Κωδικός Εργασίας (2.3)

**Ομάδα** [13]

**Μέλος A**: [Γεώργιος, Σιδέρης, 1622, sideris@uth.gr]

**Μέλος B**: [Φώτιος, Τσώκος, 1679, tsokos@inf.uth.gr]

## Περιγραφή σημείων συγχρονισμού με ψευτοκώδικα

**1.1) Η τεχνική επίλυσης (Train.c)**

Εχουμε το τρενο με μια χωρητικότητα Capacity, και Ν επιβάτες. Θέλουμε το τρένο να γεμίζει με επιβάτες να αδειάζει, να επιστρέφει, να παίρνει τους επόμενους κ.ο.κ. μέχρι το τελευταίο δρομολόγιο όπου μπορεί να ξεκινήσει χωρίς να είναι γεμάτο αν δεν υπάρχουν αρκετοί εναπομείναντες επιβάτες για να το γεμίσουν.

Η λογική που χρησιμοποιήσαμε είναι απλή:

Αφού μάθουμε τον αριθμό των συνολικών επιβατών, δημιουργούμε ένα νήμα για τον κάθε επιβάτη, και ένα νήμα για το τρένο.Οι επιβάτες περιμένουν πριν επιβιβαστούν να βεβαιωθούν ότι υπάρχει ακόμα χώρος στο τρένο (1ο σημείο συγχρονισμού),δίνοντας ο ένας επιβάτης την σειρά στον επόμενο. Αν δεν χωράνε άλλοι επιβάτες, το τρένο ξεκινά (2ο σημείο συγχρονισμού), αλλιώς περιμένουμε να γεμίσει. Αφού γεμίσει και κάνει την βόλτα, οι επιβάτες αποβιβάζονται ένας ενας (καθε επιβάτης δίνει σειρά στον επόμενο να κατέβει) και αν κατέβουν όλοι το τρένο επιστρέφει πίσω (3ο σημείο συγχρονισμού). Το πρόγραμμα μέσω της main περιμένει σήμα απο το νήμα του τρένου ότι τελειώσαν οι διαδρομές και οι επιβάτες για να τερματίσει την λειτουργία του(4ο σημείο συγχρονισμού).

**1.2) Ψευδοκώδικας**

**main** {

create threads passengers;

create thread train;

wait end; **//4ο σημείο συγχρονισμού**

}

**train** {

wait return; **//3ο σημείο συγχρονισμού**

if (NoMoreTrips) {

signal end;

}

}

**passenger** {

wait enter; **//1o σημειο συγχρονισμου**

onTrain++;

if (!Full) {

signal enter;

}

else {

signal go; **//2ο σημείο συγχρονισμου**

}

wait go;

onTrain--;

if (!empty) {

signal go;

}

else {

signal return;

}

**1.3) Αμοιβαίος αποκλεισμός και λιμοκτονία**

Tο πρόγραμμα μας εγγυάται αμοιβαίο αποκλεισμό (κάθε επιβάτης περιμένει πρώτα να δει αν χωράει πριν μπεί στο τρένο) και μέσω των mutexes δεν υπάρχει περίπτωση να μπούνε δύο μαζί.

Επίσης, δεν υπάρχει περίπτωση ένα νήμα να μείνει κολλημένο για πάντα γιατί η λύση μας προυποθέτει κάθε νήμα να “ξυπνήσει” απο το προηγούμενο του.

**1.4) Αδυναμίες**

Η λύση μας λειτουργεί για πεπερασμένο αριθμό επιβατών.Σε περίπτωση που είχαμε συνεχή και τυχαία χρονικά είσοδο επιβατών θα υπήρχε το πρόβλημα ένα τρένο να περιμένει πολλή ώρα να γεμίσει.

**1.5) Η λύση του passenger.c**

Σε αυτή την εκδοχή του προγράμματος, το τρένο δεν έχει ένα ξεχωριστο νήμα, αλλα η λειτουργία του υλοποιείται μέσα σε κάθε νήμα επιβάτη.

Ολα τα σημεία συγχρονισμού λειτουργούν όπως και στην παραπάνω εκδοχή.

## Σχολιασμός φαινομένων / αποτελεσμάτων

Το πρόγραμμα δουλεύει σωστα. Όλοι οι επιβάτες μπαίνουν σε τρένο, όλα τα τρένα είναι γεμάτα εκτός απο το τελευταίο το οποίο σωστα περιμένει όσους επιβάτες έχουν απομείνει (Passengers%trips).