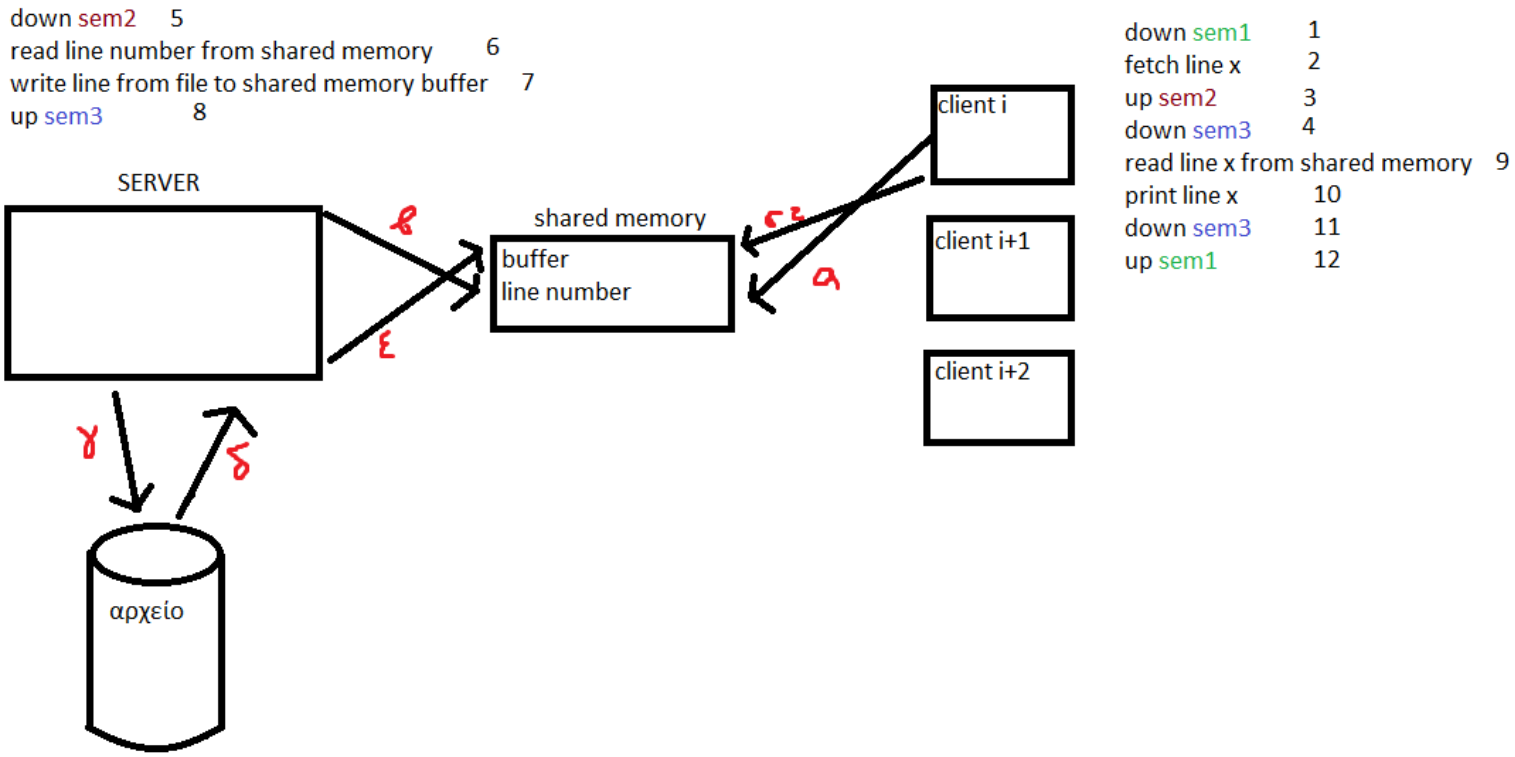

Έστω ότι ο **client I** είναι ο πρώτος client που κάνει αίτημα για μία γραμμή **x**. Ο **client I** αρχικά κατεβάζει τον σεμαφορο **sem1** που είναι αρχικοποιημένος με την τιμή **1**, ώστε να μην μπορεί άλλος client να έχει πρόσβαση στην κρίσιμη περιοχή.

Δηλαδή αν δεν υπήρχε ο **sem1** και εκείνη την στιγμή ζητούσε κάποια γραμμή από τον server και ο **client i+1,** θα έγραφε στην shared memory την γραμμή που ήθελε αυτός και έτσι ο server θα επέστρεφε λάθος γραμμή στον **client i**.

Μετέπειτα γράφει στην shared memory στην μεταβλητή **line number** ποια γραμμή θέλει να πάρει και ανεβάζει τον σεμαφόρο **sem2**.Δουλειά του **sem2** είναι να εξασφαλίσει ότι ο server μπαίνει στο critical section, αφού πρώτα ο client κάνει το αίτημα του. Ύστερα ο **client I** κάνει down τον σεμαφόρο **sem3** και περιμένει την απόκριση του server.

Ο server με την σειρά του διαβάζει από την shared memory ποια γραμμή ζήτησε ο client, την ανακτά από το αρχείο και την γράφει στον **buffer** του shared memory.Τέλος ανεβάζει τον σεμαφόρο **sem3** για να πεί στον client ότι μπορεί να συνεχίσει. Ο client διαβάζει την γραμμή από την shared memory, την εκτυπώνει και ανεβάζει τον **sem1** αφού έχει τελείωσει την δουλειά του.

Sem1: Ο sem1 εξασφαλίζει ότι όταν γράφει στο critical section(δηλαδή στην shared memory) κάποιος client, δεν μπορεί να γράψει κανένας άλλος client μέχρι να τελειώσει την δουλεία του ο client

Sem2: Ο sem2 εξασφαλίζει ότι ο server διαβάζει από την shared memory, **μόνο** μετά την αίτηση του client. Αρχικοποιείται λοιπόν σε 0 και μόλις ο client τον κάνει 1 τότε είναι η σειρά του server να εκτελέσει για να επιστρέψει μια απόκριση στον client

Sem3: Ο sem3 εξασφαλίζει ότι ο client θα διαβάσει από την shared memory **μόνο** όταν ο server έχει γράψει το μήνυμα απόκρισης εκεί. Για αυτό αρχικοποιείται με τιμή 0. Μόλις τον κάνει down ο client περιμένει από τον server να τον κάνει up ώστε να καταλάβει ότι ο server έχει τελείωσει την δουλεία του και μπορεί να προχωρήσει με την εκτύπωση της γραμμής που αιτήθηκε.

Δομή project

Στον φάκελο include έχω όλες της βιβλιοθήκες που έχω φτιάξει.

Στο sem_functions.h είναι γραμμένες συναρτήσεις που διαχειρίζονται σεμαφόρους. Ο ορισμός και η δουλεία της κάθε συνάρτησης περιγράφεται εκεί.

Στο general.h έχω μια συνάρτηση που διαβάζει την γραμμή που υποδεικνύεται και την επιστρέφει.

Στον φάκελο modules έχω τις υλοποιήσεις των συναρτήσεων που υπάρχουν στον φάκελο include.

Main program:

Για να δημιουργήσω ακριβώς K clients που υποδεικνύονται από την command line κατά την εκτέλεση του προγράμματος, έχω μια for που τρέχει από το i=0 έως i=K και κάνω fork το main program. Μέσα στην for ελέγχω αν το pid που επιστράφηκε είναι 0 – δηλαδή αν είμαι στην διεργασία παιδί- και κάνω break.

Κάθε διεργασία παιδί θα κάνει N δοσοληψίες, δηλαδή N αιτήσεις.

Αφού υπάρχουν K clients θα γίνουν συνολικά K*N αιτήσεις στον server.

Συνεπώς ο server για αριθμό επαναλήψεων K*N θα στέλνει αποκρίσεις. Για να μην κάνει busy waiting ο server χρησιμοποιείται η διαδικασία που περιεγράφηκε παραπάνω.

Έχω βάλει οι printf να εκτυπώνει με χρώμα(για λογούς διευκόλυνσης) ανάλογα τον τύπο της εκτύπωσης.

Κόκκινο -> ο μέσος χρόνος του process από την υποβολή ενός αιτήματος μέχρι την λήψη κάποιας απάντησης

Κίτρινο -> Η αντίστοιχη γραμμή που εκτυπώνεται

Πράσινο->Οι αιτήσεις από τον client και οι απαντήσεις

Άσπρο -> Γενικές εκτυπώσεις όσον αφορά τους σεμαφόρους κτλπ για τον έλεγχο ορθότητας του προγράμματος.

Σημείωση: Έχω ήδη ένα έτοιμο αρχείο txt στον φάκελο programs.

Σε περίπτωση που ο χρήστης θελήσει να βάλει το δικό του, θα πρέπει να το βάλει στον φάκελο programs.