Η εργασία αποτελείται απο τα παρακάτω αρχεία:
-definitions.h
-interface.h
-interface_sim.h
-main.c
-simulation.c
-queue_funct.c
-semaphores_funct.c
-makefile

definitions.h:
Περιέχει τον ορισμό των δομών του προγράμματος(process, semaphore, queue)
καθώς και τον ορισμό του χρόνου που διαρκεί το κάθε time slot(time_c).Μία διεργασία περιέχει pid, priority, time of life(tol) time of cs(to_cs) καθώς και old priority(old_pr, αποθηκέυται η αρχική προτεραιότητα της διεργασίας σε περίπτωση που κληρονομίσει την προτεραιότητα καποιας υψηλότερης διεργασίας, ώστε όταν τελείωσει την εκτέλεση του cs της να μπορέσει να εισαχθεί και πάλι στην σωστή ουρά προτεραιότητας).Ένας σεμαφόρος αποτελείται απο state, process(pr, η διεργασία που εκτελεί το cs της έχοντας κάνει down τον συγκεκριμένο semaphore)και το waitting time(Waitting_t, ο χρόνος που μία διεργασία υψηλότερης προτεραιότητας περιμένει στην ουρά του σεμαφόρου ενώ εκτελεί μια διεργασία χαμηλότερης προτεραιοτητας έχοντας κανει down τον συγκεκριμένο σεμαφόρο)

interface.h:
Περιέχει την δήλωση των συναρτήσεων για τον χειρισμό των δομών semaphore, queue

interface_sim.h:
Περιέχει την δήλωση της συνάρτησης προσομοίωσης

queue_funct.c:
Περιέχει τον ορισμό των συναρτήσεων χειρισμού της δομής queue(οι υλοποιήσεις των συναρτήσεων αυτών έχουν παρθεί απο τις διαφάνειες του μαθήματος Δομες δεδομένων και τεχνικές προγραμματισμού του κ. Κουμπαράκη)

semaphores_funct.c:
Περιέχει τον ορισμό των συναρτήσεων χειρισμού της δομής semaphore. Η δομή αυτή περιέχει μεταξύ των άλλων και μια process pr, που είναι η διεργασία που εκτελεί εκείνη την χρονική στιγμη το cs της έχοντας καταφέρει να κάνει down τον συγκεκριμένο σεμαφόρο. Η συνάρτηση copy αντιγράφει την διεργασία p στο τμήμα pr του semaphore s.Αντιστρόφως η συνάρτηση copy_i αντιγράφει την διεργασία pr του semaphore s στην διεργασία p.

main.c:
Παίρνει ως ορίσματα το πλήθος των σημαφόρων, το κατώφλι καθώς και το πλήθος των διεργασιών που θέλουμε να δημιουργήσουμε.Επίσης ορίζονται τα 3 λ και καλείται η συνάρτηση προσομοίωσης με ορίσματα όλα τα παραπάνω.

simulation.c:
η βασική ιδέα της προσομοίωσης ειναι η εξής:
κάθε σημαφόρος έχει την δική του ουρά αναμονής(όπου μπαίνουν οι διεργασίες που η πιθανότητα που παρήγαγαν σε ένα συγκεκριμένο time slot είναι μικρότερη του threshold). Σε κάθε time slot ελέγχουμε αν έχουμε άφιξη(arr_counter%toa == 0 && pid<max_pr). Αν υπάρχει νέα διεργασία της ανατίθενται pid,priority,time of life και τυπώνονται σχετικά μηνυματα.Επίσης κάθε πρώτη διεργασια κάθε ουράς παράγει μια πιθανότητα σε κάθε time slot.Οι διεργασιες μς την υψηλότερη προτεραιότητα προηγούνται στην διαδικασια αυτη.Ξεκινώντας, γίνονται όλες οι κατάλληλες αρχικοποιήσεις και υπολογίζεται ο χρόνος μεταξύ των αφίξεων.Παρακάτω υπάρει το κύριο while loop όπου κάθε επανάληψη ισοδυναμεί με ένα time slot.Για να τερματιστεί το while πρέπει να έχει παραχθεί ο ζητούμενος αριθμός διεργασιών και όλες οι διεργασίες να έχουν εξαντλήσει τον χρόνο ζωής τους(όλες οι ουρές να είναι άδειες και όλοι οι σημαφόροι να έχουν state=1).Μέσα στο κύριο while αρχικά ελέγχονται οι σημαφόροι:
-ελέγχουμε αν η διεργασία που εκτελείται σε κάθε σημαφόρο παρεμποδίζει(έχει μικρότερη προτεραιότητα) άλλες διεργασίες που περιμένουν να εκτελέσουν και αυξάνουμε το waitting_t για τον σημαφόρο
-ελέγχουμε τισ προτεραιότητες των διεργασιών που βρίσκονται στις ουρές των σημαφόρων προκειμένου να αυξηθεί το waitting_t των προτεραιοτήτων(μια φορά για κάθε time slot σε όλους τους σημαφόρους)
-παρακολουθούμε όλες τις προτεραιότητες στις ουρές των σημαφόρων και βρίσκουμε την υψηλότερη και σε περίπτωση που η διεργασία που εκτελεί έχει χαμηλότερη προτεραιότητα τότε πραγματοποιείται και αναφέρεται priority_invertion
-αν η τιμή του σημαφόρου είναι 0 τότε ελέγχουμε σε ποίο στάδιο βρίσκεται η διεργασίαπου εκτελεί(αν έχει τελειώσει και έχει εξαντλησει την ζωη της τότε απλά αλλάζουμε την τιμή του σημαφόρου, αν εχεί τελειώσει και η ζωή της είναι ακόμα θετική τοτε πρεέπει να εισαχθεί και πάλι στην ουρά προτεραιότητας της, αν δεν έχει ολοκληρώσει ακόμα τοτε απλά μειώνουμε την ζωή της.
-μετά τις αλλαγές αυτές ελέγχουμε αν ο σημαφόρος είναι ελεύθερος ώστε να εκτελεστεί η πράξη down απο την πρώτη διεργασία που βρίσκεται στην ουρά του σημαφόρου
Μετά ελέγχεται αν υπάρχει άφιξη και γίνονται οι κατάλληλες αναθέσεις. Έπειτα ξεκινάει η παραγωγή πιθανοτήτων:
-Πρώτη την διαδικασία αυτή εκτελεί η πρ΄τη διεργασία της υψηλότερης προτεραιότητας της οποίας η ουρά δεν είναι άδεια.Έπειτα παράγουν πιθανότητες και οι υπόλοιπες διεργασίες χαμηλότερων προτεραιοτήτων(με σειρά υψηλότερης προτεραιότητας)
-Αν η πιθανότητα είναι < κ τότε η διεργασία επιλέγει τυχαία έναν σημαφόρο στον οποίο εκτελεί down.
-Αν το down πετυχει τοτε η διεργασία παράγει ενά cs_time (μέχρι να είναι <=tol της διεργασίας) ,βγαίνει απο την ουρά προτεραιότητας και αντιγράφεται στον σημαφόρο
-Αν δεν πετύχει τότε και πάλι η διεργασία παράγει ενacs_time με τον ίδιο τρόπο, βγάινει απο την ουρά προτεραιότητας και αντιγράφεται στην ουρά αναμονής του σημαφόρου
-Αν η διεργασία μιας προτεραιότητας βγεί απο την ουρά τότε πρέπει να μην προχωρήσουμε σε επόμενη προτεραιότητα αλλά να συνεχίσουμε με το επόμενο στοιχείο της ίδιας προτεραιότητας

Τέλος πρίν την έξοδο απο το while υπολογίζουμε αν πρέπει να συνεχίσουμε τις επαναλήψεις(όπως αναφέρθηκε πιο πάνω)