ΗΜΥ 316 – Εργαστήριο Λειτουργικών Συστημάτων και Δικτύων

Assignment 9

Γενικές Οδηγίες:

Για κάθε Άσκηση, θα πρέπει να υποβάλλεται μια αναφορά σε μορφή (.pdf) και ο πηγαίος κώδικας στην πλατφόρμα του εργαστηριακού μαθήματος πριν την τελική προθεσμία της εργασίας. Η αναφορά θα πρέπει να ξεκινά με ένα εξώφυλλο που θα περιλαμβάνει τον αριθμό της εργασίας, την ομάδα σας, τα στοιχεία του φοιτητή και τα στοιχεία του άλλου μέλους της ομάδας (αν υφίσταται). Κάθε φοιτητής πρέπει να υποβάλει τη δική του αναφορά. Στην αναφορά σας, συμπεριλάβετε μόνο τον ψευδοκώδικα, όχι τον πραγματικό κώδικα, με τα σχόλια και τις περιγραφές που θέλετε να προσθέσετε, καθώς και ένα τυπικό σενάριο που χρησιμοποιήσατε για να δοκιμάσετε τα προγράμματά σας. Σημειώστε ότι η αναφορά πρέπει να είναι όσο το δυνατόν συνοπτική. Τα scripts που γράφετε πρέπει αναγνωρίζουν τυχόν λανθασμένες εισόδους και να εμφανίζουν τα κατάλληλα μηνύματα λάθους.

Προσοχή: Το παραδοτέο αναφοράς πρέπει να είναι οπωσδήποτε σε μορφή .pdf. Δεν επιτρέπεται καμιά άλλη μορφή (πχ .doc, .docx), ούτε συμπίεση (zip, .rar, .7z). Δεν επιτρέπεται να ανεβάσετε εκτελέσιμα αρχεία (.exe). Εάν σας δίνονται επιπρόσθετα αρχεία εισόδου, δεν επιτρέπεται να κάνετε καμία αλλαγή σε αυτά αν δεν σας ζητηθεί εκ των προτέρων. Τα στιγμιότυπα που θα συμπεριληφθούν να είναι ευδιάκριτα.

Όνομα αρχείου αναφοράς: ***"Group#_Surname_Assignment#.pdf"***

Όνομα αρχείων C: ***"Group#_Surname_Assignment#_Thema#.c"***

Προαπαιτούμενα:

Για την παρούσα άσκηση, είναι απαραίτητο να εγκαταστήσετε ένα λειτουργικό σύστημα Linux ή Unix στον προσωπικό σας υπολογιστή. Ένα προτεινόμενο είναι το λειτουργικό UBUNTU.

Οι προτεινόμενες επιλογές είναι:

1) Εγκατατάσταση Ubuntu μέσω WSL2 σε περιβάλλον Windows
   https://learn.microsoft.com/en-us/windows/wsl/install
2) Εγκατάσταση Ubuntu μέσω Multipass σε περιβάλλοντα MacOS, Linux & Windows
   https://canonical.com/multipass/install

Βοηθητικό υλικό:

Στους συνδέσμους παρακάτω βρίσκεται βοηθητικό υλικό για τη διεκπεραίωση της άσκησης:

- http://www.yolinux.com/TUTORIALS/LinuxTutorialPosixThreads.html
- https://randu.org/tutorials/threads/
- https://dsa.cs.tsinghua.edu.cn/oj/static/unix_signal.html
- https://www.geeksforgeeks.org/bankers-algorithm-in-operating-system-2/

Θέμα 1:

Να αναπτύξετε μια εφαρμογή στη γλώσσα προγραμματισμού C που να υλοποιεί την ελεγχόμενη εκτέλεση N διεργασιών σε ένα πολυδιεργασιακό σύστημα, με την ακόλουθη συμπεριφορά:

- Ο χρήστης εισάγει τον επιθυμητό αριθμό διεργασίων (N) με σχετικό μήνυμα.
- Ο πατέρας (parent process) δημιουργεί N διεργασίες-παιδιά χρησιμοποιώντας την fork(), όπου N είναι ακέραιος αριθμός που δίνεται από τον χρήστη.
- Μετά τη δημιουργία κάθε παιδιού, ο πατέρας καλεί την sleep(). Τα παιδιά εκτυπώνουν ένα μήνυμα επιβεβαίωσης της δημιουργίας τους και σταματούν προσωρινά τη λειτουργία τους χρησιμοποιώντας την κλήση pause().
- Κάθε παιδί εγκαθιστά έναν εξατομικευμένο χειριστή σήματος SIGCONT, ο οποίος:
    - Εκτελεί το κύριο υποπρόγραμμα της διεργασίας.
    - Εξασφαλίζει τη συνέπεια και ακολουθία των ενεργειών πριν την έξοδο της διεργασίας.
- Ο πατέρας, αφού δημιουργήσει όλες τις διεργασίες:
    - Ξεκλειδώνει διαδοχικά κάθε παιδί χρησιμοποιώντας την εντολή kill() για αποστολή σήματος SIGCONT.
    - Περιμένει τη λήξη της κάθε διεργασίας μέσω της κλήσης wait(). Το σήμα SIGCONT πρέπει να ενεργοποιεί μόνο τη διεργασία που είναι προγραμματισμένη να εκτελεστεί.
- Κάθε διεργασία πρέπει να εκτελεί τα εξής βήματα:
    - Εκτύπωση μηνύματος "Child with pid is executing".

- Εκτέλεση υποπρογράμματος που σχετίζεται με τη διεργασία (για παράδειγμα, πολύπλοκοι μαθηματικοί υπολογισμοί ή λειτουργίες αρχείων).
- Εκτύπωση μηνύματος "Child with pid has finished" πριν την έξοδο της διεργασίας.

Ζητούμενα:

- Αναφορά που θα περιέχει εκτενής επεξήγηση της υλοποίησης και εξηγήσετε πώς η χρήση χειριστών σήματος, αναστολής και επανεκκίνησης των διεργασιών βελτιστοποιεί τη διαχείριση πόρων του συστήματος.
- Τον πηγαίο κώδικα του προγράμματος σε γλώσσα προγραμματισμού C ο οποίος πρέπει να περιλαμβάνει επαρκή τεκμηρίωση (σχόλια), που να εξηγούν την υλοποίηση κάθε μέρους του προγράμματος.

Θέμα 2:

Υλοποιήστε ένα πολυνηματικό πρόγραμμα σε γλώσσα C χρησιμοποιώντας την βιβλιοθήκη Pthreads, το οποίο θα εφαρμόζει τον αλγόριθμο του τραπεζίτη για τη διαχείριση πόρων σε ένα πολυνηματικό περιβάλλον. Το πρόγραμμα θα πρέπει να υλοποιεί τα εξής:

- Ο συνολικός αριθμός διαθέσιμων πόρων m θα πρέπει να ορίζεται από τον χρήστη κατά την εκτέλεση του προγράμματος.
- Το πρόγραμμα θα διαχειρίζεται ένα δυναμικό πλήθος νημάτων n που δημιουργείται επίσης δυναμικά (εισαγωγή από χρήστη). Η δημιουργία των νημάτων να γίνεται περιοδικά και να βασίζεται σε καθορισμένο χρόνο εμφάνισης νέων αιτήσεων.
- Κάθε νήμα κατά τη δημιουργία του πρέπει να ορίζει:
  - Τις μέγιστες ανάγκες του (maximum claim vector), με τον συνολικό αριθμό να είναι περιορισμένος από το μέγεθος των διαθέσιμων πόρων.
  - Έναν αριθμό αρχικών πόρων που αιτείται άμεσα (initial request).
- Αν κάποια αίτηση πόρων από ένα νήμα δεν μπορεί να εξυπηρετηθεί λόγω ενδεχόμενης μη ασφαλούς κατάστασης, το νήμα να εισέρχεται σε κατάσταση αναμονής (blocking state) και να εξυπηρετείται αργότερα όταν το σύστημα καθίσταται ασφαλές.

- Αν κάποιο νήμα τελειώσει τη λειτουργία του, πρέπει να ελευθερώνει όλους τους πόρους του, ώστε να είναι διαθέσιμοι για άλλα νήματα.
- Ο αλγόριθμος του τραπεζίτη να εκτελείται περιοδικά για την επεξεργασία όλων των εκκρεμών αιτήσεων. Κάθε φορά που εκτελείται, να τυπώνονται λεπτομερώς τα εξής:
    - Η τρέχουσα κατάσταση των πόρων (available, allocation, need).
    - Αιτήσεις που έχουν εγκριθεί ή απορριφθεί.
    - Αν το σύστημα βρίσκεται σε ασφαλή κατάσταση.
- Να εξασφαλίσετε ότι η πρόσβαση στους πόρους γίνεται με σωστή χρήση μηχανισμών κλειδώματος (locks) για να αποφευχθούν ασυνεπείς καταστάσεις λόγω ταυτόχρονης πρόσβασης.
- Να υλοποιήσετε μηχανισμούς αντιμετώπισης deadlocks, εάν προκύψουν.
- Περιορισμοί και Πολυπλοκότητα:
    - Ο μέγιστος αριθμός πόρων max_resources σε κάθε κατηγορία πόρων να είναι >= 50.
    - Ο αριθμός νήματος να μπορεί να φτάνει μέχρι και τα 20.
    - Οι ανάγκες και αιτήσεις να δημιουργούνται δυναμικά.
- Το πρόγραμμα να υποστηρίζει την εισαγωγή επιπλέον νήματος κατά τη λειτουργία του συστήματος.
- Να μπορεί να κλιμακωθεί για τη διαχείριση διαφορετικών τύπων πόρων.

Ζητούμενα:

- Αναφορά που θα περιέχει εκτενής επεξήγηση της υλοποίησης με ψευδοκώδικα.
- Τον πηγαίο κώδικα του προγράμματος σε γλώσσα προγραμματισμού C ο οποίος πρέπει να περιλαμβάνει επαρκή τεκμηρίωση (σχόλια), που να εξηγούν την υλοποίηση κάθε μέρους του προγράμματος.

Είναι απαραίτητο για να βαθμολογηθεί η άσκηση να μεταγλωττίζεται χωρίς λάθη σύμφωνα με τις οδηγίες, και τα εκτελέσιμα που προκύπτουν από τους πηγαίους κώδικες να τρέχουν σύμφωνα με τα παραπάνω παραδείγματα. Η διόρθωση των ασκήσεων θα γίνει σε λειτουργικό σύστημα Linux, οπότε πρέπει οπωσδήποτε να εξασφαλίσετε ότι λειτουργούν οι κώδικες.