ΛΕΙΤΟΥΡΓΙΚΑ ΣΥΣΤΗΜΑΤΑ

Δεύτερη Εργασία
2018

ΔΗΜΗΤΡΙΟΣ ΣΤΕΦΑΝΟΥ
Α.Μ. 3160245
dimitriosstefanou97@gmail.com

ΕΛΒΙΣ ΣΕΧΟΥ
Α.Μ. 3160156
shehu314@hotmail.com

ΙΩΑΝΝΗΣ ΠΑΠΑΓΕΩΡΓΙΟΥ
Α.Μ. 3160129
giannhspap24@hotmail.com

Περιεχόμενα

Header και common files

Το header file της εργασίας περιέχει τις έτοιμες συναρτήσεις που δόθηκαν από το e-class συμπεριλαμβανομένων κάποιων παραπάνω που χρησιμοποιούνται από τα 3 προγράμματα που ζητούνται.

Οι υλοποιήσεις αυτών γίνονται στο αρχείο common.

Εκτός των έτοιμων functions που δίνονται από το e-class, στο common εμπεριέχονται και οι εξής συναρτήσεις:

- **isFull :** ελέγχει εάν ο buffer είναι πλήρης
- **isEmpty :** ελέγχει εάν ο buffer είναι άδειος
- **openFile :** δημιουργεί ή κάνει overwrite ήδη υπάρχοντα txt αρχεία στα οποία αποθηκεύονται οι έξοδοι των producer και consumer threads.
- **to_txt_out :** ανοίγει τα υπάρχοντα αρχεία και καταγράφει απευθείας κάθε αριθμό που δημιουργεί ένας producer και κάθε αριθμό που αφαιρεί ενας consumer από τον buffer.
- **print_2d_array:** τυπώνει όλους τους αριθμούς που έχει χρησιμοποιήσει κάθε thread είτε από consumers είτε από producers σε αύξουσα σειρά

Prodcons1

Σχετικά με την συνάρτηση **producer:**

Οι γενικές πληροφορίες/μεταβλητές καθώς και το seed των producers αποθηκεύονται σε struct.

Αρχικά, το producer thread τρέχει σε επανάληψη για την δημιουργία των αριθμών (το πλήθος τους δίνεται από τον χρήστη), ελέγχοντας πάντα εάν ο buffer είναι γεμάτος. Σε περίπτωση που γεμίσει, το thread μπαίνει σε κατάσταση αναμονής (wait) έως ότου το consumer thread στείλει σήμα (signal) ότι ο buffer είναι empty. Στον buffer εισάγονται τυχαίοι αριθμοί μέσω της rand_r οι οποίες αποθηκεύονται αμέσως μόλις δημιουργηθούν στο prod_in.txt
Μετά την επανάληψη, το εκάστοτε producer thread θα εξέλθει από την συνάρτηση και θα επιστρέψει στη main όπου θα κάνει join στο main thread. Υπάρχει int (που λειτούργει σαν boolean) τιμή, η οποία ενημερώνει ότι ο producer τελείωσε επιτυχώς.

Σχετικά με την συνάρτηση **consumer:**

Μαζί με τον producer καλείται αντίστοιχα και ένα consumer thread. Τρέχει παράλληλα ώστε να αδειάζει τον buffer από τους αριθμούς που εισάγει ο producer.
Εάν ο buffer βρεθεί άδειος, καλείται μία wait και το consumer thread μπαίνει σε αναμονή, έως ότου στείλει signal ο producer ότι ο buffer δεν είναι πια empty.
Ο consumer αφαιρεί αριθμούς από την ουρά (με pop) το οποίο ταυτόχρονα εγγράφεται στο txt αρχείο (out_cons.txt).
Το thread θα βγει από τον ατέρμων βρόχο (με break) όταν η int μεταβλητή ελέγχου από τον producer γίνει 1 και ο buffer αδειάσει (isEmpty == true). Το thread θα εξέλθει και θα επιστρέψει στο main thread όπου θα κάνει και join.

Prodcons2

Το prodcons2 λειτουργεί όμοια με το πρώτο πρόγραμμα, αλλά με μεταβλητό αριθμό από threads.
Για κάθε εισαγωγή ή εξαγωγή από την ουρά μπορεί να λειτουργεί μόνο ένα thread. Για αυτόν τον σκοπό χρησιμοποιούμε mutexes που μπλοκάρουν τα υπόλοιπα όμοια threads αποτρέποντάς τα από το να λειτουργούν ταυτόχρονα.
Για τον έλεγχο του τελειώματος της εργασίας των producers χρησιμοποιείται μία μεταβλητή μέσα στο struct με τις υπόλοιπες πληροφορίες τους.
Η check_prod_threads ελέγχει εάν όλοι οι producers έχουν τερματίσει.

Στην περίπτωση που τερματίσουν οι producers ενώ υπάρχουν ακόμη consumers σε αναμονή τότε μέσω timewait() τα threads αυτά αυτοκτονούν. Όσο για το output,για τα δύο txt αρχεία ,απλά καλούμε την to_txt_out στις *producer, *consumer , ώστε κάθε φορά που δρα ένας από αυτούς να μπαίνει στο ανάλογο txt αυτή η ενέργεια με το ανάλογο id και τον αριθμό. Έτσι λοιπόν στα δυο αρχεία έχουμε όλα τις ενέργειες με την σειρά που πραγματοποιήθηκαν. Ενώ για το stdout ,κάνουμε το εξής:

Prodcons3

Το prodcons3 παρομοίως εμπεριέχει τις λειτουργίες των προηγούμενων προγραμμάτων, με την διαφορά ότι το std_out είναι ταξινομημένο.
Το σορτάρισμα επιτυγχάνεται μέσω της δημιουργίας 2 διπλών δυναμικών πινάκων, όπου οι γραμμές αντιστοιχούν αριθμητικά σε κάθε thread, και οι στήλες είναι για τους consumers (όσα τα threads producers * τους αριθμούς που μπορεί να δημιουργήσει ένας παραγωγός αργκ :4,επειδή είναι το μαξ που μπορεί να τραβήξει ενας consumer) , ένω για τους producers απλά όσοι αριθμοί μπορεί ν φτιάξει ο ένας ,με έναν πίνακα για τους producers και έναν για τους consumers αντιστοίχως. Έτσι, με μία απλή προσπέλαση του μπορούμε να τυπώσουμε σε αύξουσα σειρά τα threads με τους αριθμούς που χρησιμοποίησαν. Και προφανώς λειτουργεί , όπως ακριβώς το prodcons2 στο input.