Κωδικός Εργασίας (2.1)

**Ομάδα** [2]

**Μέλος A**: [Θεοδοσίου Κωνσταντίνος, 1619, konstheo@uth.gr]

**Μέλος B**: [Καράκουλα Κατερίνα, 1604, karakoul@uth.gr]

## Περιγραφή σημείων συγχρονισμού με ψευτοκώδικα

|  |
| --- |
| (α) Για τη λειτουργια των mtx και safe\_mtx. Η επιθυμητη λειτουργια ειναι να ξεκλειδωσουμε ενα-ενα thread με τη mtx δινοντας του τα δεδομενα που του αντιστοιχουν. Επειτα το thread με τo safe\_mtx ειδοποιει τη main οτι πηρε επιτυχως τα δεδομενα του, και μπορει να συνεχισει τη λειτουργια της. |
| (β) main:  init(mtx), init(safe\_mtx)  lock(mtx), lock(safe\_mtx)  while(1) {  /\*pairnoume dedomena ap to xrhsth\*/  while(i<slices) {  unlock(mtx)  lock(safe\_mtx)  }  }  thread:  lock(mtx)  /\*krisimos kwdikas\*/  unlock(safe\_mtx) |
| (γ) Σε περιπτωση που ολα τα threads δημιουργηθουν και χασουν τον επεξεργαστη πριν γινει το lock(mtx), τοτε η το πρωτο unlock(mtx) που θα κανει η main θα χαθει και θα κολλησει το προγραμμα. |

## Περιγραφή σημείων συγχρονισμού με ψευτοκώδικα

|  |
| --- |
| (α) Για τη λειτουργια των blck και safe\_blck. Το safe\_blck δηλωνει στη main οτι καποιο thread εχει τελειωσει τους υπολογισμους του μπλοκαροντας το thread αυτο. Το blck χρησιμοποιειται για να μπλοκαρει η main μεχρι να ειναι προσβασιμα τα δεδομενα καποιου αλλου thread. |
| (β)main:  init(blck), init(safe\_blck)  lock(blck)  while(1) {  /\*pairnoume dedomena ap to xrhsth kai energopoioume ta threads\*/  while(i<slices){  lock(blck)  /\*zwgrafizei h main\*/  unlock(safe\_blck)  }  }  thread:  while(1){  /\*parnei dedomena kai kanei upologismous\*/  lock(safe\_blck)  unlock(blck)  } |
| (γ) Δεν διακρινουμε καποιο προβλημα σε αυτο το συγχρονισμο. |

## Σχολιασμός φαινομένων / αποτελεσμάτων

|  |
| --- |
| (α) Στην υλοποιηση της εργασιας με ενεργη αναμονη ο χρονος που κανει το προγραμμα για να ζωγραφισει το αποτελεσμα ειναι πολυ μεγαλυτερος απο αυτον του προγραμματος που χρησιμοποιει mutex. Οταν ο χρηστης δωσει πολλα slices ο χρονος στο προγραμμα με mutex μειωενεται εκθετικα σε σχεση με τη προηγουμενη υλοποιηση. Ενω οταν τα slices που δινει ο χρηστης ειναι λιγα, η διαφορα δεν ειναι τοσο μεγαλη. |
| (β) Το φαινομενο αυτο παρουσιαζεται γιατι, στην ενεργη αναμονη κανει πολλες επαναληψεις και ελεγχους ακομα και σε περιπτωσεις που δεν χρειαζονται χωρις να δινει τον επεξεργαστη. Σε αντιθεση με τα mutexes που καθε φορα που κολλανε δινουν κατευθειαν τον επεξεργαστη, χωρις να κανουν ασκοπες επαναληψεις. |