Ονοματεπώνυμο:Καρράς Κωνσταντίνος Α.Μ.:3180076
Ονοματεπώνυμο:Πέππα Χριστίνα Α.Μ.:3180154

Στην αρχή του κώδικα κάνουμε import τις απαραίτητες βιβλιοθήκες και το header αρχείο, αρχικοποιούμε 3 global μεταβλητές που θα χρειαστούμε για να κρατήσουμε τον αριθμό από τους μάγειρες, τους φούρνους και τους διανομείς, 2 μεταβλητές για τον μέσο και τον μέγιστο χρόνο ολοκλήρωσης, 2 μεταβλητές για τον μέσο και τον μέγιστο χρόνο κρυώματος, 2 δείκτες σε double διότι θα δημιουργήσουμε δυναμικούς πίνακες για να κρατάμε τους αντίστοιχους χρόνους, 1 μεταβλητή για τον σπόρο, 4 mutexes(για φούρνους, μάγειρες,διανομείς, οθόνη) και 3 condition variables(για ενεργοποίηση παραγγελίας που περιμένει φούρνο ή μάγειρα ή διανομέα).

Στη main συνάρτηση ελέγχουμε αν τα ορίσματα που έχει δώσει ο χρήστης είναι 3 και αν ο αριθμός των πελατών είναι θετικός ακέραιος. Εκχωρούμε τις τιμές εφόσον είναι σωστές, δημιουργούμε τον πίνακα των threads και κάνουμε initialize τα mutexes και τα condition variables. Ορίζουμε το μέγεθος των δυναμικών πινάκων βάσει του αριθμού των πελατών και δημιουργούμε τα νήματα με παράμετρο την συνάρτηση order και τον αριθμό της παραγγελίας(id[i] = i + 1, ώστε να ξεκινάει ο αριθμός από 1). Προηγουμένως βάζουμε το ίδιο for να ΜΗΝ δημιουργεί ακαριαία όλες τις παραγγελίες μαζί αλλά, πέρα από την 1η που δεν έχει καμία καθυστέρηση, όλες οι υπόλοιπες περιμένουν για ένα τυχαίο διάστημα από [1,5](χρησιμοποιώντας την (τοπική) μεταβλητή seed1 = seed για να μην κλειδώνουμε το mutex)(το seed είναι global και περιέχει την τιμή που δίνει ο χρήστης). Ύστερα περιμένουμε όλα τα νήματα να τελειώσουν και τρέχουμε ένα for loop όσος και ο αριθμός των πελατών για να βγάλουμε τα συγκεντρωτικά αποτελέσματα για τον μέσο και μέγιστο χρόνο ολοκλήρωσης των παραγγελιών, αλλά και για τον μέσο και μέγιστο χρόνο κρυώματος των παραγγελιών. Τέλος, κλειδώνουμε την οθόνη και εμφανίζουμε τα αντίστοιχα μηνύματα, απελευθερώνουμε τον χώρο που έχουμε δεσμεύσει δυναμικά και καταστρέφουμε τα mutexes και τα condition variables.

Στην συνάρτηση order ορίζουμε το start, το finish και το startfreezing προκειμένου να γνωρίζουμε την στιγμή που άρχισε και τελείωσε η εκάστοτε παραγγελία(και τη στιγμή που βγήκε από τον φούρνο και άρχισε να κρυώνει) και κλειδώνουμε τους μάγειρες. Όσο δεν υπάρχει κανένας μάγειρας εμφανίζουμε μήνυμα ότι η παραγγελία είναι μπλοκαρισμένη και το βάζουμε να περιμένει. Όταν βγει από το while loop τότε σημαίνει ότι βρήκε διαθέσιμο μάγειρα, μειώνουμε λοιπόν τους μάγειρες και ξεκλειδώνουμε το mutex. Χρησιμοποιούμε μία νέα τοπική μεταβλητή το seed2 στο οποίο εκχωρούμε το seed + id(για μεγαλύτερη τυχαιότητα) ώστε να πάρουμε ένα τυχαίο αριθμό, και το βάζουμε να "πέσει για ύπνο" για ένα τυχαίο χρονικό διάστημα από 1-5 (βάσει του τυχαίου αριθμού). Μετά κάνουμε για τους φούρνους ότι ακριβώς και για τους μάγειρες(όσον αφορά το κλείδωμα και την μείωση όταν βγει από το while loop). Στη συνέχεια, αποδεσμεύουμε τον μάγειρα(αφού την παραγγελία θα τη βγάλει απο τον φούρνο ο διανομέας),κλειδώνουμε δηλαδή τους μάγειρες, τους αυξάνουμε κατά έναν, κάνουμε signal σε περίπτωση που κάποια παραγγελία περιμένει παρασκευαστή και ύστερα τους ξεκλειδώνουμε, και μετά το βάζουμε "για ύπνο" για χρόνο Tbake. Εφόσον περάσει από το sleep παίρνουμε την αντίστοιχη χρονική στιγμή (δηλαδη το σημείο όπου ξεκινάει να κρυώνει η πίτσα), ενω ταυτόχρονα κλειδώνουμε τους διανομείς. Όσο δεν υπάρχει κανένας διανομέας εμφανίζουμε μήνυμα ότι η παραγγελία κρυώνει στο φούρνο και παράλληλα ότι αναμένεται διανομέας και το βάζουμε να περιμένει.Όταν βγει από το while loop σημαίνει ότι βρήκε διαθέσιμο διανομέα, τότε μειώνουμε τον αριθμό τους κατά 1 και ξεκλειδώνουμε το mutex.Ύστερα κλειδώνουμε τους φούρνους ,τους αυξάνουμε κατά 1 και κάνουμε signal σε περίπτωση που κάποια παραγγελία περιμένει φούρνο, αφού είναι

σίγουρο πως νωρίτερα ένας διανομέας έχει απελευθερώσει ένα φούρνο. Επιπλέον, το βάζουμε για ύπνο για ένα τυχαίο χρονικό διάστημα από 5 έως 15(χρησιμοποιώντας για μεγαλύτερη τυχαιότητα μία τρίτη τοπική μεταβλητή seed3 = seed - id) (που αντικατοπτρίζει την διάρκεια του διανομέα μέχρι να παραδώσει την παραγγελία στο εκάστοτε σπίτι) και το αποθηκεύουμε σε μία τοπική μεταβλητή (deliverer_return) ώστε να κρατήσουμε το τυχαίο διάστημα ίδιο και για την επιστροφή του διανομέα προς την πιτσαρία.Μετά το sleep, κλειδώνουμε την οθόνη και παίρνουμε τον χρόνο ώστε να βγάλουμε την συνολική διάρκεια της παραγγελίας και τον συνολικό χρόνο κρυώματος προκειμένου να την εκχωρήσουμε στο αντίστοιχο κελί του timer και του timer2(id - 1 εφόσον στη main περνούσαμε σαν όρισμα το id + 1), για να βγάλουμε στη main τον μέσο και μέγιστο χρόνο ολοκλήρωσης, καθώς και τον μέσο και μέγιστο χρόνο κρυώματος. Ύστερα, εμφανίζουμε το μήνυμα:"Η παρραγελία με αριθμό <oid> παραδόθηκε σε <X> λεπτά και κρύωνε για <Y> λεπτά", αποδεσμεύουμε την οθόνη και το βάζουμε "για ύπνο", μέχρι να επιστρέψει ο διανομέας στην πιτσαρία. Τέλος, κλειδώνουμε τους διανομείς, τους αυξάνουμε κατά 1, κάνουμε signal σε περίπτωση που κάποια παραγγελία περιμένει διανομέα, ξεκλειδώνουμε τους διανομείς και βγαίνουμε από το thread χωρίς να επιστρέφουμε κάτι.