# Ιόνιο Πανεπιστήμιο

**Σχολή Επιστημών της Πληροφορίας και Πληροφορικής**

**Τμήμα Πληροφορικής**



## Προγραμματιστική Εργασία #2

Μάθημα: Παράλληλος Προγραμματισμός
Διδάσκων: Μιχαήλ Στεφανιδάκης
Η' Εξάμηνο


Αστέριος Παπαμιχαήλ – Π2015059

# 1 Περιγραφή Κώδικα

## 1.1 Δήλωση Σταθερών

Αρχικά γίνεται η δήλωση των παρακάτω σταθερών:

- ο αριθμός των νημάτων (THREADS),
- το μέγεθος του τμήματος που γίνεται απευθείας ταξινόμηση (THRESHOLD),
- το μέγεθος του πίνακα που είναι η ουρά μηνυμάτων (N),
- το μέγεθος του πίνακα προς ταξινόμηση (ARRAY_SIZE)
- και στην συνέχεια τα τρία είδη μηνυμάτων (WORK, FINISH, SHUTDOWN)

## 1.2 Μηνύματα

Ένα μήνυμα της ουράς περιέχει τον τύπο (type) του, το index της αρχής (start) καθώς και του τέλους (end) του τμήματος του πίνακα όταν ο τύπος του μηνύματος είναι WORK. Οι τύποι του μηνύματος χρησιμοποιούνται για την αναγνώριση της εργασίας από κάθε thread. Στην συνέχεια γίνεται η δέσμευση της ουράς και των απαραίτητων μεταβλητών για την διαχείριση της. Από την στιγμή που έχει αναφερθεί η δομή ενός μηνύματος είναι συνετό να μελετηθεί η διαχείριση του.

## 1.3 Η συνάρτηση send( type, start, end)

Αρχικά η συνάρτηση κλειδώνει την πρόσβαση της ουράς, εάν δεν είναι ήδη κλειδωμένη από άλλο thread. Αν είναι κλειδωμένη περιμένει μέχρι την αποδέσμευσή της. Αφού αποδεσμευτεί η ουρά, την δεσμεύει και ελέγχει τον αριθμό των μηνυμάτων στην ουρά, αν είναι γεμάτη περιμένει ώσπου να βρεθεί χώρος. Όταν βρεθεί γίνεται προσθήκη της εργασίας σε αυτήν. Και τέλος στέλνει signal για να ξεκλειδώσει ένα τουλάχιστον thread από την μεταβλητή msg_in. Όμως στην συγκεκριμένη περίπτωση θα ξεκλειδώσει μόνο ένα γιατί τα υπόλοιπα είναι σταματημένα στο mutex.

## 1.4 Η συνάρτηση receive( *type, *start, *end)

Η συγκεκριμένη ακολουθεί την ίδια λογική με την συνάρτηση send με μερικές διαφορές. Αντί να ελέγχει εάν η ουρά είναι γεμάτη ελέγχει εάν είναι άδεια, αν είναι περιμένει ώσπου να μπει κάποιο μήνυμα.

## 1.5 Η συνάρτηση thread_func(*params)

Αρχικά μετατρέπει την παράμετρο params στον double pointer που δείχνει στον πίνακα που είναι προς ταξινόμηση. Στη συνέχεια μπαίνει στον ατέρμονο βρόχο (while). Καλεί την συνάρτηση receive για να πάρει κάποιο μήνυμα. Αφού λάβει μήνυμα ελέγχει τον τύπο του, αν ο τύπος του είναι FINISH το επαναπροωθεί, αν ο τύπος του είναι SHUTDOWN τότε επαναπροώθει το μήνυμα και βγαίνει από τον ατέρμονο βρόχο. Στην περίπτωση που ο τύπος του μηνύματος είναι WORK τότε ελέγχει το μέγεθος του τμήματος και αν είναι

μικρότερο του THRESHOLD ταξινομεί το τμήμα με την Insertion Sort και στέλνει μήνυμα FINISH που περιέχει και το εύρος ταξινόμησης: [start:end] και πηγαίνει στην αρχή του βρόχου. Από την άλλη, αν το μέγεθος του τμήματος είναι μεγαλύτερο του THRESHOLD, τότε καλεί την συνάρτηση partition που επιστέφει το pivot. Από την στιγμή που το pivot είναι γνωστό μπορεί να χωριστεί αναλόγως ο πίνακας. Αυτό πετυχαίνεται στέλνοντας δύο μηνύματα τύπου WORK με τα ανάλογα εύρη τμημάτων.

### 1.6 main()

Αρχικά γίνεται δέσμευση μνήμης για τον πίνακα που είναι προς ταξινόμηση και έπειτα η αρχικοποίηση του με τυχαίες τιμές. Στη συνέχεια δημιουργούνται τα threads με παράμετρο τον προαναφερόμενο πίνακα. Το 'κεντρικό' πρόγραμμα ξεκινά την διαδικασία ταξινόμησης στέλνοντας ένα μήνυμα τύπου WORK με start το πρώτο στοιχείο και end το τελευταίο. Μετά περιμένει λαμβάνοντας μόνο μηνύματα τύπου FINISH. Καθώς τα λαμβάνει υπολογίζει τον αριθμό των στοιχείων που έχουν ταξινομηθεί και όταν φτάσει αυτός στο μέγεθος του πινάκα, στέλνει μήνυμα SHUTDOWN στα threads. Συνεχίζει η διαδικασία με έλεγχο για την ορθότητα της ταξινόμησης. Τέλος τερματίζουν τα threads και απελευθερώνεται η μνήμη.