργασία restaurant μπορείτε να χρησιμοποιείτε πολλαπλά ttys για να ξεκινήσετε χειρωνακτικά ότι διεργασία επιθυμείτε να έχετε στην λύση σας. Το Σχήμα 1 δείχνει την συνολική εικόνα της της λειτουργίας του εστιατορίου.

Σε αυτή την άσκηση θα πρέπει να:

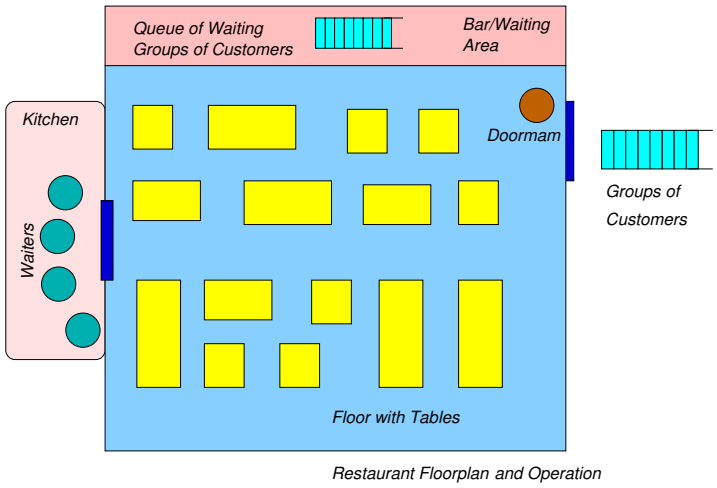
1. χρησιμοποιήσετε ένα σετ από σηματοφόρους ώστε να έχετε μια επιτυχή συνεργασία μεταξύ των ανεξάρτητων διεργασιών, *shared memory segment* για την σωστή εξυπηρέτηση των γκρουπ των πελατών, όπως επίσης και να δημιουργήσετε σχετικές δομές όπου χρειάζονται,
2. έχετε όλες τις διεργασίες να προσαρτήσουν το παραπάνω *shared memory segment* ώστε να μπορούν να προσπελάσουν περιεχόμενα ενδιαφέροντός τους,
3. χρησιμοποιήστε *POSIX Semaphores* για να υλοποιήσετε τη λύση σας που θα εμπεριέχει σχετικές κλήσεις P() και V(),
4. παρέχετε τη δυνατότητα τα προγράμματά σας να παραμείνουν 'απασχολημένα' δηλ. να κάνουν `sleep()` για περιόδους χρόνου που καθορίζονται με παραμετρικό τρόπο.

Θα πρέπει να *επιδείξετε την ορθότητά* της λύσης σας.

Διατύπωση του Προβλήματος:

Το Σχήμα 1 παρέχει μια προτεινόμενη βασική οργάνωση διεργασιών που δουλεύουν ταυτόχρονα για να επιτύχουν την συνεργατική λειτουργία του εστιατορίου.

Κάθε γκρουπ μόλις φτάσει στο εστιατόριο, πρέπει με την βοήθεια του doorman να τοποθετηθεί σε ένα τραπέζι. Με την διεργασία γκρουπ, πάντα αναφερόμαστε σε μια ομάδα ατόμων δηλ. μια παρέα που αποτελείται από 1 μέχρι 8 άτομα (ο αριθμός δημιουργείται τυχαία). Η διεργασία customer πρέπει να δηλώσει στον doorman το



Σχήμα 1: Διάγραμμα για την λειτουργία του Εστιατορίου.

πλήθος των ατόμων που αποτελείται ώστε ο doorman να ελέγξει αν υπάρχει διαθέσιμο τραπέζι. Όταν φτάνει ένα γκρουπ στην είσοδο μπορεί να περιμένει κάποια ώρα για εξυπηρέτηση αν ο doorman δεν είναι άμεσα διαθέσιμος δημιουργώντας έτσι μια ουρά η οποία δεν έχει περιορισμό στο μέγεθος. Εφόσον πάρει ένα τραπέζι, η διαδικασία customer απευθύνεται στο πρώτο διαθέσιμο σερβιτόρο, τον οποίο θα απασχολήσει ένα τυχαίο χρόνο για να παραγγείλει, κι ύστερα από ένα τυχαίο διάστημα (μέσα στο οποίο το γκρουπ τρώει και συζητά), θα απευθυνθεί και πάλι στον ίδιο σερβιτόρο ζητώντας να πληρώσει και να φύγει.

Ο doorman εξυπηρετεί τους πελάτες με FCFS-τρόπο δίνοντας τις εξής επιλογές:

1. να καθίσουν εφόσον υπάρχει διαθέσιμο τραπέζι,
2. να περιμένουν στο μπαρ, εφόσον υπάρχει χώρος. Όταν βρίσκονται στο μπαρ οι επιλογές που έχουν είναι:
   - να καθίσουν μόλις ελευθερωθεί τραπέζι,
   - να φύγουν αν δεν θέλουν να περιμένουν άλλο.
3. να φύγουν (αναγκαστικά αν δεν υπάρχει τραπέζι ή χώρος στο μπαρ, ή αν επιλέξουν ότι δε θέλουν να περιμένουν στο μπαρ).

Οι πελάτες που επιλέγουν να περιμένουν στο μπαρ, έχουν προτεραιότητα, το οποίο σημαίνει ότι μόλις ένα τραπέζι αδειάσει ο doorman πρέπει πρώτα να εξυπηρετήσει το πρώτο πελάτη από το μπαρ που χωράει στο τραπέζι. Το εστιατόριο αποτελείται από τραπέζια που έχουν χωρητικότητά 2, 4, 6, και 8 θέσεων. Ο συγκεκριμένος αριθμός των παραπάνω τραπεζιών μπορεί να δίνεται από ένα configuration file.

Το εστιατόριο διαθέτει W σερβιτόρους, οι οποίοι αναλαμβάνουν να εξυπηρετήσουν τους πελάτες (παραγγελία και πληρωμή) και όταν φύγει ένας πελάτης ενημερώνουν ότι το συγκεκριμένο τραπέζι έχει αδειάσει. Εφόσον ένας σερβιτόρος ξεκινήσει να εξυπηρετεί ένα νέο customer, θα τον εξυπηρετεί αποκλειστικά εκείνος μέχρι η διαδικασία customer να φύγει.

Συμπεριφορά Διεργασιών

Η διεργασία *πελάτης* θα πρέπει να εκτελεί τις παρακάτω λειτουργίες:

1. Στην είσοδο:
   (α′) Αναμονή στην ουρά για την εξυπηρέτηση από το doorman.
   (β′) Ενημέρωση doorman για το μέγεθος του γκρουπ.
   (γ′) Πιθανή αναμονή στο μπαρ.

   (δ′) Πιθανή αναχώρησή (αν δεν υπάρχει χώρος στο εστιατόριο και μπαρ).
   (ε′) Πιθανή εγκατάσταση σε τραπέζι.
2. Στο μπαρ (εφόσον χρειαστεί ένα γκρουπ να περιμένει)
   (α′) Εγκατάσταση σε τραπέζι που γίνεται διαθέσιμο.
   (β′) Πιθανή αναχώρηση.
3. Στο τραπέζι:
   (α′) Παραγγελία
   (β′) Πληρωμή
   (γ′) Αναχώρηση

Η διεργασία *σερβιτόρος* θα πρέπει να εκτελεί τις παρακάτω λειτουργίες:

1. να πάρει παραγγελία,
2. να δώσει το λογαριασμό (σε παρέα από την οποία πήρε παραγγελία)
3. να ενημερώσει όταν το τραπέζι αδειάσει.

Η διεργασία *doorman* θα πρέπει να εκτελεί τις παρακάτω λειτουργίες:

1. να ελέγχει την διαθεσιμότητα τραπεζιών,
2. να τοποθετήσει ένα γκρουπ (customer) που περιμένει στο μπαρ να καθίσει,
3. να εξυπηρετήσει ένα πελάτη από τη ουρά στην είσοδο,
4. να «οδηγήσει» ένα γκρουπ στο μπαρ,
5. να τοποθετήσει ένα customer σε τραπέζι.

Η διεργασία *restaurant* αναλαμβάνει το διάβασμα από ένα configuration file που περιέχει βασικές παραμέτρους για τη δημιουργία του εστιατορίου όπως:

1. αριθμό σερβιτόρων
2. χωρητικότητα μπαρ
3. αριθμό τραπεζιών και χωρητικότητα καθενός

Στη συνέχεια αναλαμβάνει τη δημιουργία κοινής μνήμης (shared memory) με χρήσιμες πληροφορίες όπου θα έχουν πρόσβαση *όλες* οι διεργασίες. Επίσης το restaurant αναλαμβάνει και τη δημιουργία των διεργασιών. Οποιαδήποτε χρονική στιγμή η διεργασία restaurant μπορεί να εξάγει στατιστικά στοιχεία που αφορούν στη λειτουργία του εστιατορίου. Η κοινή μνήμη θα πρέπει να έχει πληροφορίες για τη κατάσταση στο μπαρ, αν το κάθε τραπέζι είναι διαθέσιμο (και από πιο γκρουπ), στατιστικά σε σχέση με το χρόνο παραμονής, τις εισπράξεις και ότι άλλο κρίνεται απαραίτητο.

Σχεδιασμός των Προγραμμάτων σας:

Έχετε ελευθερία να επιλέξετε οποιαδήποτε δομή επιθυμείτε για τα προγράμματα σας. Θα πρέπει να υιοθετήσετε ένα (περιορισμένο) αριθμό από σηματοφόρους, βοηθητικές δομές και 4 ειδών προγράμματα. Όπως έχουμε αναφέρει, ίσως θα ήταν καλή ιδέα να αναπτύξετε ένα πρόγραμμα (το restaurant) που δημιουργεί το shared segment, αρχικοποιεί δομές και καταστάσεις, και τέλος γνωστοποιεί το ID του κοινού τμήματος σε άλλα ενδιαφερόμενα εκτελέσιμα από την γραμμή εντολής τους. Το εν λόγω πρόγραμμα θα μπορούσε να αρχικοποιήσει και τους σηματοφόρους που είναι απαραίτητοι για την λύση που επιθυμούμε.

Τα προγράμματά σας θα πρέπει να δημιουργούν εξόδους που με εύκολα κατανοητό τρόπο να μπορούν να δείξουν την ορθότητα αλλά και το ταυτόχρονο της εκτέλεσής τους με άλλες διεργασίες.

Στο τέλος της διεκπεραίωσης όλων των γκρουπ πελατών, επιβάλλεται κάποιο πρόγραμμα να *καθαρίσει* και να *διαγράψει* το κοινό τμήμα μνήμης και τους σηματοφόρους που χρησιμοποιήθηκαν. Η απελευθέρωση (purging)

τέτοιων πόρων είναι επιτακτική. Σε διαφορετική περίπτωση υπάρχει κίνδυνος ο πυρήνας να μην μπορεί να εξυπηρετήσει μέλλουσες ανάγκες σε κοινή μνήμη και σηματοφόρους.

Γραμμή Κλήσης των Προγραμμάτων:

Ο *customer* μπορεί να κληθεί ως εξής:
./customer -n people -d period -s shmid
όπου

- *customer* είναι το εκτελέσιμο του γκρουπ πελατών,
- η σημαία *-n people* παρέχει τον αριθμό των ατόμων που είναι σε μια παρέα και μπορεί να είναι ένας αριθμός από 1 έως και 8,
- η σημαία *-d period* παρέχει την μέγιστη δυνατή χρονική διάρκεια για την οποία το γκρουπ θα παραμείνει στο εστιατόριο αφότου έχει δώσει την παραγγελία του (κάνοντας `sleep()`).
- η σημαία *-s shmid* δίνει το κλειδί που το κοινό τμήμα μνήμης έχει (και όπου βρίσκονται δομές, σηματοφόροι, και οποιαδήποτε άλλη βοηθητική δομή/μεταβλητή που απαιτείται).

Το πρόγραμμα waiter μπορεί να κληθεί εξής:
./waiter -d period -m moneyamount -s shmid
όπου

- *waiter* είναι το εκτελέσιμο του σερβιτόρου,
- η σημαία *-d period* παρέχει την μέγιστη χρονική διάρκεια (t_{waiter}) για την οποία ένας σερβιτόρος θα πάρει για να εξυπηρετήσει ένα γκρουπ (είτε να πάρει παραγγελία είτε να τους δώσει το λογαριασμό). Ο πραγματικός χρόνος αναμονής επιλέγεται τυχαία στο διάστημα $[1 \dots t_{counter}]$,
- η σημαία *-m moneyamount* παρέχει το μέγιστο λογαριασμό που μπορεί να εκδοθεί σε ένα γκρουπ. Το πραγματικό ποσό επιλέγεται τυχαία στο διάστημα $[1 \dots moneyamount]$,
- η σημαία *-s shmid* δίνει το κλειδί που το κοινό τμήμα μνήμης έχει (και όπου βρίσκονται σηματοφόροι και οποιαδήποτε άλλη βοηθητική δομή/μεταβλητή που απαιτείται).

Το πρόγραμμα πορτιέρης μπορεί να καλεστεί ως εξής:
./doorman -d time -s shmid
όπου

- *doorman* είναι το εκτελέσιμο,
- η σημαία *-d period* παρέχει την μέγιστη χρονική διάρκεια (t_{waiter}) για την οποία ο πορτιέρης παίρνει για να εξυπηρετήσει μια παρέα που έχει εμφανιστεί στην είσοδο και αφότου έχει ξεκινήσει η εξυπηρέτηση αυτού του γκρουπ,
- η σημαία *-s shmid* δίνει το κλειδί που το κοινό τμήμα μνήμης έχει (και όπου βρίσκονται σηματοφόροι, και οποιαδήποτε άλλη βοηθητική δομή/μεταβλητή που απαιτείται).

Το πρόγραμμα *restaurant* τότε μπορεί να κληθεί ως εξής:
./restaurant -n customers -l configfile -d time
όπου

- *restaurant* είναι το εκτελέσιμο,
- η σημαία *-n* ορίζει τον μέγιστο αριθμών εισερχόμενων γκρουπ που το πρόγραμμα σας θα αντιμετωπίσει. Μετά από αυτόν το αριθμό, και αφότου όλοι οι εκκρεμούντες πελάτες έχουν αναχωρήσει το *ρεσταυρανθ* ολοκληρώνει την εργασία του τυπώνοντας τα στατιστικά του εστιατορίου,
- η σημαία *-l* ορίζει το αρχείο *configfile* με πληροφορίες για το εστιατόριο όπως αριθμός και χωρητικότητά τραπεζιών, χωρητικότητά μπαρ, μέγιστος αριθμός από σερβιτόρους που μπορούν να δουλεύουν οποιαδήποτε στιγμή στο εστιατόριο, και στατιστικά χρήσης του εστιατορίου.

- η σημαία *-d* ορίζει τον αριθμό *time* των δευτερολέπτων μετά από το οποία το πρόγραμμα να τυπώνει στην έξοδα όλα τα στατιστικά που είναι: αριθμός ανθρώπων στα τραπέζια, άνθρωποι στο μπαρ, άνθρωποι/γκρουπς που έχουν εξυπηρετηθεί, γκρουπς που έφυγαν, έσοδα ως τώρα, γκρουπς που αναχώρησαν αφότου περίμεναν στο μπαρ, και συνολικά έσοδα ανά τραπέζι μέχρι στιγμής.

Η σειρά με την οποία εμφανίζονται οι σημαίες δεν είναι προκαθορισμένη. Προφανώς μπορείτε να χρησιμοποιήσετε πιο πολλές σημαίες στα παραπάνω προγράμματα ώστε να διευκολυνθείτε ή και να σχεδιάσετε τα προγράμματά σας *ριζικά διαφορετικά*.

Τέλος καλό θα ήταν –και για λόγους επιβεβαίωσης και φυσικά μόνο για το σκοπό της άσκησης– να υπάρχει ένα μηχανισμός logging με την μορφή ενός append-only αρχείου. Εδώ, αποτυπώνεται όλη η δραστηριότητά μέχρι στιγμής ώστε να έχετε ένα εύκολο τρόπο να βλέπετε τι γίνεται όσον αφορά στην ταυτόχρονη εκτέλεση των παραπάνω προγραμμάτων.

Διαδικαστικά:

- Υπεύθυνοι για την άσκηση αυτή (ερωτήσεις, αξιολόγηση, βαθμολόγηση κλπ.) η Αργυρώ Κοκογιαννάκη `argirok+AT-di`, και Δήμος Βασιλικάκης `charidimosv+AT-di`.
- Το πρόγραμμά σας θα πρέπει να γραφτεί σε `C` (ή `C++` αν θέλετε αλλά χωρίς την χρήση `STL/Templates`) και να τρέχει στις μηχανές `Linux workstations` του τμήματος.
- Η χρήση του `Makefile` είναι *επιτακτική!*
- Παρακολουθείτε την ιστοσελίδα του μαθήματος `http://www.di.uoa.gr/~ad/k22/` για επιπρόσθετες ανακοινώσεις αλλά και την ηλεκτρονική-λίστα (η-λίστα) του μαθήματος στο URL `https://piazza.com/uoa.gr/fall2016/k22/home`
- Μέσα από την σελίδα `https://piazza.com/uoa.gr/fall2016/k22/home` θα μπορείτε να κάνετε ερωτήσεις και να δείτε απαντήσεις ή/και διευκρινήσεις που δίνονται σχετικά με την άσκηση (όπως έγινε και στις προηγούμενες ασκήσεις).

Τι πρέπει να Παραδοθεί:

1. Μια σύντομη και περιεκτική εξήγηση για τις επιλογές που έχετε κάνει στο σχεδιασμό του προγράμματος σας (1-2 σελίδες σε ASCII κειμένου είναι αρκετές).
2. Οπωσδήποτε ένα `Makefile` (που να μπορεί να χρησιμοποιηθεί για να γίνει αυτόματα το compile του προγράμματος σας). Πιο πολλές λεπτομέρειες για το (`Makefile`) και πως αυτό δημιουργείται δίνονται στην ιστοσελίδα του μαθήματος.
3. Ένα `tar-file` με όλη σας την δουλειά σε έναν κατάλογο πιθανώς να φέρει το όνομα σας και θα περιέχει όλη σας την δουλειά δηλ. `source files, header files, output files` (αν υπάρχουν) και οτιδήποτε άλλο χρειάζεται.

Άλλες Σημαντικές Παρατηρήσεις:

1. Οι εργασίες είναι **ατομικές**.
2. Το πρόγραμμα σας θα πρέπει να τρέχει στα `Linux` συστήματα του τμήματος αλλιώς **δεν μπορεί να βαθμολογηθεί**.
3. Αν και αναμένεται να συζητήσετε με φίλους και συνεργάτες το πως θα επιχειρήσετε να δώσετε λύση στο πρόβλημα, αντιγραφή κώδικα (οποιαδήποτε μορφής) είναι κάτι που **δεν επιτρέπεται** και δεν πρέπει να γίνει. Οποιοσδήποτε βρεθεί αναμεμειγμένος σε αντιγραφή κώδικά **απλά παίρνει μηδέν** στο μάθημα. Αυτό ισχύει για **όσους εμπλέκονται ανεξάρτητα** από το ποιος έδωσε/πήρε κλπ.
4. Υποβολές που **δεν χρησιμοποιούν** `Makefile` χάνουν αυτόματα 5% του βαθμού.
5. Σε καμιά περίπτωση τα *Windows δεν είναι επιλέξιμη* πλατφόρμα για την παρουσίαση αυτής της άσκησης.