ECE 316 - Operating Systems and Networking Laboratory

Practical Assignment 8

For each Assignment, a report (.pdf) and the source-code (e.g., .c, .cpp, .m, .bat etc.) of the solution should be submitted through Microsoft Teams "ECE316 – Operating Systems and Networks Laboratory" no later than the due date of the Assignment. The report should start with a cover page that clearly contains the assignment number and the Team number and the names of the members. In your report, include only the pseudocode, not the actual code, with any comments and description you may want to add, as well as a typical scenario that you used to test your programs. Please note that the report should be as concise as possible. Caution: You are not allowed to upload executables (.exe)!

If input test files are given, you are not allowed to make any changes to the provided input files.

Report File Naming Format: "Team#_Assignment#.pdf"

1. **[20%]** Να γραφτούν 3 pthreads για το UNIX τα οποία να τυπώνουν στην οθόνη συγχρονισμένα ένα μήνυμα που να περιέχει την ταυτότητα του νήματος και την τιμή ενός μετρητή ο οποίος να αυξάνεται κάθε φορά κατά ένα. Δηλαδή θα πρέπει να φαίνεται στην οθόνη:

```
Thread 0 counter=0
Thread 1 counter=1
Thread 2 counter=2
Thread 0 counter=3
Thread 1 counter=4
Thread 2 counter=5
      ⋮
```

   Το πρόγραμμα να τερματίζει μετά από την εκτύπωση 20 μηνυμάτων. Για την υλοποίηση να χρησιμοποιήσετε μια κλειδωνιά (mutex lock) για τον μετρητή και μεταβλητές συνθήκης (condition variables) για τον έλεγχο του επόμενου νήματος που πρέπει να τρέξει.

2. **[50%]** Να γραφεί ένας χρονοδρομολογητής **με ανάδραση (Feedback)** με τα εξής χαρακτηριστικά:

   a) Ο χρονοδρομολογητής θα πρέπει να δέχεται το κβάντο χρόνου **qt** καθώς και τις διεργασίες που πρέπει να εκτελεστούν ως παραμέτρους στη γραμμή εντολών με την εξής μορφοποίηση:

```
./scheduler qt path_process1 startTime1 executionTime1
path_process2 startTime2 executionTime2 ...
```

b) Κάθε μία από τις διεργασίες που θα δημιουργεί και θα εναλλάσσει ο χρονοδρομολογητής θα πρέπει να είναι ξεχωριστό πρόγραμμα το οποίο θα τυπώνει ένα μήνυμα κάθε 0.2s με την ακόλουθη μορφή:

```
Program i: Messages printed: 1.
Program i: Messages printed: 2.
⋮
```

c) Κατά την κυκλική έναρξη και παύση της κάθε διεργασίας η ίδια θα τυπώνει τα μηνύματα:

```
Process <pid> is executing.
<execution of process pid>
Process <pid> is suspended.
```

d) Η διακοπή και εκτέλεση των διεργασιών θα γίνεται με τα σήματα SIGTSTP και SIGCONT

e) Να μελετηθεί και να σχολιαστεί η συμπεριφορά του συστήματος για μικρές και μεγάλες τιμές του κβάντου χρόνου (π.χ. qt = 0.2s, 0.4s, 0.8s, 1.6s…) για 6 διεργασίες με τους ακόλουθους χρόνους έναρξης και εκτέλεσης.

startTimes= [0,2,4,6,8,9]
execTimes= [4,6,3,4,5,1]

Σημειώσεις:

i) Για την εκτέλεση μιας διεργασίας παιδί ως ξεχωριστό πρόγραμμα χρησιμοποιείστε την εντολή `execl(...)`.

ii) Για την μέτρηση του χρόνου που πέρασε από την έναρξη της λειτουργίας του χρονοπρογραμματιστή χρησιμοποιείστε την εντολή *`gettimeofday`*`(...)`.

3. **[30%]** Να γράψετε ένα πολυ-νηματικό (multi-threaded) πρόγραμμα σε Pthreads που υλοποιεί τον αλγόριθμο του τραπεζίτη.

Το κυρίως πρόγραμμα θα πρέπει να δημιουργεί νήματα περιοδικά τα οποία μπορούν να ζητούν πόρους και να ελευθερώνουν πόρους με τυχαίο τρόπο.

Κάθε δημιουργημένο νήμα ενημερώνει τον τραπεζίτη για το μέγιστο αριθμό πόρων που μπορεί να ζητήσει, καθώς και τον αριθμό πόρων που ζητά αρχικά ώστε ο τραπεζίτης να αποφασίσει αν θα αναθέσει πόρους ή όχι στο συγκεκριμένο νήμα.

Ο τραπεζίτης θα πρέπει να παραχωρεί τους πόρους σε ένα νήμα μόνο αν το σύστημα παραμένει σε ασφαλή κατάσταση διαφορετικά το νήμα μπαίνει σε κατάσταση αναμονής μέχρι να μπορεί να εξυπηρετηθεί.

Είναι σημαντικό η πρόσβαση στους πόρους να είναι ελεγχόμενη με την χρήση κλειδωνιών (mutex locks) ώστε να αποφεύγονται οποιαδήποτε προβλήματα από την ταυτόχρονη πρόσβαση των διαφόρων νημάτων σε ένα πόρο.

Κάθε φορά που ο αλγόριθμος του τραπεζίτη επεξεργάζεται αιτήσεις θα πρέπει να τυπώνονται οι σχετικές πληροφορίες στην οθόνη για έλεγχο της ορθότητας του αλγορίθμου.

Επιπρόσθετα, θεωρείστε ότι έχουμε τα εξής δεδομένα:

a. Μέγιστος αριθμός πόρων: $\boldsymbol{max} = [34, 46, 52, 43]$.
b. Ο μέγιστος αριθμός πόρων, $\boldsymbol{max_i}$, που μπορεί ζητήσει ένα νήμα είναι 8 ανά πόρο, ενώ το διάνυσμα των μέγιστων πόρων επιλέγεται τυχαία από την ομοιόμορφη κατανομή. (Discrete Uniform Distribution)
c. Ο αριθμός πόρων που ζητά ένα νήμα άμεσα προς ανάθεση δημιουργείται επίσης από την ομοιόμορφη κατανομή έτσι ώστε: $\boldsymbol{allocation_i} \leq \boldsymbol{max_i}$.
d. Ένα νήμα μπορεί να αποφασίζει ανά τακτά χρονικά διαστήματα κατά πόσο χρειάζεται περισσότερους πόρους ή κατά πόσο πρέπει να ελευθερώσει τους πόρους και να τερματίσει.
e. Ο αλγόριθμος του τραπεζίτη θα πρέπει να τρέχει περιοδικά ώστε να επεξεργάζεται όσες αιτήσεις εκκρεμούν, ώστε να αναθέτει πόρους ή να βάζει σε αναμονή τα συγκεκριμένα νήματα.