Ονοματεπώνυμο: *Θεοφανόπουλος Μιχαήλ*
Αριθμός Μητρώου: *1115201800053*
3η Εργασία Λειτουργικά Συστήματα 2020-2021

Αρχικά για να μεταγλωττιστεί το πρόγραμμα πρέπει να δωθεί η εντολή:
>>make all
Η μεταγλώττιση γίνεται με τη χρήση ενός makefile το οποίο χρησιμοποιεί separate compilation για την μεταγλώττιση του προγράμματος. Για να τρέξει το πρόγραμμα πρέπει να εκτελεστούν τα εκτελέσιμα αρχεία (chef,saladmaker) τα οποία χρειάζονται τις παραμέτρους που αναφέρονται στην εκφώνηση, καθώς και μια ακόμα παράμετρο στο saladmaker την -o (1,2,3) η οποία υποδηλώνει το ποιος από τους τρείς σαλατοποιούς θέλουμε να αναπαριστά το saladmaker.cpp στην δεδομένη κλήση.

Το πρόγραμμα περιέχει 10 αρχεία:

1. *Το chef.cpp, το οποίο αναπαριστά τον σεφ της διαδικασίας.*
2. *Το chef.hpp, το οποίο περιέχει τις βιβλιοθήκες που χρειάζεται ο κώδικας του σεφ για να τρέξει και διάφορα define που θα χρειαστεί στην υλοποίησή του. Τα messages που έχουν γίνει defined είναι ουσιαστικά τα messages που απαιτούνται να γράφει ο κάθε σεφ κατά την κλήση του, σύμφωνα με τη μορφοποίση εξόδου της άσκησης που υπάρχει στην σελίδα του μαθήματος.*
3. *Το saladmaker.cpp το οποίο αναπαριστά τους σαλατοποιούς της διαδικασίας. Ανάλογα με τα arguments που δίνονται κατά την κλήση του προγράμματος, κάθε εκτέλεση έχει τη δυνατότητα να αναπαριστά διαφορετικό σαλατοποιό και να έχει διαφορετικό cooking time.*
4. *Το saladmaker.hpp, το οποίο περιέχει τις βιβλιοθήκες που χρειάζεται ο κώδικας του σαλατοποιού για να τρέξει και διάφορα define που θα χρειαστεί στην υλοποίησή του. Τα messages που έχουν γίνει defined είναι ουσιαστικά τα messages που απαιτούνται να γράφει ο κάθε σαλατοποιός κατά την κλήση του, σύμφωνα με τη μορφοποίηση εξόδου της άσκησης που υπάρχει στην σελίδα του μαθήματος.*
5. *Το list.cpp, το οποίο αναπαριστά μια απλή λίστα με περιεχόμενο ένα string και χρησιμοποιείται αποκλειστικά για την εκτύπωση διαστημάτων στα οποία δύο ή περισσότεροι σαλατοποιοί μαγείρευαν ταυτοχρόνως.*
6. *Το list.hpp, το οποίο περιέχει τις βιβλιοθήκες που χρειάζεται ο κώδικας της λίστας για να τρέξει καθώς και την κλάση List.*
7. *Το functions.cpp, το οποίο περιέχει ορισμένες συναρτήσεις γενικής χρήσεως.*
8. *Το functions.hpp το οποίο περιέχει τις βιβλιοθήκες που χρειάζονται οι συναρτήσεις αυτές.*
9. *Τα logfile.cpp και logfile.hpp. Τα δεδομένο αρχεία επέλεξα να υπάρχουν ώστε να ικανοποιήσω την απαίτηση της ύπαρξης ενός γενικού log file στο οποίο θα γράφουν τα μηνύματα όλες οι διεργασίες. Η κλάση καθώς και οι συναρτήσεις της που περιέχονται στα αρχεία αυτά στην ουσία έχουν ένα ofstream, ένα id και μία λίστα. Κάθε φορά που μια διεργασία επιθυμεί να γράψει στο log file, καλεί την write της κλάσης αυτής (φυσικά θα πρέπει να έχει δημιουργήσει αντικείμενο της παραπάνω κλάσης ήδη) και η ίδια η συνάρτηση ανοίγει το αρχείο, γράφει το μήνυμα και στο τέλος το κλείνει. Ακολούθησα την συγκεκριμένη υλοποίηση καθώς είδα ότι γενικά υπάρχει θέμα στο να ανοίγουν ένα αρχείο πολλαπλές διεργασίες για γράψιμο, οπότε με τον παραπάνω τρόπο κάθε φορά ανοίγει και κλείνει το αρχείο μοναδική φορά ανά μήνυμα και αυτό με την βοήθεια της κλάσης Logfile. Εκτός από τις παραπάνω λειτουργίες, η δεδομένη*

κλάση υλοποιεί και την λειτουργία εύρεσης κοινών διαστημάτων, πάνω πάντα στο ενιαίο και μοναδικό logfile. Να σημειώσω πως το logfile κάθε φορα παίρνει την μορφη log(CHEF_ID).txt με σκοπό να δημιουργείται ξεχωριστό αρχείο για κάθε εκτέλεση. Η ίδια λογική ακολουθείται και για κάθε άλλο αρχείο που δημιουργείται στο πρόγραμμα.

Επέλεξα να υλοποιήσω την εργασία σε C++ καθώς θεώρησα χρήσιμες ορισμένες λειτουργίες της που δεν υπάρχουν στην C, όπως την κλάση string, την ύπαρξη των user-made κλάσεων και τα stringstream.

Το πρόγραμμα αποδεσμεύει σωστά όλη τη μνήμη που χρησιμοποιεί, αφαιρεί το shared memory segment και διαγράφει τους σημαφόρους. Ως αποτέλεσμα, πάντα φτιάχνονται το πολύ +2 έξτρα σαλάτες, λόγω ασυγχρονισμού των διαδικασιών. Στο αρχείο chef.cpp πριν τον τερματισμό του προγράμματος κάνω ένα sleep(5) με σκοπό να δωθεί λίγος έξτρα χρόνος στους σαλατοποιούς να τερματίσουν ώστε να εκτυπωθούν τα σωστά αποτελέσματα.

Επέλεξα την υλοποίηση της άσκησης με Named Semaphores, καθώς δεν ήθελα να χρειαστεί να περάσω τους σημαφόρους στο shared memory segment και θεώρησα πιο σωστό οι σημαφόροι να είναι αποθηκευμένοι στη μνήμη και όποτε και όποιος θέλει να μπορεί ανά πάσα στιγμή να έχει πρόσβαση σε αυτούς.

Υλοποίησα την εύρεση κοινών διαστημάτων με την χρήση δύο συναρτήσεων, της findConcurrent και της quickFind. Οι συναρτήσεις αυτές κάνουν έλεγχο των ψηφίων του id του κάθε process, καθώς το πρόγραμμα μου έβγαζε διαφορετικά και ενδεχομένως λανθασμένα αποτελέσματα ανάλογα με το μέγεθος του integer που αναπαριστά το id μέσα στο αρχείο. Για τον λόγο αυτό και επειδή δεν κατάφερα να διαχειριστώ σωστά την συνάρτηση string::find(), επέλεξα να χειρίζομαι με διαφορετικό τρόπο τα strings που περιέχονται μέσα στο αρχείο ανάλογα το μέγεθος του id. Έλαβα επίσης υπόψιν μου το παρακάτω άρθρο το οποίο αναφέρει πώς η ελάχιστη τιμή που το λειτουργικό σύστημα μπορεί να δώσει σε μία εργασία για procces id είναι το 1 και η μέγιστη το 2^15 = 32768.
https://serverfault.com/questions/279178/what-is-the-range-of-a-pid-on-linux-and-solaris

Στο shared memory segment περιέχονται τα παρακάτω segments:

- *Αριθμός σαλατών που πρέπει να κατασκευαστούν.*
- *Αριθμός σαλατών που έχει κατασκευάσει ο πρώτος σαλατοποιός, το δεδομένο segment προβλέπεται να το ανανεώνει κάθε φορά το πρόγραμμα του saladmaker κατά την εκτέλεση του.*
- *Αριθμός σαλατών που έχει κατασκευάσει ο δεύτερος σαλατοποιός, το δεδομένο segment προβλέπεται να το ανανεώνει κάθε φορά το πρόγραμμα του saladmaker κατά την εκτέλεση του.*
- *Αριθμός σαλατών που έχει κατασκευάσει ο τρίτος σαλατοποιός, το δεδομένο segment προβλέπεται να το ανανεώνει κάθε φορά το πρόγραμμα του saladmaker κατά την εκτέλεση του.*
- *Process id της διαδικασίας του chef. Το segment αυτό είναι χρήσιμο κατά την κλήση της logfile::write για να γνωρίζει η κλάση ποιο αρχείο να ανοίξει ώστε να εκτελέσει της λειτουργίες της. Όπως είπαμε προηγουμένως, τα ονόματα των logfiles περιέχουν πάντα το ID της διαδικασίας η οποία γράφει στο δεδομένο αρχείο και στην περίπτωση του ενιαίου logfile περιέχει το ID της διαδικασίας του chef.*

Για την μορφοποίηση του χρόνου όπως απαιτείται στην μορφοποίηση εξόδου, χρησιμοποιώ τις έτοιμες συναρτήσεις gettimeofday(), time() , localtime(). Επίσης κάνω χρήση το stringstream με σκοπό να περιορίζω τον αριθμό ψηφίων σε 2 και να κάνω fill τους αριθμούς με μοναδικό ψηφίο με μηδέν.

Καλη διόρθωση!!