University of Cyprus
Department of Electrical and Computer Engineering

Αναφορά ΗΜΥ316

Πανεπιστήμιο Κύπρου

Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

Assignment 8 (Due 31/03/2022)

Ονόμα και Ταυτότητα :

Εντουίνα Κάρουλλα 1042364

Ομάδα : 7

Φοίβος Λύμπουρας	1016477
Στέλιος Καραγίωργης	1021340
Θεοδόσιος Ιωάννου	1020844

Άσκηση 1

Στην πρώτη άσκηση έπρεπε να γράψουμε 3 pthreads τα οποία να τυπώνουν στην οθόνη συγχρονισμένα ένα μήνυμα που να περιέχει την ταυτότητα του νήματος και την τιμή ενός μετρητή ο οποίος αυξάνεται κάθε φορά κατά ένα.

Για τον σκοπό αυτό δημιουργήσαμε 3 pthreads και 3 functions (ένα για κάθε pthread). Το κάθε pthread στέλνει στην ανάλογη συνάρτηση το μήνυμα που θέλει να τυπώσει.

Για τον συγχρονισμό και για το τύπωμα της κάθε συνάρτησης με την σειρά στην οποία μας ζητήθηκε χρησιμοποιήσαμε mutex lock και cond_wait, cond_signal. Το mutex lock εμποδίζει τα άλλα pthreads να τρέχουν ταυτόχρονα.

Για να σταματά το κάθε thread έπειτα από την τύπωση του μηνύματος χρησιμοποιήσαμε μια μεταβλητή count που παίρνει τις τιμές 1,2,3 ανάλογα με το thread στο οποίο θέλουμε να τυπώσουμε. Δηλαδή το count ξεκινά με την τιμή μηδέν και όταν καλείται η πρώτη συνάρτηση για να τυπωθεί το πρώτο thread (Tread 0) όταν η μεταβλητή γίνει ίση με 1 κάνει pthread_cond_wait(&cond1, &mutex1) και προχωρά στην επόμενη συνάρτηση με την χρήση της pthread_cond_signal(&cond2). Η συνάρτηση pthread_cond_signal είναι αυτή που θα ορίσει ποιο thread θα τυπωθεί στη συνέχεια.

Αποτελέσματα:

```
Thread 0 Counter 0
Thread 1 Counter 1
Thread 2 Counter 2
Thread 0 Counter 3
Thread 1 Counter 4
Thread 2 Counter 5
Thread 0 Counter 6
Thread 1 Counter 7
Thread 2 Counter 8
Thread 0 Counter 9
Thread 1 Counter 10
Thread 2 Counter 11
Thread 0 Counter 12
Thread 1 Counter 13
Thread 2 Counter 14
Thread 0 Counter 15
Thread 1 Counter 16
Thread 2 Counter 17
Thread 0 Counter 18
Thread 1 Counter 19
```

Άσκηση 2

Στην δεύτερη άσκηση μας ζητήθηκε να γράψουμε ένα χρονοδρομολογητή (scheduler) Feedback με κάποια χαρακτηριστικά.

Οι λειτουργία του χρονοδρομολογητή Round-Robin είναι η εξής επιλέγει ένα διάστημα q(quantum) και εκτελεί κάθε διεργασία για q χρονικές μονάδες και στη συνέχεια εναλλάσσει διεργασία. Τα πλεονεκτήματα αυτής της πολιτικής είναι ότι είναι απλή, είναι δίκαιη, καμιά διεργασία δεν υποφέρει από παρατεταμένη στέρηση και παρουσιάζει καλή απόκριση για μικρές διεργασίες (μικρές διεργασίες δεν περιμένουν πολύ πίσω από μεγάλες διεργασίες). Τα μειονεκτήματα αυτής της πολιτικής είναι ότι πιθανόν να έχει μειωμένη διεκπεραιωτική ικανότητα εάν το q είναι πολύ μικρό και δίνει προτεραιότητα στις διεργασίας με περισσότερους υπολογισμούς παρά σε διεργασίες με Ε/Ε.

Για την υλοποίηση της άσκησης 1 δημιουργήσαμε συνολικά 5 διεργασίες και ένα χρονοδρομολογητή.

Για την κάθε διεργασία χρησιμοποιήσαμε τις εξής εντολές:

Fflush(stdout): καθαρίζει το output buffer για να μπορέσουν να εκτυπωθούν τα μηνύματα

Execl: εκτέλεση μιας διεργασίας παιδί ως ξεχωριστό πρόγραμμα

Atof(argc[1]): για να δημιουργήσουμε το argv[1] από atoi σε float

Signal(SIGCONT, signal_cont_handler): δηλώνει το signal SIGCONT στο signal_cont_handler

Usleep(): χρησιμοποιείται για καθυστέρηση

Για την κάθε διεργασία χρησιμοποιήσαμε τα εξής σήματα:

SIGTSTP για διακοπή της διεργασίας ενεργοποιώντας τον stop-handler

SIGCONT για επανεκκίνηση της διεργασίας ενεργοποιώντας τον signal_condition

Για την κάθε διεργασία χρησιμοποιήσαμε τα εξής arguments:

Το πρώτο είναι το start time

Το δεύτερο είναι το execution time

Για τον χρονοδρομολογητή χρησιμοποιήσαμε τις εξής εντολές:

Gettimeofday() για την μέτρηση του χρόνου που πέρασε από την έναρξη της λειτουργίας του χρονοπρογραμματιστή

atof(argv[1]) για να δημιουργήσουμε το argv[1] από atoi σε float

```
ekar02@Device:~$ g++ processass8.c -lpthread -o pr
ekar02@Device:~$ g++ processass8.c -lpthread -o pr1
ekar02@Device:~$ g++ processass8.c -lpthread -o pr2
ekar02@Device:~$ g++ processass8.c -lpthread -o pr3
ekar02@Device:~$ g++ processass8.c -lpthread -o pr4
^[[Aekar02@Deviceg++ processass8.c -lpthread -o pr5
ekar02@Device:~$ ./scheduler 1 pr 0 4 pr1 2 6 pr2 4 4 pr3 6 4 pr4 8 5 pr5 9 1
pr 0 4
pr1 2 6
pr2 4 4
pr3 6 4
pr4 8 5
pr5 9 1
Child pid: 0 223
Parent pid: 222
Child pid: 1 224
Parent pid: 222
Child pid: 2 225
Parent pid: 222
Child pid: 3 226
Parent pid: 222
Child pid: 4 227
Parent pid: 222
Child pid: 5 228
Parent pid: 222

Processes time: 24


Process 223 is executing.

Program: 1 messages printed: 1

Process 223 is executing.

Program: 1 messages printed: 2


Process 223 is suspended.
Process 224 is executing.


Program: 2 messages printed: 1

Process 224 is executing.

Program: 2 messages printed: 2

Process 224 is suspended.


Process 225 is executing.

Program: 3 messages printed: 1

Process 225 is executing.

Program: 3 messages printed: 2


Process 225 is suspended.
Process 226 is executing.


Program: 4 messages printed: 1

Process 226 is executing.

Program: 4 messages printed: 2

Process 226 is suspended.


Process 227 is executing.

Program: 5 messages printed: 1
```

```
Process 227 is suspended.
Process 228 is executing.


Program: 6 messages printed: 1


Process 228 is suspended.
Process 227 is executing.


Program: 5 messages printed: 2


Process 227 is suspended.
Process 223 is executing.


Program: 1 messages printed: 3


Process 224 is executing.
Process 223 is suspended.


Program: 2 messages printed: 3


Process 225 is executing.
Process 224 is suspended.


Program: 3 messages printed: 3


Process 225 is suspended.
Process 226 is executing.


Program: 4 messages printed: 3


Process 227 is executing.
Process 226 is suspended.


Program: 5 messages printed: 3


Process 223 is executing.
Process 227 is suspended.


Program: 1 messages printed: 4


Process 224 is executing.
Process 223 is suspended.


Program: 2 messages printed: 4


Process 225 is executing.
```

```
Process 225 is executing.
Process 224 is suspended.


Program: 3 messages printed: 4


Process 225 is suspended.
Process 226 is executing.


Program: 4 messages printed: 4


Process 226 is suspended.
Process 227 is executing.


Program: 5 messages printed: 4


Process 227 is suspended.
Process 224 is executing.


Program: 2 messages printed: 5


Process 224 is suspended.
Process 227 is executing.


Program: 5 messages printed: 5


Process 227 is suspended.
Process 224 is executing.


Program: 2 messages printed: 6

Process 224 is suspended.

Scheduler time: 13
ekar02@Device:~$
```

Άσκηση 3

Ο αλγόριθμος του τραπεζίτη λειτουργεί ως εξής: τα δημιουργημένα νήματα ζητούν πόρους και ελευθερώνουν πόρους με τυχαίο τρόπο. Έτσι κάθε δημιουργημένο νήμα ενημερώνει τον τραπεζίτη για τον μέγιστο αριθμό πόρων που μπορεί να ζητήσει και τον αριθμό πόρων που χρειάζεται. Με αυτό τον τρόπο ο τραπεζίτης αποφασίζει αν θα αναθέσει πόρους ή όχι στο συγκεκριμένο νήμα, έτσι ώστε το σύστημα πάντα να παραμένει σε ασφαλή κατάσταση και να μην βρίσκετε σε αδιέξοδο. Αν ο τραπεζίτης δεν μπορεί να αναθέσει τους πόρους που ζήτησε ένα νήμα, τότε το νήμα μπαίνει σε κατάσταση αναμονής, μέχρι κάποιο άλλο νήμα να ελευθερώσει πόρους και να μπορεί ο τραπεζίτης να δώσει τους πόρους που ζήτησε το νήμα προηγουμένως, εφόσον δεν θα είναι τώρα σε ασφαλή κατάσταση.

```
ekar02@Device:~$ ./ask3as8
Enter available VECTOR
4 5 5 5
Enter Allocation Matrix
3 3 3 3
5 4 3 3
2 3 3 2
6 7 2 1
3 4 1 7
Available Vector is:
4, 5, 5, 5,
Allocation Matrix is:
{ 3, 3, 3, 3, }
{ 5, 4, 3, 3, }
{ 2, 3, 3, 2, }
{ 6, 7, 2, 1, }
{ 3, 4, 1, 7, }
Need Matrix is:
{ 5, 5, 5, 5, }
{ 3, 4, 5, 5, }
{ 6, 5, 5, 6, }
{ 2, 1, 6, 7, }
{ 5, 4, 7, 1, }
0, 1, 2, 4,
Resources are being allocated...
Checking is the state is safe...

X====================X
|State is not safe.          |
X====================X
```

```
Available Vector is:
1, 1, 1, 1,
Allocation Matrix is:
{ 2, 1, 1, 1, }
{ 3, 5, 6, 2, }
{ 4, 4, 4, 4, }
{ 3, 5, 5, 3, }
{ 7, 2, 1, 5, }
Need Matrix is:
{ 6, 7, 7, 7, }
{ 5, 3, 2, 6, }
{ 4, 4, 4, 4, }
{ 5, 3, 3, 5, }
{ 1, 6, 7, 3, }
1, 3, 2, 1,
Resources are being allocated...
Not enough Resources.
1, 2, 1, 0,
Resources are being allocated...
Not enough Resources.
4, 2, 0, 0,
Resources are being allocated..
Not enough Resources.
0, 4, 2, 2,
4, 1, 0, 4,
Released.
Metrix Available:
8, 12, 9, 9,
Metrix Allocated:
{ 2, 1, 1, 1, }
{ 3, 1, 3, 1, }
{ 2, 2, 1, 1, }
{ 2, 1, 3, 3, }
{ 3, 1, 1, 1, }
Metrix Need:
{ 6, 7, 7, 7, }
{ 5, 7, 5, 7, }
{ 6, 6, 7, 7, }
{ 6, 7, 5, 5, }
{ 5, 7, 7, 7, }
1, 2, 2, 2,
Resources are being allocated...
Checking is the state is safe...

x======================x
|Safe Mode. Resources Allocated|
x======================x
```