Λειτουργικά Συστήματα

2020 - 2021

1η Εργαστηριακή Άσκηση

Μέρος 1 [30 μονάδες]

Ερώτημα Α [10]: Εξηγήστε προσεκτικά τι κάνει το παρακάτω πρόγραμμα.

```
#include <unistd.h>
#include <stdio.h>
#include <sys/wait.h>
#define  N       30
int main()
{
      pid_t pid[N];
      int i;
      int child_status;

      for (i = 0; i < N; i++)
      {
            pid[i] = fork();

            if (pid[i] == 0)
            {
                  sleep(60-2*i);
                  exit(100+i);
            }
      }
for (i = 0; i < N; i++)
      {
      pid_t wpid = waitpid(pid[i], &child_status, 0);

      if (WIFEXITED(child_status))
            {
                  printf("Child%d terminated with exit status %d\n", wpid,
WEXITSTATUS(child_status));
            }
else
            {
                  printf("Child%d terminated abnormally\n", wpid);
            }
      }

      return (0);
}
```

Ερώτημα Β [20]:: Δημιουργήστε ένα πρόγραμμα στο οποίο μία διεργασία στο Linux/Unix παράγει άλλες Ν θυγατρικές της, όπου το Ν το ορίζει ο χρήστης. Κάθε θυγατρική διαδικασία i, παίρνει ένα εισιτήριο με βάση τον αλγόριθμο του αρτοποιού (Bakery Algorithm), προκειμένου να αυξήσει κατά i το περιεχόμενο όλων των θέσεων ενός Shared Array SA[1..N]. Ο SA είναι πίνακας ακεραίων αριθμών και αρχικά όλες οι θέσεις του πίνακα έχουν την τιμή 0. Μόλις δημιουργηθούν οι Ν διεργασίες και μόνο τότε θα εκτελεί ο πατέρας Ν φορές τη waitpid() ώστε να περιμένει την επιτυχή ολοκλήρωση όλων των θυγατρικών διεργασιών.

Μέρος 2 [70 μονάδες]

Ερώτημα Α [10]

Δίνεται ο ακόλουθος «παράλληλος» κώδικας. Θεωρήστε ότι οι X και Y είναι ακέραιες μεταβλητές που έχουν αρχικά τις τιμές 0 και 10, αντίστοιχα.

```
cobegin
  X := X + 1;
  X := Y + 1;
coend
```

Επίσης θεωρήστε ότι καθεμία από τις δύο εντολές καταχώρησης τιμής στη μεταβλητή X δεν εκτελείται ατομικά-αδιαίρετα αλλά σε 3 βήματα. Συγκεκριμένα:

Η εντολή X := X + 1 εκτελείται σειριακά σε τρία βήματα:

- Βήμα 1: TX := X; /* η τιμή της μεταβλητής X καταχωρείται σε ένα καταχωρητή TX */
- Βήμα 2: TX := TX +1; /* η τιμή του καταχωρητή TX αυξάνεται κατά 1 */
- Βήμα 3: X := TX; /* η τιμή του καταχωρητή TX καταχωρείται στη μεταβλητή X */

Η εντολή X := Y + 1 εκτελείται σειριακά σε τρία βήματα:

- Βήμα 1: TY := Y; /* η τιμή της μεταβλητής Y καταχωρείται σε ένα καταχωρητή TY */
- Βήμα 2: TY := TY +1; /* η τιμή του καταχωρητή TY αυξάνεται κατά 1 */
- Βήμα 3: X := TY; /* η τιμή του καταχωρητή TY καταχωρείται στη μεταβλητή X */

Να δώσετε τις τελικές τιμές που είναι δυνατό να λάβει η μεταβλητή X μετά το πέρας της εκτέλεσης του παράλληλου κώδικα, λαμβάνοντας υπόψη τα δυνατά σενάρια εκτέλεσης (αναμείξεις-interleavings) των εντολών του κώδικα.

Ερώτημα Β [15]

Θεωρήστε τις διεργασίες Process1, Process2 και Process3 που εκτελούνται «παράλληλα» (δηλ. εκτελούνται σύμφωνα με το σχήμα εκτέλεσης “cobegin Process1; Process2; Process3; coend”). Όπως παρουσιάζεται από τον κώδικα των διεργασιών, κάθε διεργασία εκτυπώνει, κατ’ επανάληψη, συγκεκριμένα γράμματα του αγγλικού αλφαβήτου. Συγκεκριμένα:

- Η διεργασία Process1 εκτυπώνει, κατ’ επανάληψη, πρώτα το “P” και μετά το “I”.
- Η διεργασία Process2 εκτυπώνει, κατ’ επανάληψη, το “Z”.
- Η διεργασία Process3 εκτυπώνει, κατ’ επανάληψη, το “A”.

Ο κώδικας των διεργασιών δίνεται ακολούθως:

```
    Process1              Process2              Process3
while (TRUE) {       while (TRUE) {       while (TRUE) {
 print("P");          print("Z");          print("A");
 print("I");         }                    }
}
```

(α) Να συμπληρώσετε τον παραπάνω κώδικα των τριών διεργασιών με εντολές wait (ή down ή P) και signal (ή up ή V) σε σημαφόρους (που πρέπει να αρχικοποιήσετε κατάλληλα σε μη αρνητικές ακέραιες τιμές), προκειμένου να εξασφαλίσετε ότι από την εκτέλεση των διεργασιών θα εκτυπώνεται, για μία και μόνο φορά, η λέξη “PIZZA”.

(β) Να τροποποιήσετε τον κώδικα που δώσατε στο υποερώτημα (α), προσθέτοντας και άλλες εντολές wait (ή down ή P) και signal (ή up ή V), ώστε να εξασφαλίσετε ότι κατά την εκτέλεση των διεργασιών θα εκτυπώνεται επαναληπτικά η λέξη “PIZZA” (δηλ. ότι εκτυπώνεται η συμβολοσειρά: “PIZZAPIZZAPIZZAPIZZA……”).

Σημείωση: Προτείνεται να απαντήσετε στα παραπάνω ερωτήματα χρησιμοποιώντας τρεις (3) σημαφόρους.

Ερώτημα Γ [10]

Σε ένα υπολογιστικό σύστημα εκτελούνται διεργασίες δύο τύπων, οι διεργασίες τύπου Α και οι διεργασίες τύπου Β. Κάθε διεργασία τύπου Α και κάθε διεργασία τύπου Β εκτελεί εντολές down και up σε δύο σημαφόρους s1 και s2, όπως παρουσιάζεται στον παρακάτω κώδικα.

```
Διεργασία_Α
 down(s1);
 up(s2);
```

```
Διεργασία_Β
 down(s2);
 down(s2);
 up(s1);
 up(s2);
```

Θεωρήστε ότι οι s1 και s2 είναι γενικοί σημαφόροι-μετρητές, που δεν λαμβάνουν αρνητικές τιμές, και ότι αρχικοποιούνται στις τιμές 2 και 0, αντίστοιχα.

Έστω ότι εισέρχονται στο σύστημα ταυτόχρονα για να εκτελεστούν 3 διεργασίες τύπου Α (οι διεργασίες Α1, Α2 και Α3) και 2 διεργασίες τύπου Β (οι διεργασίες Β1 και Β2).

(α) Εξετάστε αν είναι δυνατό να ολοκληρωθεί η εκτέλεση των διεργασιών με την εξής σειρά:

1. Α1 (πρώτα ολοκληρώνεται η εκτέλεση της Α1)
2. Α2 (μετά ολοκληρώνεται η εκτέλεση της Α2)
3. Β1 (μετά ολοκληρώνεται η εκτέλεση της Β1)
4. Α3 (μετά ολοκληρώνεται η εκτέλεση της Α3)
5. Β2 (τελευταία ολοκληρώνεται η εκτέλεση της Β2)

Συγκεκριμένα, ζητείται να παρουσιάσετε ένα σενάριο εκτέλεσης των διεργασιών που έχει ως αποτέλεσμα την παραπάνω σειρά ολοκλήρωσης της εκτέλεσης των διεργασιών, δίνοντας τις τιμές που λαμβάνουν οι σημαφόροι s1 και s2, μετά από την εκτέλεση κάθε εντολής των διεργασιών. Αν δεν είναι δυνατό να συμβεί η παραπάνω σειρά, τεκμηριώστε στην απάντησή σας γιατί.

(β) Δίνοντας ανάλογη τεκμηρίωση με αυτή που δώσατε στην απάντησή σας στο προηγούμενο υποερώτημα, εξετάστε αν είναι δυνατό να ολοκληρωθεί η εκτέλεση των διεργασιών με την εξής σειρά:

1. Α1 (πρώτα ολοκληρώνεται η εκτέλεση της Α1)
2. Α2 (μετά ολοκληρώνεται η εκτέλεση της Α2)
3. Β1 (μετά ολοκληρώνεται η εκτέλεση της Β1)
4. Β2 (μετά ολοκληρώνεται η εκτέλεση της Β2)
5. Α3 (τελευταία ολοκληρώνεται η εκτέλεση της Α3)

Ερώτημα Δ [15]

Σε ένα υπολογιστικό σύστημα εκτελούνται «παράλληλα» N διεργασίες, οι Process_1, Process_2, …. Process_N (δηλ. εκτελούνται σύμφωνα με το σχήμα εκτέλεσης "cobegin Process_1; Process_2; …; Process_N; coend"). Καθεμία διεργασία έχει τον ίδιο ακριβώς κώδικα (εκτελεί τις ίδιες εντολές) και όλες οι διεργασίες έχουν πρόσβαση σε δύο κοινά διαμοιραζόμενες ακέραιες μεταβλητές K και L που αρχικοποιούνται στην τιμή 1. Για παράδειγμα ο κώδικας της διεργασίας Process_i (1 <= i <= N) έχει τη μορφή:

```
shared var K = L = 1;

     Process_i
while (TRUE) {
 L:=K;
 K:=K+11;
 print_num(L, L+10);
}
```

Θεωρήστε ότι η ρουτίνα print_num() υπολογίζει (δεν χρειάζεται να σας απασχολήσει πως) και εκτυπώνει διαδοχικά τους 11 αριθμούς που υπάρχουν μεταξύ των αριθμών L και L+10. Επίσης, θεωρήστε ότι κάθε εντολή στον κώδικα των διεργασιών εκτελείται ατομικά-αδιαίρετα.

Στόχος της παράλληλης εκτέλεσης των διεργασιών είναι, κάθε φορά που εκτελείται η ρουτίνα print_num() από μια διεργασία, να εκτυπώνονται διαδοχικά οι 11 αριθμοί που ανήκουν στο επόμενο (και κάθε φορά διαφορετικό) διάστημα αριθμών. Συγκεκριμένα, το ζητούμενο αποτέλεσμα από την παράλληλη εκτέλεση των διεργασιών είναι:

- Την πρώτη φορά που θα εκτελεστεί η ρουτίνα print_num() από μια διεργασία να εκτυπωθούν διαδοχικά οι αριθμοί από το 1 έως και το 11.
- Τη δεύτερη φορά που θα εκτελεστεί η ρουτίνα print_num() από μια διεργασία να εκτυπωθούν διαδοχικά οι αριθμοί από το 12 έως και το 22.
- Την τρίτη φορά που θα εκτελεστεί η ρουτίνα print_num() από μια διεργασία να εκτυπωθούν διαδοχικά οι αριθμοί από το 23 έως και το 33.
- Την τέταρτη φορά που θα εκτελεστεί η ρουτίνα print_num() από μια διεργασία να εκτυπωθούν διαδοχικά οι αριθμοί από το 34 έως και το 44.
- ….κ.ο.κ.

(α) Κατά την παράλληλη εκτέλεση των διεργασιών ο παραπάνω κώδικας μπορεί να μην οδηγήσει στο ζητούμενο αποτέλεσμα. Να δώσετε ένα σενάριο εκτέλεσης που να το αποδεικνύει αυτό.

(β) Να συμπληρώσετε τον κώδικα των διεργασιών με εντολές wait (ή down ή P) και signal (ή up ή V) σε σημαφόρους (που πρέπει να αρχικοποιήσετε κατάλληλα), προκειμένου να εξασφαλίσετε ότι κατά την παράλληλη εκτέλεση των διεργασιών εκτυπώνεται το ζητούμενο αποτέλεσμα.

Ερώτημα Ε [20]

Πέντε διεργασίες καταφθάνουν σε ένα υπολογιστικό σύστημα σύμφωνα με τα δεδομένα του πίνακα που ακολουθεί. Σχεδιάζοντας τα κατάλληλα διαγράμματα Gantt δείξτε πώς θα εκτελεστούν οι διεργασίες αυτές στην Κεντρική Μονάδα Επεξεργασίας (ΚΜΕ), και υπολογίστε τους μέσους χρόνους διεκπεραίωσης (ΜΧΔ) και αναμονής (ΜΧΑ), για κάθε έναν από τους παρακάτω αλγόριθμους χρονοδρομολόγησης:

- FCFS (First Come First Served).

- SJF (Shortest Job First).
- SRTF (Shortest Remaining Time First).
- PS (Priority Scheduling) – μη προεκχωρητικός (non-preemptive priority).
- RR (Round Robin) με κβάντο χρόνου 4 χρονικές μονάδες.

Όνομα Διεργασίας	**Χρονική Στιγμή Άφιξης**	**Απαιτήσεις Χρόνου Εκτέλεσης**	**Προτεραιότητα**
P1	0	14	1
P2	2	5	3
P3	4	4	2
P4	7	10	5
P5	12	7	4

Παραδοχές:

1. Ο χρόνος εναλλαγής (context switch) είναι αμελητέος.
2. Για τον αλγόριθμο PS, μεγαλύτερες τιμές προτεραιότητας σηματοδοτούν μεγαλύτερες προτεραιότητες.
3. Για τον αλγόριθμο RR θεωρήστε πως αν τη χρονική στιγμή που μια νέα διεργασία έρχεται στο σύστημα διακόπτεται η εκτέλεση μιας διεργασίας (γιατί τελειώνει το κβάντο χρόνου της), τότε η νέα διεργασία εισέρχεται πριν από αυτήν που διακόπτεται στην ουρά των έτοιμων διεργασιών.

Καλή Επιτυχία!!!

Αριθμός μελών ανά ομάδα: 1 έως και 3
Ημερομηνία Παράδοσης: 21/01/2021

Αποστολή εργασιών σε ZIP ή RAR μορφή (ο συμπιεσμένος φάκελος να περιέχει txt file με τα στοιχεία της ομάδας, pdf και C files) στο e-class καθώς και με e-mail στους διδάσκοντες:

makri@ceid.upatras.gr, aristeid@ceid.upatras.gr, sioutas@ceid.upatras.gr