Λειτουργικά Συστήματα

Άσκηση 3

1 Εισαγωγικά

Όταν δύο ή περισσότερες διεργασίες ή νήματα προσπελαύνουν τον ίδιο πόρο (resource) ή γράφουν στις ίδιες θέσεις μνήμης, απαιτείται συγχρονισμός για αποφυγή race conditions. Για τον συγχρονισμό διεργασιών (processes) και νημάτων (threads) παρέχονται τα εξής εργαλεία:

Processes: Semaphores (σημαφόροι)

Threads: Mutexes και condition variables

Στην άσκηση αυτή θα εξοικειωθείτε με τη χρήση των παραπάνω εργαλείων σε διαφορετικά σενάρια. Για τη διεξαγωγή της άσκησης κατεβάστε το σχετικό code template από τη διεύθυνση `http://rio.ceid.upatras.gr:8000/~spyros/os/code.tar`.

2 Συγχρονισμός διεργασιών

Μελετήστε τον κώδικα των προγραμμάτων `syn_process_1` και `syn_process_2`. Ο στόχος αυτής της άσκησης είναι να τα τροποποιήσετε κάνοντας χρήση σημαφόρων ώστε τα δύο αυτά προγράμματα να παράγουν «σωστό» output, χωρίς όμως να τροποποιήσετε καθόλου τη `display()`.

Τα προγράμματα αυτά μπορείτε να τα κάνετε compile ως εξής:

```
gcc syn_process_1.c display.c -o syn_process_1
gcc syn_process_2.c display.c -o syn_process_2
```

- Για το `syn_process_1`, θέλουμε να αποφύγουμε το «ανακάτεμα» των μηνυμάτων που τυπώνονται. Για παράδειγμα το παρακάτω θεωρείται «σωστό»:

```
Hello world
Hello world
Kalimera kosme
Hello world
Kalimera kosme
Kalimera kosme
```

ενώ το παρακάτω θεωρείται «λάθος»:

```
HelKalo world!
limera kosme
HKeallilo merawokorldsme
```

- Στο `syn_process_2` το παρακάτω output θεωρείται «σωστό», δηλαδή πάντα να τυπώνεται πρώτα το `ab` ακολουθούμενο από το `cd\n`:

```
abcd
abcd
abcd
abcd
abcd
...
```

ενώ το παρακάτω θεωρείται «λάθος»:

```
cd
cd
abababcd
cd
cd
ababcd
```

3 Συγχρονισμός νημάτων

Μετατρέψτε τα προγράμματα `syn_process_1.c` και `syn_process_2.c` να χρησιμοποιούν νήματα αντί για διεργασίες, και ονομάστε τα αντίστοιχα προγράμματα `syn_thread_1.c` και `syn_thread_2.c`.

Για τα νήματα πρέπει να χρησιμοποιήσετε υποχρεωτικά `pthreads`, που είναι το standard API. Αν και τα εργαλεία συγχρονισμού διεργασιών που χρησιμοποιήσατε στα προηγούμενα προγράμματα θα μπορούσαν να χρησιμοποιηθούν κι εδώ, στα προγράμματα αυτά **οφείλετε να χρησιμοποιήσετε υποχρεωτικά τα εργαλεία συγχρονισμού νημάτων που προσφέρει το API της βιβλιοθήκης `pthreads`**.

4 Οδηγίες παράδοσης και αυτόματου feedback

Για την ορθότερη υλοποίηση της άσκησης και τη διόρθωση σφαλμάτων, έχετε στη διάθεσή σας ένα σύστημα αυτόματης διόρθωσης, το οποίο μπορείτε να χρησιμοποιήσετε όσες φορές θέλετε. Με το σύστημα αυτό, μας στέλνετε την υλοποίησή σας, και μέσα σε λίγα λεπτά σας επιστρέφονται συνοπτικά αποτελέσματα του ελέγχου που κάνουμε.

Τα τεστ στα οποία υποβάλλουμε τις υλοποιήσεις σας σε αυτό το στάδιο πιθανόν να είναι λιγότερα (και λίγο απλούστερα) από το σύνολο των τεστ που θα τρέξουμε για την βαθμολόγηση της άσκησης. Δηλαδή, το ότι περνάτε όλα τα τεστ επιτυχώς δεν συνεπάγεται αυτόματα 10 στα 10.

Για να μας στείλετε την άσκηση, ακολουθήστε τα παρακάτω βήματα:

- Στείλτε μας συνημμένα τα αρχεία `syn_process_[12].c`, `syn_thread_[12].c`, και `report.pdf` στη διεύθυνση `spyros+hw3@ceid.upatras.gr`.
- Τα αρχεία `display.c` και `display.h` ΜΗΝ μας τα στείλετε, θα τα προσθέσουμε ούτως ή άλλως μόνοι μας για να βεβαιωθούμε ότι δεν έχουν τροποποιηθεί. Ακόμα κι αν τα στείλετε θα τα αντικαταστήσουμε με τα δικά μας versions.
- Το subject του email ΠΡΕΠΕΙ να είναι αποκλειστικά και μόνο ο Αριθμός Μητρώου σας (π.χ., 1234), χωρίς έξτρα κενά, χωρίς ″AM: 1234″, χωρίς ″Re:″, χωρίς οτιδήποτε άλλο πέρα από τα τέσσερα αριθμητικά ψηφία του ΑΜ σας. Το email πρέπει να σταλεί από τον λογαριασμό σας στο CEID.
- Θα λάβετε άμεσα επιβεβαίωση για το email σας, και σε λίγα λεπτά θα λάβετε και δεύτερο email που θα σας δίνει μια ένδειξη του πόσα τεστ περάσατε. Το περιεχόμενο των τεστ είναι επίτηδες κρυφό, ώστε να διερευνήσετε όλες τις πιθανές αιτίες σφαλμάτων. Μερικά και μόνο μερικά τεστ παρέχουν κάποιο hint για το είδος του λάθους, αν δεν τα περάσετε.

Μπορείτε να στείλετε την υλοποίησή σας όσες φορές θέλετε. Η **τελευταία** υλοποίηση που θα λάβουμε θα είναι και αυτή που θα βαθμολογηθεί. Όλες οι προηγούμενες θα αγνοηθούν. Αν συνεχίσετε να στέλνετε υλοποιήσεις μετά το τέλος της προθεσμίας, για κάθε μέρα καθυστέρησης θα χρεώνεστε βαθμούς όπως έχει εξηγηθεί αναλυτικά στο πρώτο μάθημα (και στις αντίστοιχες διαφάνειες).

ΣΗΜΑΝΤΙΚΟ: Για την παράδοση της τελικής μορφής της άσκησης οφείλετε να συμπεριλάβετε και ένα σύντομο **report.pdf** 1-2 σελίδων, που να περιγράφει τη λογική που ακολουθήσατε και προβλήματα που αντιμετωπίσατε, **υποχρεωτικά σε format PDF**.

Καλή επιτυχία!

```c
#include <unistd.h>
#include <sys/types.h>
#include <sys/wait.h>
#include "display.h"

int main()
{
  int i;

  if (fork())
  {
    for (i=0;i<10;i++)
      display("Hello world\n");
    wait(NULL);
  }
  else
  {
    for (i=0;i<10;i++)
      display("Kalimera kosme\n");
  }

  return 0;
}
```

Listing 1: syn_process_1.c

```c
#include <unistd.h>
#include <sys/types.h>
#include <sys/wait.h>
#include "display.h"

int main()
{
  int i;

  if (fork())
  {
    for (i=0;i<10;i++)
      display("ab");
    wait(NULL);
  }
  else
  {
    for (i=0;i<10;i++)
      display("cd\n");
  }

  return 0;
}
```

Listing 2: syn_process_2.c

```
/* DO NOT EDIT THIS FILE!!!  */

#ifndef __CEID_OS_DISPLAY_H__
#define __CEID_OS_DISPLAY_H__
void display(char *);
#endif
```

Listing 3: `display.h`

```
/* DO NOT EDIT THIS FILE!!!  */

#include <stdio.h>
#include <unistd.h>
#include "display.h"

void display(char *str)
{
  char *p;
  for (p=str; *p; p++)
  {
    write(1, p, 1);
    usleep(100);
  }
}
```

Listing 4: `display.c`