ΛΕΙΤΟΥΡΓΙΚΑ ΣΥΣΤΗΜΑΤΑ
2020-2021
ΠΡΩΤΗ ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ

ΜΕΡΟΣ 1
ΕΡΩΤΗΜΑ Α:

Τρεχοντας το προγραμμα στα linux παιρνουμε το παρακατω αποτελεσμα:

Child9356 terminated with exit status 100
Child9357 terminated with exit status 101
Child9358 terminated with exit status 102
Child9359 terminated with exit status 103
Child9360 terminated with exit status 104
Child9361 terminated with exit status 105
Child9362 terminated with exit status 106
Child9363 terminated with exit status 107
Child9364 terminated with exit status 108
Child9365 terminated with exit status 109
Child9366 terminated with exit status 110
Child9367 terminated with exit status 111
Child9368 terminated with exit status 112
Child9369 terminated with exit status 113
Child9370 terminated with exit status 114
Child9371 terminated with exit status 115
Child9372 terminated with exit status 116
Child9373 terminated with exit status 117
Child9374 terminated with exit status 118
Child9375 terminated with exit status 119
Child9376 terminated with exit status 120
Child9377 terminated with exit status 121
Child9378 terminated with exit status 122
Child9379 terminated with exit status 123
Child9380 terminated with exit status 124
Child9381 terminated with exit status 125
Child9382 terminated with exit status 126
Child9383 terminated with exit status 127
Child9384 terminated with exit status 128
Child9385 terminated with exit status 129

Αρχικα μεσω της βιβλιοθηκης <sys/wait.h> καταλαβαινουμε οτι θα δουλεψουμε με συμβολικα constants για χρηση με την waitpid() συναρτηση
στην αρχη του κωδικα τρεχουμε μια λουπα για τα N=(30)

μετα γινεται η κληση της fork η οποια διασπα την τρεχουσα διεργασια σε δυο αλλες διεργασιες το γονεα και το παιδι,η κληση της καταχωρειται στο pid[i] αναλογα με το i της λουπας
υστερα με μια if κοιταμε αμα η pid[i]==0 τοτε αυτο σημαινει οτι μεσω της fork υπαρχει μια νεα

διεργασια παιδιου αν αληθευει τοτε καλουμε την sleep() function οπου εκει εισαγουμε μια αναμονη στο συστημα (στο προγραμμα μας 60-2*i) για οσα δευτερολεπτα της δωσουμε και εκετελειται η exit οπου τερματιζει την διεργασια οπου καλει τη συναρτηση
μετα τρεχουμε μια λουπα ξανα για N οπου καταχωρουμε στην pid_t wpid την συναρτηση waitpid με ορισματα το pid[i] το &child_status και το 0 δηλαδη το αποτελεσμα της συναρτησης με τα παραπανω ορισματα, με την κληση της waitpid προκαλουμε αναμονη μιας διεργασιας μεχρι το παιδι της που καθοριζεται απο το pid να τερματισει
ανοιγουμαι μια if και χρησιμοποιουμε την wifexited με ορισμα το child_status ωστε εαν δεν επιστρεψει 0 τοτε το παιδι εχει τερματιστει φυσιολογικα και θα εμφανιζει στην οθονη με printf το αναλογο μηνυμα
και με την wexitstatus με ορισμα child_status επιστρεφουμε τον κωδικο εξοδου της διαδικασιας παιδι,αυτο μπορει να υπολογιστει μονο στη περιπτωση που η wifexited επιστρεψει μη μηδεν αλλιως εμφανιζει στην οθονη με την printf το αναλογο μηνυμα.

ΜΕΡΟΣ 2

ΕΡΩΤΗΜΑ Α:

x=0
y=10
cobegin
x:=x+1;
x:=y+1;
coend

Μεταβλητη χ	Καταχωρητης Τχ	Τx
0	0+1	1
Μεταβλητη y	Καταχωρητης Ty	Ty
10	10+1	11

1η περιπτωση

E1	E2	X
x+1		1
	y+1	11

2η περιπτωση

E1	E2	X
y+1		11
	x+1	12

Οι δυνατες τιμες της μεταβλητης X μετα το τελος της παραλληλης εκτελεσης των εντολων ειναι η 11 και η 12.Αυτο διοτι:
Η τελικη τιμη X δεν μπορει να ειναι μικροτερη απο 11 διοτι αρχικα η X εχει την τιμη 0 υστερα απο την εκτελεση των εντολων στην 1η περιπτωση δεν υπαρχει καποια εντολη που να μειωνει την X κατω απο το 11
Η τελικη τιμη δεν μπορει να ξεπερασει το 12 διοτι αν εκετελεστουν μαζι οι δυο εντολες στην 2η περιπτωση τοτε θα εχουμε αποτελεσμα 12
Οι τιμες αυτες ειναι και οι μοναδικες που μπορουμε να παρουμε απο τις παραπανω εντολες
Ομως με τις περιπτωσεις ΑΝΑΜΕΙΞΗΣ (interleaving) των εντολων εχουμε ως αποτελεσμα τελικα την X να περνει τις τιμες 1 και 11, αντιστοιχα.

E1	E2	X
	X καταχωρείται σε ένα καταχωρητή TX	0
	X+1	1
Y+1		11
	η τιμή του καταχωρητή TX καταχωρείται στη μεταβλητή X	1

E1	E2	X
	Y καταχωρείται σε ένα καταχωρητή TY	10
	Y+1	11
X+1		1
	η τιμή του καταχωρητή TY καταχωρείται στη μεταβλητή X	11

ΕΡΩΤΗΜΑ Β

α)

```
var S12,S22,S23
S12:=S22:=S23:=0;
cobegin
begin D1; up(S12); end;
begin down(S12); D2; up(S22); end;
begin down(S22); D2; up(S23); end;
begin down(S23); D3; end;
coend
```

β)

```
var S12,S22,S23
S12:=S22:=S23:=0;
cobegin
repeat
begin D1; up(S12); end;
begin down(S12); D2; up(S22); end;
begin down(S22); D2; up(S23); end;
begin down(S23); D3; end;
forever;
coend
```

ΕΡΩΤΗΜΑ Γ

α)

Αρχικα ενας σηματοφορος-μετρητης περιεχει εναν ακεραιο αριθμο μη μηδενικο και υποστηριζει 2 διαδικασιες την wait (down) και την signal (up). Η signal αυξανει την τιμη του μετρητη κατα 1 και δινει αμεσως αποτελεσμα, ενω η wait θα περιμενει αν ο μετρητης δειξει 0.Αν ο μετρητης δεν ειναι 0 τοτε η wait μειωνει κατα 1 και δινει αμεσως αποτελεσμα.Αν το αποτελεσμα ενος σημαφορου-μετρητη γινει αρνητικο τοτε η διεργασια μπλοκαρει και δεν μπορει να συνεχισει παρα μονο αν μια αλλη διεργασια την ξεμπλοκαρει αυξανοντας την.

A1		
WAIT S1	S1-1	S1=1
RESULT		
SIGNAL S2	S2+1	S2=1
RESULT		

A2		
WAIT S1	S1-1	S1=0
RESULT		
POST S2	S2+1	S2=2
RESULT		

B1		
WAIT S2	S2+1	S2=1
RESULT		
WAIT S2	S2+1	S2=0
RESULT		
SIGNAL S1	S1+1	S1=1
RESULT		
SIGNAL S2	S2+1	S2=1
RESULT		

A3		
WAIT S1	S1-1	S1=0
RESULT		
SIGNAL S2	S2+1	S2=2
RESULT		

B2		
WAIT S2	S2-1	S2=1
RESULT		
WAIT S2	S2-1	S2=0
RESULT		
SIGNAL S1	S1+1	S1=1
RESULT		
SIGNAL S2	S2+1	S2=1
RESULT		

END RESULTS
S1=1
S2=1

Αποτι φαινεται και απο τα πινακακια που δηλωνουν το σεναριο εκτελεσης των διεργασιων ειναι δυνατον να συμβει η παραπανω σειρα χωρις καποιο προβλημα.Αυτο σημαινει οτι δεν μπλοκαρε καποια διεργασια και δεν αλλαξε η σειρα εκτελεσης των διεργασιων.

β)

οι διεργασιες με αριθμο εκτελεσης 1,2,3 ειναι ιδιες με τις παραπανω και δεν αλλαζει κατι οποτε συνεχιζουμε με την σειρα εκτελεση των διεργασιων 4,5 που διαφερουν απο το (α) ερωτημα

απο ερωτημα (α) εχουμε οτι:
S1=1
S2=1

B2		
WAIT S2	S2-1	S2=0
RESULT		
WAIT S2	S2-1	S2=-1 εδω εχουμε αρνητικη τιμη στον μετρητη οποτε η διεργασια μπλοκαρει
NO RESULT		
SIGNAL S1	Blocked	
SIGNAL S2	Blocked	

Η εκτελεση των διεργασιων δεν γινεται να εκτελεστει με την σειρα που μας δινετε στο ερωτημα (β) εφοσον μπλοκαρει στο βημα 4

A3		
WAIT S1	S1-1	S1=0
RESULT		
SIGNAL S2	S2+1	S2=0 εδω ξεμπλοκαρει η διεργασια B2
RESULT		

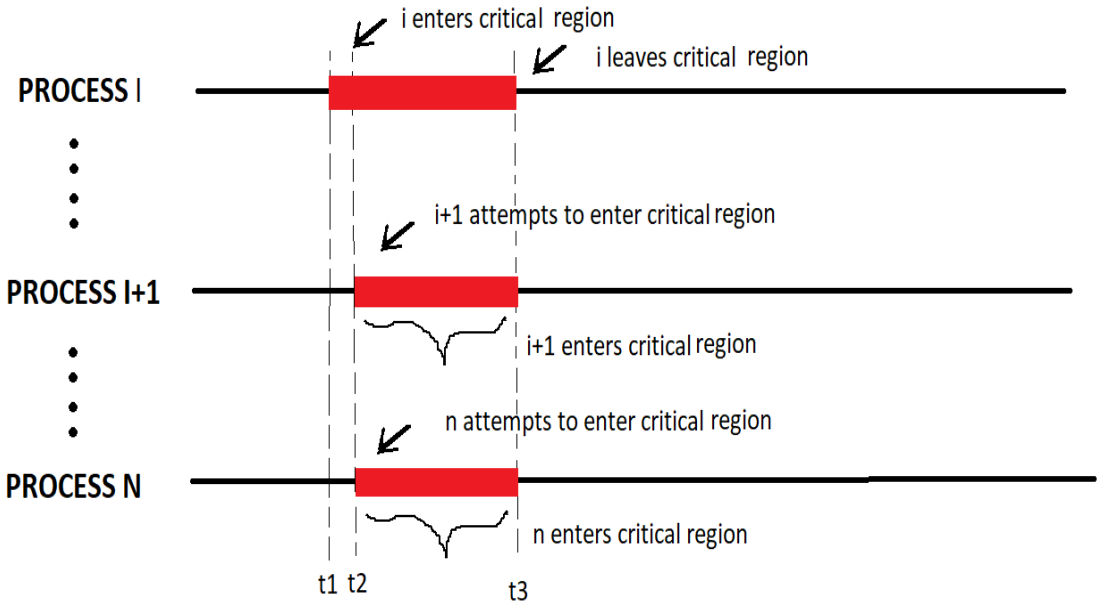
εδω συνεχιζει η διεργασια B2

B2		
SIGNAL S1	S1+1	S1=1
RESULT		
SIGNAL S2	S2+1	S2=1
RESULT		

END RESULTS
S1=1
S2=1

ΕΡΩΤΗΜΑ Δ

α)



Με βαση το σχηματικο που φτιαξαμε για να δειξουμε το πως οι διεργασιες μη εχοντας σηματοφορους για να υλοποιηθει ο αμοιβαιος αποκλεισμος συμπερανουμε οτι εφοσον οι διεργασιες τρεχουν “παραλληλα” τοτε ενω τρεχει π.χ. η διεργασια I τοτε μπορει να τρεξει και η διεργασια I+1 μαζι της και να μην εχουμε το αποτελεσμα που μας παραθεσατε στην εργασια.Για να το κανουμε περισσοτερο κατανοητο θα τρεξουμε τις διεργασιες ωστε να δειξουμε το προβλημα.

L=1
K=1

ΤΡΕΞΙΜΟ ΔΙΕΡΓΑΣΙΩΝ	APOTELESMA	ΟΘΟΝΗ
PROCESS I ENTERS		
WHILE (TRUE)		
L:=K	L=1	
K:=K+11	K=12	
PROCESS I+1 ENTERS		
L:=K	L=12	
K:K+11	K=23	
PROCESS I CONTINUES		
print_num(L,L+10)		12,13,…,22
PROCESS I FINISHES		
PROCESS I+1 CONTINUES		
print_num(L,L+10)		12,13,…,22
PROCESS I+1 FINISHES		

Αυτο ειναι ενα παραδειγμα που υποδεικνυει οτι οι διεργασιες δεν θα δωσουν το ζητουμενο αποτελεσμα

β)

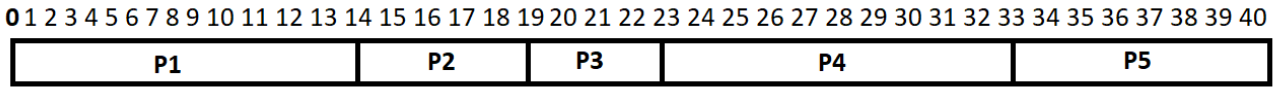
var s : semaphore;
begin
s:=1;
cobegin
shared var K=L= 1;

PROCESS I

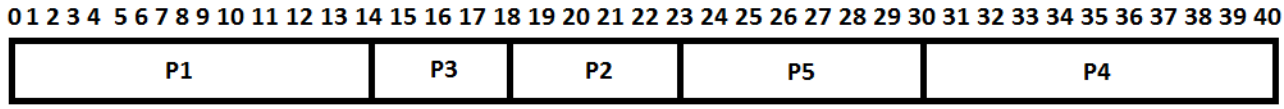
```
repeat
down(s);
while (s=0)
{
L:=K;
K:=K+11;
print_num(L,L+10);
}
up(s);
forever;
```

ΕΡΩΤΗΜΑ Ε

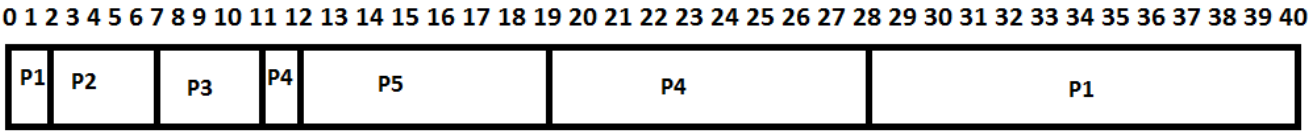
FCFS (First Come First Served) :



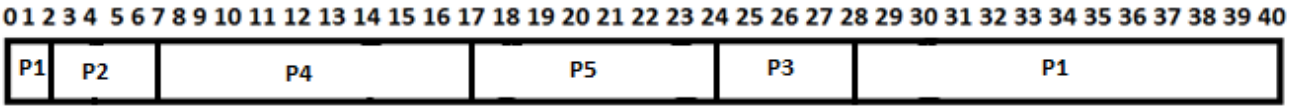
SJF (Shortest Job First) :



SRTF (Shortest Remaining Time First) :



PS (Priority Scheduling) – μη προεκχωρητικός (non-preemptive priority) :

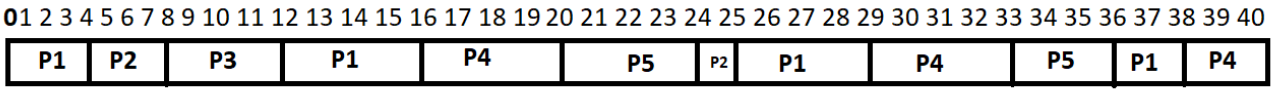


RR (Round Robin) :

τροπος σκεψης

request queue

P1=14
P2 =1
P3=0
P1=10
P4=6
P5=3
P2=0
P1=6
P4=2
P5=0
P1=2
P4=2
P1=0
P4=0



ΜΧΔ

ΔΙΕΡΓΑΣΙΑ	FCFS	SJF	SRTF	NON PREEMPTIVE PRIORITY	RR
P1	14-0=14	14-0=14	40-0=40	40-0=40	38-0=38
P2	19-2=17	23-2=21	7-2=5	7-2=5	25-2=23
P3	23-4=19	18-4=14	11-4=7	28-4=24	12-4=8
P4	33-7=26	40-7=33	28-7=21	17-7=10	40-7=33
P5	40-12=28	30-12=18	19-12=7	24-12=12	36-12=24
ΜΧΔ	104/5=20,8	100/5=20	80/5=16	91/5=18,2	126/5=25,2

ΜΧΑ

ΔΙΕΡΓΑΣΙΑ	FCFS	SJF	SRTF	NON PREEMPTIVE PRIORITY	RR
P1	14-14=0	14-14=0	40-14=26	40-14=26	38-14=24
P2	17-5=12	21-5=16	5-5=0	5-5=0	23-5=18
P3	19-4=15	14-4=10	7-4=3	24-4=20	8-4=4
P4	26-10=16	33-10=23	21-10=11	10-10=0	33-10=23
P5	28-7=21	18-7=11	7-7=0	12-7=5	24-7=17
ΜΧΑ	64/5=12,8	60/5=12	40/5=8	51/5=10,2	86/5=17,2