ΛΕΙΤΟΥΡΓΙΚΑ ΣΥΣΤΗΜΑΤΑ

2019-2020

1η εργαστηριακή Άσκηση

Μάριος-Αλέξανδρος Μορφόπουλος 1058102
Παναγιώτης Χριστόπουλος 1054409
Χρήστος Στεμιτσιώτης 1054375
Κωνσταντίνος Μελισσουργός 1054318

Μέρος 1

Ερώτημα Α:

Η system call fork() όταν καλείται δημιουργεί ένα ακριβές αντίγραφο της διεργασίας(child) το οποίο εκτελείται παράλληλα με την αρχική διεργασία(parent). Επίσης επιστρέφει 0 στη διεργασία παιδί και στη διεργασία πατέρα τον αριθμό διεργασίας (pid) του παιδιού. Στον κώδικα του ερωτήματος 1, εκτελείται η fork 200 φορές, δημιουργούνται 200 διεργασίες parent και 200 διεργασιες child.

Ερώτημα Β-Γ-Δ:

Τα ερωτήματα Β-Γ-Δ καλύπτονται απο τα συνημμένα αρχεία κώδικα, όπου εξηγούνται αναλυτικά οι λειτουργίες τους σαν σχόλια. Παρατίθενται ενδεικτικά στιγμιότυπα από τη λειτουργία του κάθε τμήματος κώδικα:

Ερώτημα Β:

```
panos@panos-D15D:~/Desktop/ProjectOS$ ./er2
Για το child process no 1 parent process id:24610 child process id:24611
Για το child process no 2 parent process id:24610 child process id:24614
Για το child process no 3 parent process id:24610 child process id:24615
Για το child process no 4 parent process id:24610 child process id:24647
Για το child process no 5 parent process id:24610 child process id:24648
Για το child process no 6 parent process id:24610 child process id:24649
Για το child process no 7 parent process id:24610 child process id:24671
Για το child process no 8 parent process id:24610 child process id:24672
Για το child process no 9 parent process id:24610 child process id:24673
Για το child process no 10 parent process id:24610 child process id:24674
```

Ερώτημα Γ:

```
panos@panos-D15D:~/Desktop/ProjectOS$ ./er3
Δημιουργήθηκε το 1ο παιδί με pid 24757
Δημιουργήθηκε το 2ο παιδί με pid 24758
Δημιουργήθηκε το 3ο παιδί με pid 24759
Δημιουργήθηκε το 4ο παιδί με pid 24761
Δημιουργήθηκε το 5ο παιδί με pid 24763
Δημιουργήθηκε το 6ο παιδί με pid 24764
Δημιουργήθηκε το 7ο παιδί με pid 24765
Δημιουργήθηκε το 8ο παιδί με pid 24766
Δημιουργήθηκε το 9ο παιδί με pid 24767
Δημιουργήθηκε το 10ο παιδί με pid 24768
Το pid μου είναι 24767 είμαι ο πατέρας του 24768 child, ο πατέρας μου είναι 24766
Το pid μου είναι 24766 είμαι ο πατέρας του 24767 child, ο πατέρας μου είναι 24765
Το pid μου είναι 24765 είμαι ο πατέρας του 24766 child, ο πατέρας μου είναι 24764
Το pid μου είναι 24764 είμαι ο πατέρας του 24765 child, ο πατέρας μου είναι 24763
Το pid μου είναι 24763 είμαι ο πατέρας του 24764 child, ο πατέρας μου είναι 24761
Το pid μου είναι 24761 είμαι ο πατέρας του 24763 child, ο πατέρας μου είναι 24759
Το pid μου είναι 24759 είμαι ο πατέρας του 24761 child, ο πατέρας μου είναι 24758
Το pid μου είναι 24758 είμαι ο πατέρας του 24759 child, ο πατέρας μου είναι 24757
Το pid μου είναι 24757 είμαι ο πατέρας του 24758 child, ο πατέρας μου είναι 24756
Το pid μου είναι 24756 είμαι ο πατέρας του 24757 child, ο πατέρας μου είναι 24755
```

Ερώτημα Δ:

```
panos@panos-D15D:~/Desktop/ProjectOS$ ./er4
Έναρξη χρόνου 1577054084
Τελική ένδειξη ρολογιού 257792
Χρόνος εκτέλεσης 5000 διεργασιών =0.25654599999999999627
```

Μέρος 2

Ερώτημα A_1:

(Process 1)
for k = 1 to 10 do
begin
down(s1);
down(s5);
E1.1;
up(s1);
up(s5);
down(s2);
E1.2;
up(s2);
end

(Process 2)
for j = 1 to 10 do
begin
down(s1);
down(s3);
E2.1;
up(s1);
up(s3);
down(s2);
down(s4);
E2.2;
up(s2);
up(s4);
end

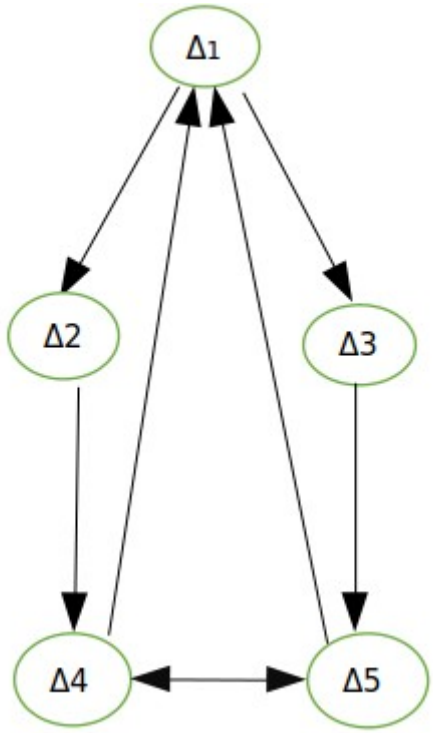
```
(Process3)
for l = 1 to 10 do
begin
down(s3);
E3.1;
up(s3);
down(s4);
down(s5);
E3.2;
up(s4);
up(s5);
end
```

Ερώτημα A_2:

```
var s1 s2 s3 s4 :semaphores;
begin
s1=1; s2=s3=0;
cobegin

(Process 1)
for k = 1 to 10 do
begin
down(s1);
E1.1;
E1.2;
up(s2);
end
 (Process 2)
for j = 1 to 10 do
begin
down(s2);
E2.1;
E2.2;
up(s3);
end
(Process 3)
for l = 1 to 10 do
 begin
 down(s3);
E3.1;
E3.2;
up(s1);
end
```

Ερώτημα B_1:



Δ1;
Cobegin
Δ2;
Δ3;
Coend
Cobegin
Δ4;
Δ5;
Coend

Η Διεργασία $Δ_1$ είναι η πρώτη που ξεκινάει να εκτελείται, στη συνέχεια με τη βοήθεια σημαφόρου ενεργοποιούνται παράλληλα οι διεργασίες $Δ_2$, $Δ_3$ οι οποιες εκτελούνται. Παράλληλα ξεκινάνε και οι $Δ_4$,$Δ_5$ με τους αντίστοιχους σημαφόρους που θα ορίσουμε στις διεργασίες $Δ_2$ και $Δ_3$. Στα κρίσιμα σημεία των $Δ_4$ και $Δ_5$ (πρόσθεση τυχαίου αριθμού) θα χρειαστεί σημαφόρος ώστε να γίνονται ξεχωριστά, με σκοπό τα buff2 και buff3 να είναι πλήρως ενημερωμένα πριν την σύγκρισή τους .

Ερώτημα B_2:

```
var s1,s2:semaphores
var shared z,y,x;
Begin
s1=s2=1;
(Process Δ1)
z = random(1,10);
write(buf1,z);
```

```
end
cobegin
(Process Δ2)
 begin
read(buf1,z);
write(buf2,z);
end
 (Process Δ3)
begin
read(buf1,z);
write(buf3,z);
end
coend
(Process Δ4)
begin
read(buf2,z);
x=z+random(1,10);
down(s1);
if x>y
then write(x);
up(s1);
end
(Process Δ5)
begin
read(buf2,z);
y=z+random(1,10);
down(s1);
if y>=x
then write(y);
up(s1);
end
coend
end
```

Ερώτημα Γ:

Α’ Τρόπος

Διεργασία Δ_0	Διεργασία Δ_1	flag0	flag1	turn
		FALSE	FALSE	0
Flag0 = TRUE		TRUE	FALSE	0
Εκτελεί το εξωτερικό while		TRUE	FALSE	0
Εισέρχεται στο ΚΡΙΣΙΜΟ ΤΜΗΜΑ		TRUE	FALSE	0
	Flag1 = TRUE	TRUE	TRUE	0
	Εκτελεί το εξωτερικό while	TRUE	TRUE	0
	Εκτελεί το εσωτερικό while	TRUE	TRUE	0
Εξέρχεται από το ΚΡΙΣΙΜΟ ΤΜΗΜΑ		TRUE	TRUE	0
Flag0 = FALSE		FALSE	TRUE	0
	Εξέρχεται από το εσωτερικό while	FALSE	TRUE	0
Flag0 = TRUE		TRUE	TRUE	0
Εκτελεί το εξωτερικό WHILE		TRUE	TRUE	0
Εισέρχεται στο ΚΡΙΣΙΜΟ ΤΜΗΜΑ		TRUE	TRUE	0
	turn=1	TRUE	TRUE	1
	Εκτέλει το εξωτερικό while	TRUE	TRUE	1
	Εισέρχεται στο ΚΡΙΣΙΜΟ ΤΜΗΜΑ	TRUE	TRUE	1

Β’ Τρόπος

Διεργασία Δ_0	Διεργασία Δ_1	flag0	flag1	turn
		FALSE	FALSE	0
Flag0 = TRUE		TRUE	FALSE	0
Εκτελεί το εξωτερικό while		TRUE	FALSE	0
Εισέρχεται στο ΚΡΙΣΙΜΟ ΤΜΗΜΑ		TRUE	FALSE	0
	Flag1 = TRUE	TRUE	TRUE	0
	Εκτελεί το εξωτερικό while	TRUE	TRUE	0
	Εκτελεί το εσωτερικό while	TRUE	TRUE	0
Εξέρχεται από το ΚΡΙΣΙΜΟ ΤΜΗΜΑ		TRUE	TRUE	0
Flag0 = FALSE		FALSE	TRUE	0
	Εξέρχεται από το εσωτερικό while	FALSE	TRUE	0
Flag0 = TRUE		TRUE	TRUE	0
Εκτελεί το εξωτερικό While		TRUE	TRUE	0
	Turn = 1	TRUE	TRUE	0
	Εκτελεί το εξωτερικό while	TRUE	TRUE	0
	Εισέρχεται στο ΚΡΙΣΙΜΟ ΤΜΗΜΑ	TRUE	TRUE	0
	Flag1 = FALSE	TRUE	FALSE	1
	Flag1 = TRUE	TRUE	TRUE	1
	Εκτελεί το εξωτερικό while	TRUE	TRUE	1
	Εισέρχεται στο ΚΡΙΣΙΜΟ ΤΜΗΜΑ	TRUE	TRUE	1
Εισέρχεται στο ΚΡΙΣΙΜΟ ΤΜΗΜΑ		TRUE	TRUE	1

Ερώτημα Δ:

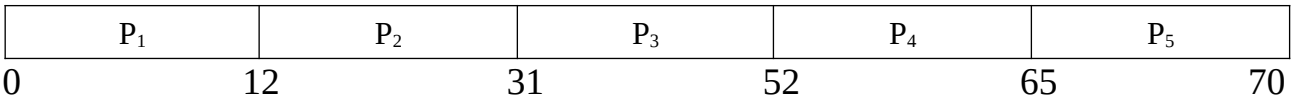
W_1	W_2	W_3	Supervisor1	Supervisor2
	wait(s) επειδή s=0 μπλοκάρει			
wait(s) επειδή s=0 μπλοκάρει				
		wait(s) επειδή s=0 μπλοκάρει		
		Ξεμπλοκάρει και ξεκινάει εργασία s=0		
				signal(s) θέτει στο s την τιμή 1
	Ξεμπλοκάρει και ξεκινάει εργασία s=0			
				signal(s) θέτει στο s την τιμή 1
Ξεμπλοκάρει και ξεκινάει εργασία s=0				
				signal(s) θέτει στο s την τιμή 1
			signal(s) s=1 signal(s) s=2 signal(s) s=3	
				wait(w) μπλοκάρει στο w
			wait(w) μπλοκάρει στο w	
		Signal w θέτει στο w=1 και αποχωρεί		
				Ξεμπλοκάρει και θέτει στο w=0 και συνεχίζει
	Signal w θέτει στο w=1 και αποχωρεί			
			Ξεμπλοκάρει και θέτει στο w=0 και συνεχίζει	
Signal w θέτει στο w=1 και αποχωρεί				
				wait(w) θέτει στο w=0 και προχωρά wait(w) επειδή το w=0 μπλοκάρει
			wait(w) επειδή το w=0 μπλοκάρει	

Από τον παραπάνω πίνακα παρατηρούμε ότι με αυτό το σενάριο που περιγράψαμε μπλοκάρουν ταυτόχρονα και οι 2 Supervisors (1-2), οπότε το σύστημα οδηγείται σε αδιέξοδο.

Ερώτημα Ε:

Διεργασία	Χρόνος Άφιξης/Εισόδου	Χρόνος Εκτέλεσης/ Διάρκεια	Προτεραιότητα
P_1	0	12	3
P_2	5	19	3
P_3	8	21	5
P_4	11	13	2
P_5	15	5	3

α) FCFS(First Come First Served/FIFO)



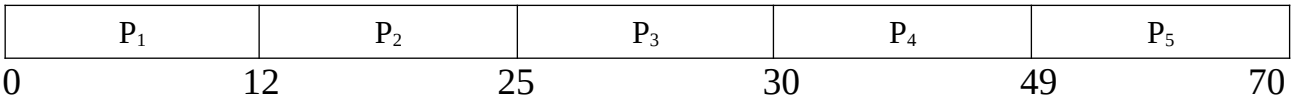
Γενικά ισχύει ότι η εκτέλεση των διεργασιών θα ολοκληρωθεί τη χρονική στιγμή 12+19+21+13+5=70.

Ο FCFS τοποθετεί τις διεργασίες στην ουρά εκτέλεσης βάσει του τρόπου που φθάνουν στο σύστημα. Έτσι, αρχικά η ουρά περιέχει τις P_1,P_2,P_3,P_4,P_5 με αυτή τη σειρά (δηλαδή η P_1 είναι η πρώτη διεργασία και η P_5 η τελευταία)

ΜΧΟ(ή ΜΧΔ)= $\frac{(12-0)+(31-5)+(52-8)+(65-11)+(70-15)}{5}$ =38,2

ΜΧΑ= $\frac{0+(12-5)+(31-8)+(52-11)+(65-15)}{5}$ =24,2

β) SJF (Shortest Job First)



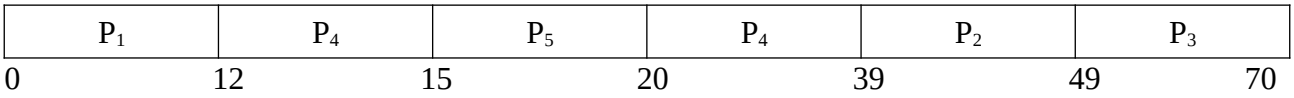
Από τη θεωρία γνωρίζουμε ότι ο αλγόριθμος χρονοδρομολόγησης SJF είναι εξ'ορισμού μη προεκτοπιστικός. Γενικά ισχύει ότι η εκτέλεση των διεργασιών θα ολοκληρωθεί τη χρονική στιγμή 12+19+21+13+5=70.

Στο μη προεκτοπιστικό SJF όταν μπαίνει μία διεργασία στην ουρά αναμονής με μικρότερη διάρκεια από αυτή που εκτελείται στη CPU, η νέα διεργασία δεν διακόπτει (δηλαδή δεν προεκτοπίζει) την εκτελούμενη διεργασία και την αφήνει να ολοκληρωθεί

$$\text{ΜΧΟ(ή ΜΧΔ)}= \frac{(12-0)+(49-5)+(70-8)+(25-11)+(30-15)}{5} =29,4$$

$$\text{ΜΧΑ}= \frac{(0-0)+(30-5)+(49-8)+(12-11)+(25-15)}{5} =15,4$$

γ) SRTF(Shortest Remaining Time First)

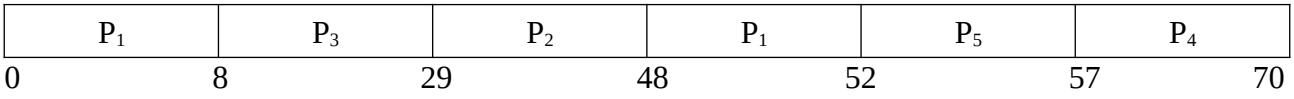


Από τη θεωρία γνωρίζουμε ότι ο αλγόριθμος χρονοδρομολόγησης SRTF είναι εξ'ορισμού προεκτοπιστικός. Γενικά ισχύει ότι η εκτέλεση των διεργασιών θα ολοκληρωθεί τη χρονική στιγμή 12+19+21+13+5=70. Ουσιαστικά ο SRTF είναι αλγόριθμος εξυπηρέτησης με βάση τη μικρότερη εναπομείνουσα διάρκεια

$$\text{ΜΧΟ(ή ΜΧΔ)}= \frac{(12-0)+(49-5)+(70-8)+(30-11)+(20-15)}{5} =28,4$$

$$\text{ΜΧΑ}= \frac{(0-0)+(30-5)+(49-8)+(12-11)+(20-15)+(15-15)}{5} =14,4$$

δ) Προεκχωρητικός Προτεραιότητας(Preemptive Priority Scheduling) με χρήση αλγορίθμου FCFS(First Come First Served) στις περιπτώσεις που έχουμε ίδιες προτεραιότητες



Από τη θεωρία γνωρίζουμε ότι ο αλγόριθμος χρονοδρομολόγησης PPS είναι εξ'ορισμού με βάση τη μεγαλύτερη προτεραιότητα. Γενικά ισχύει ότι η εκτέλεση των διεργασιών θα ολοκληρωθεί τη χρονική στιγμή 12+19+21+13+5=70.

$$\text{ΜΧΟ(ή ΜΧΔ)}= \frac{(52-0)+(48-5)+(29-8)+(70-11)+(57-15)}{5} =43,4$$

$$\text{ΜΧΑ}= \frac{(0-0)+(48-8)+(29-5)+(8-8)+(57-11)+(52-15)}{5} =29,4$$

ε) Round Robin

Το διάγραμμα Gantt είναι το ακόλουθο:

P_1	P_2	**P_1**	P_3	**P_4**	**P_5**	P_2	P_3	**P_4**	**P_2**	**P_3**

0 8 16 20 28 36 41 49 57 62 65 70

Οι διεργασίες σταματούν να εκτελούνται στην έντονη γραμματοσειρά τους.

Ο μέσος χρόνος ολοκλήρωσης είναι

$$\text{ΜΧΔ}= \frac{(20+0)+(65-5)+(70-8)+(62-11)+(41-15)}{5} =43,8$$

Ο μέσος χρόνος Αναμονής είναι:

$$\text{ΜΧΑ}= \frac{((0-0)+(16-8))+((8-5)+(41-16)+(62-49))+((20-8)+(49-28)+(65-57))+((28-11)+(57-36))}{5}$$

=29,8