ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΕΠΛ 222 — ΛΕΙΤΟΥΡΓΙΚΑ ΣΥΣΤΗΜΑΤΑ (7.5 ECTS)

Ακαδημαϊκό Έτος 2017-2018, 4ο Εξάμηνο

Εξέταση Ημιεξαμήνου

Ημερομηνία : 24 Μαρτίου 2018
Διάρκεια εξέτασης : 2:15 ώρες
Διδάσκων καθηγητής : Γιώργος Α. Παπαδόπουλος

Απαντήστε όλες τις ερωτήσεις. Ο αριθμός των μονάδων της κάθε (υπο-) ερώτησης φαίνεται σε παρένθεση.

1. Θεωρείστε το ακόλουθο πρόγραμμα:

```
sema mutex=1, goB=0, goC=0;

process A()          process B()          process C()
{                    {                    {
 wait(mutex);         wait(mutex);         wait(mutex);
 …                    …                    …
 signal(goB);         wait(goB);           wait(goC);
 …                    signal(goC);         …
 signal(mutex);       …                    signal(mutex);
}                     signal(mutex);      }
                     }

parbegin A(); B(); C(); parend
```

Υπάρχει κάποια σειρά εκτέλεσης των διεργασιών έτσι ώστε:

α) Και οι τρεις διεργασίες να υποπέσουν σε αδιέξοδο; **(6%)**

β) Ακριβώς δύο από τις τρεις διεργασίες να υποπέσουν σε αδιέξοδο; **(6%)**

γ) Καμιά διεργασία να μην υποπέσει σε αδιέξοδο; **(6%)**

Ανάλογα με την απάντηση σας στο κάθε ένα από τα ερωτήματα αυτά, είτε δώστε μία σειρά εκτέλεσης των διεργασιών που να οδηγεί σε αδιέξοδο είτε εξηγείστε γιατί δεν μπορεί να υπάρξει αδιέξοδο.

2. Μερικές φορές είναι αναγκαίο να συγχρονιστούν δύο ή περισσότερες διεργασίες, έτσι ώστε όλες οι διεργασίες να ολοκληρώσουν την εκτέλεση μίας πρώτης ομάδας εντολών τους πριν αρχίσουν να εκτελούν την επόμενη ομάδα εντολών τους. Για ένα τέτοιο σενάριο με δύο διεργασίες, κάνοντας χρήση σημαφόρων θα μπορούσαμε να έχουμε τον εξής κώδικα:

```
sema s1=0, s2=0;

process A()          process B()
{                    {
 group_1;             group_1;
 signal(s1);          signal(s2);
 wait(s2);            wait(s1);
 group_2;             group_2;
}                    }

parbegin A(); B(); parend
```

Επεκτείνετε τον ανωτέρω κώδικα έτσι ώστε να επιτυγχάνεται το επιθυμητό σενάριο με εμπλοκή τριών διεργασιών. **(8%)**

3. Θεωρείστε την ακόλουθη υλοποίηση του αμοιβαίου αποκλεισμού:

```
int turn=-1;        // initially, it's nobody's turn
int want[2]={0,0}; // flag per process, initially 0

process P0()                    process P1()
{                               {
 want[0]=1;                      want[1]=1;
 turn=0;                         turn=1;
 while (want[1]&&turn==1)        while (want[0]&&turn==0)
    /* do nothing */ ;              /* do nothing */ ;
 <critical section>              <critical section>
 want[0]=0;                      want[1]=0;
}                               }

parbegin P0(); P1(); parend
```

Η υλοποίηση αυτή δεν είναι σωστή. Εξηγείστε ποιο είναι το λάθος και δείξτε το σχετικό πρόβλημα που μπορεί να δημιουργηθεί μέσω της παρουσίασης της πιθανής σειράς εκτέλεσης των εντολών των δύο διεργασιών, συμπληρώνοντας τον κατωτέρω πίνακα (προσθέστε όσες χρονικές στιγμές θεωρείτε ότι χρειάζονται για να εξηγήσετε το πρόβλημα). **(8%)**

Χρονική Στιγμή	Διεργασία P0	Διεργασία P1
0		
1	want[0]=1	
2		

4. Θεωρείστε τον ακόλουθο σκελετό ενός παρακολουθητή:

```
monitor AlarmClock()
{
 <local data structures>

 tick()
 { … }

 wakeMe(int n)
 { … }
}
```

Ο παρακολουθητής αυτός δίνει τη δυνατότητα σε μία διεργασία να θέσει τον εαυτό της υπό αναστολή για κάποιο χρονικό διάστημα, μέσω της εκτέλεσης της συνάρτησης `wakeMe`. Συγκεκριμένα, η εκτέλεση από μία διεργασία της εντολής `AlarmClock.wakeMe(10)` θα θέσει τη διεργασία αυτή υπό αναστολή για 10 μονάδες χρόνου. Για να αντιλαμβάνεται ο παρακολουθητής το πέρασμα του χρόνου, υπάρχει σε αυτόν και η συνάρτηση `tick` η οποία καλείται από το ρολόι της μηχανής (μέσω ενός διακόπτη) με το πέρασμα της κάθε μονάδας χρόνου.

Μπορείτε να θεωρήσετε ότι το σύστημα υποστηρίζει την ύπαρξη μίας παραλλαγής της εντολής `wait(cond_var,priority)` όπου π.χ. `wait(wake_up,5)` σημαίνει ότι η διεργασία που τίθεται υπό αναστολή στη μεταβλητή συνθήκης `wake_up` έχει προτεραιότητα 5. Όταν γίνει `signal` στη μεταβλητή συνθήκης `wake_up`, θα ενεργοποιηθεί η διεργασία εκείνη που έχει τη μεγαλύτερη

προτεραιότητα (προσοχή: μικρότερος αριθμός, μεγαλύτερη προτεραιότητα!).

Δώστε ολοκληρωμένο τον κώδικα του παρακολουθητή (συμπεριλαμβανομένου φυσικά και του κώδικα για τις δύο συναρτήσεις του παρακολουθητή). **(16%)**

5. Θεωρείστε τον ακόλουθο σκελετό ενός παρακολουθητή που υλοποιεί μία παραλλαγή του σεναρίου του κοιμώμενου κουρέα:

```
monitor BarberShop()
{
 <local data structures>

 get_haircut()
 { … }

 get_next_customer()
 { … }

 finished_cut()
 { … }
}
```

Το κουρείο έχει δύο πόρτες, μία για είσοδο και μία για έξοδο, και μία καρέκλα. Οι πελάτες μπαίνουν από την είσοδο και βγαίνουν από την έξοδο. Μόνο ένας πελάτης μπορεί να βρίσκεται στο κουρείο ανά πάσα στιγμή. Όταν δεν υπάρχουν πελάτες, ο κουρέας κοιμάται στην καρέκλα. Όταν μπει στο κουρείο ένας πελάτης, ο κουρέας ξυπνάει και ο πελάτης κάθεται στην καρέκλα. Αν φτάσει ένας πελάτης στο κουρείο και ο κουρέας είναι απασχολημένος με κάποιον άλλο πελάτη, ο πρώτος πελάτης περιμένει έξω από την είσοδο του κουρείου. Όταν ένας πελάτης κουρευτεί, φεύγει από την έξοδο και αν υπάρχουν πελάτες που περιμένουν στην είσοδο ο πρώτος από αυτούς ενεργοποιείται, διαφορετικά ο κουρέας ξανακοιμάται. Η συνάρτηση `get_haircut` καλείται από έναν πελάτη ο οποίος αφού την καλέσει τίθεται υπό αναστολή μέχρι να ολοκληρώσει ο κουρέας το κούρεμα. Η συνάρτηση `get_next_customer` καλείται από τον κουρέα για να εξυπηρετήσει έναν πελάτη (αν δεν υπάρχει πελάτης που περιμένει, ο κουρέας κοιμάται). Τέλος, η συνάρτηση `finished_cut` καλείται από τον κουρέα για να οδηγήσει τον πελάτη που μόλις κούρεψε στην έξοδο του κουρείου. Δώστε ολοκληρωμένο τον κώδικα του παρακολουθητή (συμπεριλαμβανομένου φυσικά και του κώδικα για τις τρεις συναρτήσεις του παρακολουθητή). Επίσης, δώστε τον κώδικα των διεργασιών `barber()` και `customer()` που καλούν τις συναρτήσεις του παρακολουθητή. **(26%)**

6. Θεωρείστε ένα σύστημα με 4 διεργασίες Δ και 3 είδη πόρων Π. Ο κατωτέρω πίνακας δείχνει για κάθε διεργασία Δi την ποσότητα μονάδων που έχει δεσμεύσει από κάθε είδος πόρων Πj και τη μέγιστη ποσότητα μονάδων που μπορεί να χρειαστεί από κάθε είδος πόρων, καθώς επίσης και τη συνολική ποσότητα μονάδων από κάθε είδος πόρων.

Διεργασία	Ποσότητα πόρων που έχουν δεσμευτεί από κάθε είδος			Μέγιστη ποσότητα πόρων που τυχόν θα χρειαστεί η διεργασία		
	Π1	Π2	Π3	Π1	Π2	Π3
Δ0	2	3	10	4	10	19
Δ1	1	6	3	5	9	5
Δ2	4	7	3	9	13	6
Δ3	2	3	1	4	5	3

Συνολική ποσότητα διαθέσιμων μονάδων για κάθε είδος πόρων

Π1	Π2	Π3
11	21	19

Με βάση τον πίνακα αυτό και κάνοντας χρήση της λογικής του αλγόριθμου του τραπεζίτη, απαντήστε τις ακόλουθες ερωτήσεις:

α) Είναι η τρέχουσα κατάσταση ασφαλής και γιατί; **(6%)**

β) Απαντήστε ξανά την ερώτηση (α) αν ο συνολικός αριθμός των μονάδων για τον πόρο `Π2` είναι 20. **(6%)**

7. Θεωρείστε την ακόλουθη πολιτική κατανομής πόρων σε διεργασίες: Οποιαδήποτε στιγμή μπορεί ένας πόρος να δεσμευθεί ή να αποδεσμευθεί από κάποια διεργασία. Αν η αίτηση κάποιας διεργασίας για πόρους δεν μπορεί να ικανοποιηθεί γιατί οι πόροι αυτοί είναι δεσμευμένοι, τότε εξετάζουμε τις διεργασίες που είναι υπό αναστολή και περιμένουν την απελευθέρωση κάποιων πόρων. Αν κάποιες από αυτές τις διεργασίες που είναι υπό αναστολή έχουν ήδη δεσμεύσει τους πόρους που χρειάζεται η πρώτη διεργασία, τότε οι αναγκαίοι πόροι αφαιρούνται από αυτές τις διεργασίες και δίνονται στην πρώτη διεργασία που τις χρειάζεται αλλιώς αυτή η διεργασία τίθεται υπό αναστολή. Όταν οι πόροι που χρειάζεται μία διεργασία υπό αναστολή καθίστανται διαθέσιμοι, η διεργασία αυτή ενεργοποιείται και εισέρχεται στην ουρά των διεργασιών που είναι έτοιμες για εκτέλεση.

Επί παραδείγματι, θεωρείστε ένα σενάριο με τρεις διεργασίες και τρεις κατηγορίες πόρων με διαθέσιμη ποσότητα `(4,2,2)`. Η διεργασία `Δ0` ζητάει και παίρνει `(2,2,1)`. Η `Δ1` ζητάει και παίρνει `(1,0,1)`. Η `Δ0` ζητάει τώρα `(0,0,1)` και τίθεται υπό αναστολή. Η `Δ2` ζητάει `(2,0,0)`, παίρνει το διαθέσιμο πόρο `(1,0,0)` αλλά με βάση την ανωτέρω πολιτική παίρνει και τον πόρο της `Δ0` η οποία είναι υπό αναστολή.

α) Με βάση αυτήν την πολιτική μπορεί να δημιουργηθεί αδιέξοδο; **(6%)**

β) Με βάση αυτήν την πολιτική μπορεί να δημιουργηθεί παρατεταμένη στέρηση; **(6%)**

Σημείωση: Στις απαντήσεις σας πρέπει να φαίνονται καθαρά οι υπολογισμοί που κάνατε για να καταλήξετε σε αυτές. Απλή αναφορά σε αποτελέσματα δεν θεωρείται απάντηση.

Καλή Επιτυχία!