

ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ  
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ  
Λειτουργικά Συστήματα  
Εαρινό Εξάμηνο 2019-2020

Ελένη Σαζώνη, Α.Μ: 3160153  
Αγγελική Φέκα, Α.Μ: 3140290  
Αριστείδης Χρονόπουλος, Α.Μ: 3160194

## 2η Προγραμματιστική Εργασία

### 1. Δομή πηγαίου κώδικα:

- Αρχικά καλείται η μέθοδος **main(int argc, char \*argv[ ])**, η οποία παίρνει ως όρισμα από τον χρήστη τον αριθμό των πελατών και τον σπόρο (seed) για την **rand\_r**. Με **rand\_r** παράγουμε τυχαία τον χρόνο που δεχόμαστε κάθε νέα παραγγελία και τον αριθμό από πίτσες που περιέχει η κάθε παραγγελία. Στην περίπτωση που ο χρήστης δεν δώσει τον απαραίτητο αριθμό ορισμάτων, του εμφανίζεται το κατάλληλο μήνυμα σφάλματος.
- Ανάλογα με τον αριθμό των πελατών δημιουργούμε και τα αντίστοιχα νήματα με την χρήση της **pthread\_create**.
- Επίσης, σύμφωνα με τον αριθμό των πελατών δημιουργούμε και τρεις(3) πίνακες με τον ανάλογο μέγεθος προκειμένου να αποθηκεύσουμε σε αυτούς τους τα απαραίτητα δεδομένα για την εργασία μας. Συγκεκριμένα δημιουργούμε έναν πίνακα για τις παραγγελίες, έναν πίνακα για να αποθηκεύουμε τον χρόνο παράδοσης των παραγγελιών και έναν πίνακα για να αποθηκεύσουμε τον χρόνο κρυώματος των παραγγελιών.
- Με την **pthread\_create** καλείται η **execute(void \* customer\_id)** για κάθε νήμα που δημιουργούμε, η οποία υλοποιεί τα εξής:
  - Δέχεται κάθε παραγγελία μετά από ένα τυχαίο χρόνο **accept\_delay\_time** και κρατά την χρονική στιγμή που δέχεται την παραγγελία στην μεταβλητή **order\_start** με την χρήση της **clock\_gettime**.
  - Κρατάμε τον χρόνο όταν μια παραγγελία παραδίδεται σε κάποιον παρασκευαστή στην μεταβλητή **prep\_start**.
  - Υπολογίζει τυχαία με την χρήση της **rand\_r** των αριθμό των πιτσών που θα έχει η κάθε παραγγελία και περιμένει **T\_prep** για κάθε μια από αυτές.
  - Ύστερα όλες οι πίτσες ψήνονται όλες μαζί εφόσον έχει βρεθεί διαθέσιμος φούρνος και περιμένουμε για χρόνο **T\_bake** μέχρι να ολοκληρωθεί το ψήσιμο.
  - Εφόσον ολοκληρωθεί το ψήσιμο ο πρώτος διαθέσιμος delivery βγάζει την παραγγελία από τον φούρνο και ξεκινάει για να την παραδώσει. Ο χρόνος που χρειάζεται ο delivery για την παράδοση της παραγγελίας στο σπίτι υπολογίζεται τυχαία με την χρήση της **rand\_r**.
  - Τέλος, υπολογίζεται ο χρόνος που χρειάζεται από την εισαγωγή μιας παραγγελίας μέχρι την παράδοσή της στον πελάτη, όπως και ο χρόνος που

χρειάζεται από την ολοκλήρωση του ψησίματος της παραγγελίας μέχρι και την παράδοση της στον πελάτη. Οι χρόνοι αυτοί αποθηκεύονται στις μεταβλητές **time\_X** και **time\_Y** αντίστοιχα.

- Όταν τα νήματα ολοκληρώνονται και έλεγχος επιστρέφει στην *main* διασχίζουμε του πίνακες που δημιουργήσαμε έτσι ώστε να βγάλουμε τα τελικά αποτελέσματα. Συγκεκριμένα, στους πίνακες *max\_delivery* και *max\_cool*, τους οποίους διαχειριζόμαστε μέσω *global* μεταβλητής τύπου *double\**, και όταν ο έλεγχος επιστρέφει στην *main* υπολογίζουμε την μέγιστη τιμή χρόνου παράδοσης παραγγελιών και την μέγιστη τιμή χρόνου κρυώματος αντίστοιχα.

## 2. Περιγραφή Υλοποίησης Μεθόδων:

- Για τη δημιουργία των νημάτων κάναμε χρήση της `int pthread_create(pthread_t * thread, const pthread_attr_t * attr, void *(* start_routine ) (void *), void * arg );` Το νέο thread που δημιουργείται ξεκινά εκτελώντας την μέθοδο που καλείται ως `start_routine()`.
- Προκειμένου να πετύχουμε τον συγχρονισμό των νημάτων χρησιμοποιήσαμε *mutexes*. Ουσιαστικά ένα *mutex* είναι ένα *lock*, το οποίο θέτουμε πριν την χρήση των κοινών δεδομένων των threads, και το ελευθερώνουμε εφόσον τελειώσουμε με την χρήση αυτή. Συγκεκριμένα χρησιμοποιούμε τα *mutex* **mutex\_available\_cooks**, **mutex\_available\_ovens**, **mutex\_available\_deliverer**, **mutex\_count**, **mutex\_console**, **mutex\_total\_order\_completion**, **mutex\_sum** και **mutex\_avg** πριν από κάθε υπολογισμό των κοινών δεδομένων, δηλαδή των μετρητών **available\_cooks** (μετρητής για τον αριθμό των διαθέσιμων cooks κάθε χρονική στιγμή), **available\_ovens** (μετρητής για τον αριθμό των διαθέσιμων φούρνων κάθε χρονική στιγμή), **available\_deliverer** (μετρητής για τον αριθμό των διαθέσιμων deliverer), **count** (μετρητής για τον αριθμό των παραγγελιών) και πριν από τον υπολογισμό του συνολικού χρόνου προετοιμασίας της παραγγελίας. Κάθε φορά που υπολογίζουμε έναν από αυτούς τους μετρητές χρησιμοποιούμε **pthread\_mutex\_lock** προκειμένου να κλειδώσουμε το επιθυμητό *mutex* έτσι ώστε να μην είναι προσβάσιμο εκείνο το σημείο του κώδικα στα υπάρχοντα threads. Όταν τελειώσουμε με τους υπολογισμούς μας χρησιμοποιούμε **pthread\_mutex\_unlock** έτσι ώστε να διαθέσουμε τους μετρητές με τις νέες τιμές στα threads μας.
- Με την χρήση της `int pthread_join(pthread_t thread, void **retval);` περιμένει το συγκεκριμένο thread που δέχεται σαν όρισμα να τελειώσει την εκτέλεση του. Αν το *retval* δεν είναι *NULL*, τότε η `pthread_join` αντιγράφει το *exit status* του thread στην θέση που υποδεικνύει η *\*retval*. Στην περίπτωση μας κάθε thread επιστρέφει τον χρόνο ολοκλήρωσης του.

### 3. Αποτελέσματα:

```
helena@rogue-one: ~/Desktop/OS
Order with id:57 has been delivered in 117.000000 seconds and cooled off for 8.000000 seconds
Order with id:66 has been delivered in 121.000000 seconds and cooled off for 5.000000 seconds
Order with id:83 has been delivered in 124.000000 seconds and cooled off for 5.000000 seconds
Order with id:74 has been delivered in 123.000000 seconds and cooled off for 8.000000 seconds
Order with id:85 has been delivered in 129.000000 seconds and cooled off for 8.000000 seconds
Order with id:98 has been delivered in 131.000000 seconds and cooled off for 7.000000 seconds
Order with id:13 has been delivered in 131.000000 seconds and cooled off for 6.000000 seconds
Order with id:91 has been delivered in 133.000000 seconds and cooled off for 8.000000 seconds
Order with id:25 has been delivered in 137.000000 seconds and cooled off for 7.000000 seconds
Order with id:21 has been delivered in 138.000000 seconds and cooled off for 10.000000 seconds
Order with id:63 has been delivered in 142.000000 seconds and cooled off for 6.000000 seconds
Order with id:34 has been delivered in 144.000000 seconds and cooled off for 6.000000 seconds
Order with id:33 has been delivered in 143.000000 seconds and cooled off for 10.000000 seconds
Order with id:26 has been delivered in 144.000000 seconds and cooled off for 10.000000 seconds
Order with id:48 has been delivered in 149.000000 seconds and cooled off for 5.000000 seconds
Order with id:39 has been delivered in 150.000000 seconds and cooled off for 7.000000 seconds
Order with id:37 has been delivered in 149.000000 seconds and cooled off for 9.000000 seconds
Order with id:49 has been delivered in 155.000000 seconds and cooled off for 5.000000 seconds
Order with id:63 has been delivered in 154.000000 seconds and cooled off for 6.000000 seconds
Order with id:65 has been delivered in 158.000000 seconds and cooled off for 5.000000 seconds
Order with id:42 has been delivered in 155.000000 seconds and cooled off for 9.000000 seconds
Order with id:69 has been delivered in 161.000000 seconds and cooled off for 7.000000 seconds
Order with id:68 has been delivered in 164.000000 seconds and cooled off for 6.000000 seconds
Order with id:73 has been delivered in 166.000000 seconds and cooled off for 6.000000 seconds
Order with id:72 has been delivered in 165.000000 seconds and cooled off for 9.000000 seconds
Order with id:100 has been delivered in 172.000000 seconds and cooled off for 6.000000 seconds
Order with id:3 has been delivered in 172.000000 seconds and cooled off for 8.000000 seconds
Order with id:4 has been delivered in 174.000000 seconds and cooled off for 7.000000 seconds
Order with id:76 has been delivered in 173.000000 seconds and cooled off for 10.000000 seconds
Order with id:28 has been delivered in 179.000000 seconds and cooled off for 7.000000 seconds
Order with id:5 has been delivered in 179.000000 seconds and cooled off for 10.000000 seconds
Order with id:45 has been delivered in 185.000000 seconds and cooled off for 8.000000 seconds
Order with id:22 has been delivered in 184.000000 seconds and cooled off for 10.000000 seconds
Order with id:32 has been delivered in 185.000000 seconds and cooled off for 10.000000 seconds
Order with id:46 has been delivered in 187.000000 seconds and cooled off for 8.000000 seconds
Order with id:67 has been delivered in 190.000000 seconds and cooled off for 5.000000 seconds
Order with id:52 has been delivered in 191.000000 seconds and cooled off for 7.000000 seconds
Order with id:75 has been delivered in 194.000000 seconds and cooled off for 6.000000 seconds
Order with id:70 has been delivered in 195.000000 seconds and cooled off for 6.000000 seconds
Order with id:58 has been delivered in 192.000000 seconds and cooled off for 10.000000 seconds
Order with id:80 has been delivered in 199.000000 seconds and cooled off for 7.000000 seconds
Order with id:84 has been delivered in 203.000000 seconds and cooled off for 9.000000 seconds
Order with id:90 has been delivered in 204.000000 seconds and cooled off for 8.000000 seconds
Order with id:89 has been delivered in 206.000000 seconds and cooled off for 8.000000 seconds
Order with id:96 has been delivered in 208.000000 seconds and cooled off for 6.000000 seconds
Order with id:97 has been delivered in 211.000000 seconds and cooled off for 7.000000 seconds
Order with id:93 has been delivered in 209.000000 seconds and cooled off for 10.000000 seconds
Order with id:92 has been delivered in 213.000000 seconds and cooled off for 7.000000 seconds
Average time for an order to be delivered: 114.970000
Average time for an order to cool: 7.380000
Maximum delivery duration was: 213.000000
Maximum cooling duration of an order was: 10.000000
(base) helena@rogue-one:~/Desktop/OS$
```