ΛΕΙΤΟΥΡΓΙΚΑ ΣΥΣΤΗΜΑΤΑ
ΑΝΑΦΟΡΑ ΕΡΓΑΣΙΑΣ

Μαρία Φωτεινή Αρνίδη - p3150004
Γιώργος Σέρβος - p3150155

Στην αναφορά που ακολουθεί γίνεται μια περιεκτική ανάλυση της εργασίας που υλοποιήσαμε χωρισμένη κατα αρχεία και μεθόδους.

-pizza.h:
-- Σ'αυτό το αρχείο το header file γίνεται δήλωση σταθερών και μεθόδων που χρησιμοποιούνται στο πρόγραμμα.

-pizza.c:
--Σ'αυτό το αρχείο υλοποιείται όλη η λογική που περιγράφεται στην εκφώνηση της εργασίας.
Αρχικά έξω από την main function δηλώνουμε μεταβλητές που θα χρησιμοποιήσουν τα threads.'Επειτα ,παίρνουμε τα ορίσματα που περάσαμε στο πρόγραμμα.
και στην συνέχεια κάνουμε έναν λογικό έλεγχο ότι ο αριθμός των φούρνων δεν είναι μικρότερος από τον μέγιστο αριθμό από πίτσες σε μία παραγγελία, γιατί διαφορετικά δεν θα μπορούσε να τρέξει το πρόγραμμα (αφού οι πίτσες μιας παραγγελίας ξεκινούν να ψήνονται παράλληλα. Στη συνέχεια εκτυπώνουμε τις σταθερές με τις οποίες τρέχει το πρόγραμμα ώστε να γνωρίζουμε τι output παράγεται ανάλογα με ποιο input.
Κάνουμε initialize τα mutexes & condition variables και allocate μνήμη που θα χρειαστούμε (με χρήση malloc).
Στη συνέχεια ξεκινάμε τα διάφορα threads καθένα ξεχωριστό για την λειτουργία που τα χρειαζόμαστε.Μετά, παίρνουμε και υλοποιούμε την περίπτωση να μην δοθεί καμία παραγγελία για να αποφύγουμε το να κολλήσει το πρόγραμμα μας επ' αόριστον.
Ακόμα, υπολογίζουμε τα στατιστικά που χρειάζεται και τα εκτυπώνουμε και τέλος κάνουμε free τα mutexes και τα condition variable που είχαμε δεσμεύσει.

**currenttimeseconds(**): επιστρέφει τα δευτερόλεπτα που πέρασαν από 01-01-1970 για να χρησιμοποιηθεί στα στατιστικά χρόνου.

randomint() : γεννήτρια ψευδοτυχαίων στο εύρος των μεταβλητών που από την εκφώνηση έχουν οριστεί.

output(char *out): Με την pthread_mutex_lock/unlock σιγουρεύουμε ότι θα κάνει print ένα thread τη φορά.

orderrun(void *ordno): Παίρνουμε το χρόνο που ξεκινάει το thread και βάζουμε το thread του πελάτη να περιμένει για διαθέσιμο τηλεφωνητή (ένα thread "περνάει" κάθε φορά). Στη συνέχεια περιμένει να ολοκληρωθεί η πληρωμή. Αν η πληρωμή αποτύχει το thread κλείνει (η παραγγελία ουσιαστικά ακυρώνεται) (γρ. 477-490). Διαφορετικά, προσθέτουμε έναν διαθέσιμο τηλεφωνητή (αφου η παραγγελία δόθηκε και το τηλέφωνο έκλεισε), δίνεται σήμα σε κάποιον available cook και συνεχίζεται η διαδικασία.

cookrun(void *arg): Εδώ μπαίνουμε σε ένα while όπου αρχικά σιγουρεύουμε πώς μόνο ένας cook θα πάρει την παραγγελία. Η Cookcurrentorder είναι η παραγγελία που κοιτάει ο μάγειρας προκειμένου να πάει να τη μαγειρέψει. Το maxorderno είναι ουσιαστικά το πόσοι πελάτες ήρθαν. Αν δεν υπάρχει άλλη δουλειά να γίνει βγαίνει από το while (1ο if), αν δεν έχει δοθεί ακόμα αριθμός από πίτσες σημαίνει πως δεν έχει περάσει ακόμα παραγγελία και οπότε περιμένει (2ο if) και αν δεν περάσει η πληρωμή προχωρά στην επόμενη παραγγελία (3ο if). Ορίζουμε λοιπόν με ποια παραγγελία ασχολούμαστε(handlingorder), περιμένει ο cook να γίνουν διαθέσιμοι αρκετοί φούρνοι, παίρνουμε το πόσους φούρνους χρειαζόμαστε και για κάθε έναν ελέγχουμε αν είναι διαθέσιμος και αν ναι του κάνουμε signal ώστε να ξεκινήσει να ψήνει.

ovenrun(void *arg): Έχουμε ένα while από το οποίο κάνουμε break όταν έχουν τελειώσει όλοι οι μάγειρες. Το thread του φούρνου κάνει sleep για TBAKE (χρόνος ψησίματος) και μόλις είναι έτοιμες όλες οι πίτσες της παραγγελίας , κάνει signal στον packeter (γρ. 644-648).

packeterrun(void *arg): Κλειδώνουμε το mutex για να κοιτάξουμε την κύρια μεταβλητή(αυτό δεν ξέρω αν πρέπει να μπει) readytopack αν όλες οι πίτσες της παραγγελίας έχουν ψηθεί τότε η παραγγελία μπαίνει στην ουρά για πακετάρισμα. Στην συνέχεια παίρνουμε όσες παραγγελίες είναι έτοιμες και κάνουμε unlock το mutex. Στην περίπτωση που δεν είναι καμία παραγγελία έτοιμη και οι φούρνοι είναι άδειοι και οι μάγειρες έχουν τελειώσει είμαστε έτοιμοι για exit κάνοντας free την packetqueue και στην συνέχεια στέλνουμε σήμα στους φούρνους. Στην περίπτωση που οι φούρνοι δεν είναι άδειοι περιμένουμε να μας καλέσει κάποιος φούρνος για να συνεχίσει η διαδικασία. Αμέσως μετά στον κώδικα γίνεται το πακετάρισμα. Παίρνουμε δηλαδή μία από τις έτοιμες παραγγελιές “πέφτουμε για ύπνο” , κρατάμε τον χρόνο και αποδεσμεύουμε τους αντίστοιχους φούρνους , στην συνέχεια “ξυπνάμε” για να πάρει σειρά η επόμενη έτοιμη παραγγελία και δίνουμε σήμα στο ντελίβερι.

delivererrun(void * arg) : Κλειδώνω το mutex για να ελέγξω την μεταβλητή readytopack. Στην συνέχεια ελέγχω αν υπάρχει παραγγελια έτοιμη για ντελίβερι και ξεκλειδώνω. Επειτα, ελέγχω αν βρέθηκε έτοιμη παραγγελία. Στην περίπτωση που δεν έχει βρεθεί κάνει exit και περιμένει σήμα από τον packeter. Στην περίπτωση που έχει βρεθεί συνεχίζει και ξεκινά το delivery με τον αντίστοιχο μετρητή χρόνου και τις εκτυπώσεις των στατιστικών. Τέλος, γίνεται η επιστροφή στο μαγαζί. (έχω έναν ενδοιασμό για το delivery θα σου πω στην κλείσει αναλυτικά)