1η ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ

ΜΕΡΟΣ Α

Ερώτημα Α:

Στην πρώτη for έχουμε 30 διεργασίες κάθε μια από τις οποίες περιμένει 60-2i δευτερόλεπτα να τερματιστεί με exit status 100+i όπου i=0,1,2..,29 είναι ο αύξων αριθμός της διεργασίας. Στην συνέχεια η δεύτερη for προκαλεί την αναμονή κάθε μια διεργασίας μέχρι το παιδί της που καθορίζεται από το pid να τερματίσει. Μετά εάν (if) το child_status επιστρέψει τιμή μη-μηδενική σημαίνει ότι τα παιδί έχει τερματιστεί μη φυσιολογικά και επιστρέφει τον κωδικό εξόδου(exit status) της διαδικασίας παιδί, αλλιώς (else) επιστρέφει μηδέν και το παιδί έχει τερματιστεί φυσιολογικά.

Ερώτημα Β:

Αρχείο: erg1ErwtimaB.c

ΜΕΡΟΣ Β

Ερώτημα Α:

1η Περίπτωση το **X=11**

α)

```
TX := X;                // TX=0

TX := TX +1;            // TX=1
```

```
X := TX;                // X=1
                                                TY := Y;                // TY=10
                                                TY := TY +1;            // TY = 11
                                                X := TY;                // X =11
```

- Ο καταχωτητης TX παίρνει την τιμη 0
- Ο TX αυξανεται κατά 1 και η νεα τιμη του είναι 1
- Αποθηκευεται η τιμη του καταχωτητη TX στην μεταβλητη X =1
- Ο καταχωτητης TY παίρνει την τιμη 10
- Ο TY αυξανεται κατά 1 και η νεα τιμη του είναι 11
- Αποθηκευεται η τιμη του καταχωτητη TY στην μεταβλητη X =11.

β)

```
TX := X;                // TX=0
                                        TY := Y;                // TY=10
TX := TX +1;            // TX= 1
                                        TY := TY +1;            // TY =  11
X := TX;                // X=1
                                        X := TY;                // X =11
```

- Ο καταχωτητης TX παίρνει την τιμη 0
- Ο καταχωτητης TY παίρνει την τιμη 10
- Ο TX αυξανεται κατά 1 και η νεα τιμη του είναι 1

- Ο ΤΥ αυξανεται κατά 1 και η νεα τιμη του είναι 11
- Αποθηκευεται η τιμη του καταχωτητη ΤΧ στην μεταβλητη Χ =1
- Αποθηκευεται η τιμη του καταχωτητη ΤΥ στην μεταβλητη Χ =11.

2η Περίπτωση το **X=12**

```
                                    TY := Y;                // TY=10
                                    TY := TY +1;            // TY = 11
                                    X := TY;                // X =11
TX := X;                // TX=11
TX := TX +1;            // TX=12
X := TX;                // X=12
```

- Ο καταχωτητης ΤΥ παίρνει την τιμη 10
- Ο ΤΥ αυξανεται κατά 1 και η νεα τιμη του είναι 11
- Αποθηκευεται η τιμη του καταχωτητη ΤΥ στην μεταβλητη Χ =11
- Ο καταχωτητης ΤΧ παίρνει την τιμη 0
- Ο ΤΧ αυξανεται κατά 1 και η νεα τιμη του είναι 1
- Αποθηκευεται η τιμη του καταχωτητη ΤΧ στην μεταβλητη Χ =12.

3η Περίπτωση το **X=1**

```
                                        TY := Y;            // TY=10
TX := X;            // TX=0
                                        TY := TY +1;        // TY= 11
TX := TX +1;        // TX= 1
                                        X := TY;            // X=11
X := TX;            // X =1
```

- Ο καταχωτητης TY παίρνει την τιμη 10
- Ο καταχωτητης TX παίρνει την τιμη 0
- Ο TY αυξανεται κατά 1 και η νεα τιμη του είναι 11
- Ο TX αυξανεται κατά 1 και η νεα τιμη του είναι 1
- Αποθηκευεται η τιμη του καταχωτητη TY στην μεταβλητη X =11
- Αποθηκευεται η τιμη του καταχωτητη TY στην μεταβλητη X =1.

Ερώτημα Β:

(a)

```
var S1, S2, S3: semaphore;

S1=1;

S2 = S3 = 0;

    cobegin
```

```
while (TRUE) {
wait(S1);
print("P");
print("I");
signal(S2) ;
 }
```

```
while (TRUE) {
wait(S2);
print("Z");
signal(S3);
 }
```

```
while (TRUE) {
wait(S3);
signal(S2);
wait(S3);
 print("A");
}
```

```
coend
```

(b)

```
var S1, S2, S3: semaphore;
S1=1;
S2 = S3 = 0;
cobegin
```

```
while (TRUE) {
wait(S1);
print("P");
print("I");
```

```
while (TRUE) {
wait(S2);
print("Z");
signal(S3);
```

```
while (TRUE) {
wait(S3);
signal(S2);
wait(S3);
```

signal(S2) ; } print("A");

} **signal(S1);**

}

coend

Ερώτημα Γ:

(α)

A1:	A2:	B1:	A3:	B3:
down(s1);	down(s1);	down(s2);	down(s1);	down(s2);
up(s2);	up(s2);	down(s2);	up(s2);	down(s2);
		up(s1);		up(s1);
		up(s2);		up(s2);

Αρχικα s1=2,s2=0:

- εισερχεται στην διεργασια A1 και λογω της εντολης down(s1); το s1=1
- στην διεργασια A1 και λογω της εντολης up(s2); το s2=1 **η A1 ολοκληρώθηκε**
- εισερχεται στην διεργασια A2 και λογω της εντολης down(s1); το s1=0
- εισερχεται στην διεργασια B1 και λογω της εντολης down(s2); το s2=0
- στην διεργασια A2 και λογω της εντολης up(s2); το s2=1 **η A2 ολοκληρώθηκε**
- στην διεργασια B1 και λογω της εντολης down(s2); το s2=0
- στην διεργασια B1 και λογω της εντολης up(s1); το s1=1
- στην διεργασια B1 και λογω της εντολης up(s2); το s2=1 **η B1 ολοκληρώθηκε**
- εισερχεται στην διεργασια A3 και λογω της εντολης down(s1); το s1=0

- εισερχεται στην διεργασια B2 και λογω της εντολης down(s2); το s2=0
- στην διεργασια A3 και λογω της εντολης up(s2); το s2=1 **η A3 ολοκληρώθηκε**
- στην διεργασια B2 και λογω της εντολης down(s2); το s2=0
- στην διεργασια B2 και λογω της εντολης up(s1); το s1=1
- στην διεργασια B2 και λογω της εντολης up(s2); το s2=1 **η B2 ολοκληρώθηκε**

Αρα είναι δυνατο να εκτελεστουν με την παραπανω σειρα.

(β)

A1:	A2:	B1:	B3:	A3:
down(s1);	down(s1);	down(s2);	down(s2);	down(s1);
up(s2);	up(s2);	down(s2);	down(s2);	up(s2);
		up(s1);	up(s1);	
		up(s2);	up(s2);	

Αρχικα s1=2,s2=0:

- εισέρχεται στην διεργασία A1 και λογω της εντολής down(s1); το s1=1
- στην διεργασία A1 και λογω της εντολής up(s2); το s2=1 **η A1 ολοκληρώθηκε**
- εισέρχεται στην διεργασία A2 και λογω της εντολής down(s1); το s1=0
- εισέρχεται στην διεργασία B1 και λογω της εντολής down(s2); το s2=0
- στην διεργασία A2 και λογω της εντολής up(s2); το s2=1 **η A2 ολοκληρώθηκε**
- στην διεργασια B1 και λογω της εντολής down(s2); το s2=0
- στην διεργασία B1 και λογω της εντολής up(s1); το s1=1
- στην διεργασία B1 και λόγω της εντολής up(s2); το s2=1 **η B1 ολοκληρώθηκε**
- εισέρχεται στην διεργασία B2 και λόγω της εντολής down(s2); το s2=0

Ωστόσο για να εκτελέσει την επομένη εντολή down(s2); θα πρέπει πρώτα να εκτελεστεί η διεργασία A3 και να κάνει up(s2). Άρα δεν είναι δυνατό να συμβεί η παραπάνω σειρά.

Ερώτημα Δ:

(α)

Έχουμε:

shared var K = L = 1;

Process_1	Process_2	….	Process_N
while (TRUE) {	while (TRUE) {		while (TRUE) {
L:=K; K:=K+11;	L:=K; K:=K+11;		L:=K; K:=K+11;
print_num(L, L+10);	print_num(L, L+10);		print_num(L, L+10);
}	}		}

Κατά την παράλληλη εκτέλεση των διεργασιών ο κώδικας μπορεί να μην οδηγήσει στο ζητούμενο αποτέλεσμα και αυτό γιατί οι εντολές κάθε διεργασίας να μπορεί να εκτελεστούν:

Process_1 L:=K; // Άρα L=1

Process_2 L:=K; // Άρα L=1

Process_N L:=K; // Άρα L=1

Process_1 K:=K+11; // Άρα K=12

Process_2 K:=K+11; // Άρα K=12

Process_N K:=K+11; // Άρα K=12

Process_1 print_num(1, 11);

Process_2 print_num(1, 11);

Process_N print_num(1, 11);

Με αποτέλεσμα όλες να εκτυπώσουν τους αριθμούς από το 1 εώς το 11.

(β)

shared var K = L = 1;

var s1,s2,..sN: semaphore;

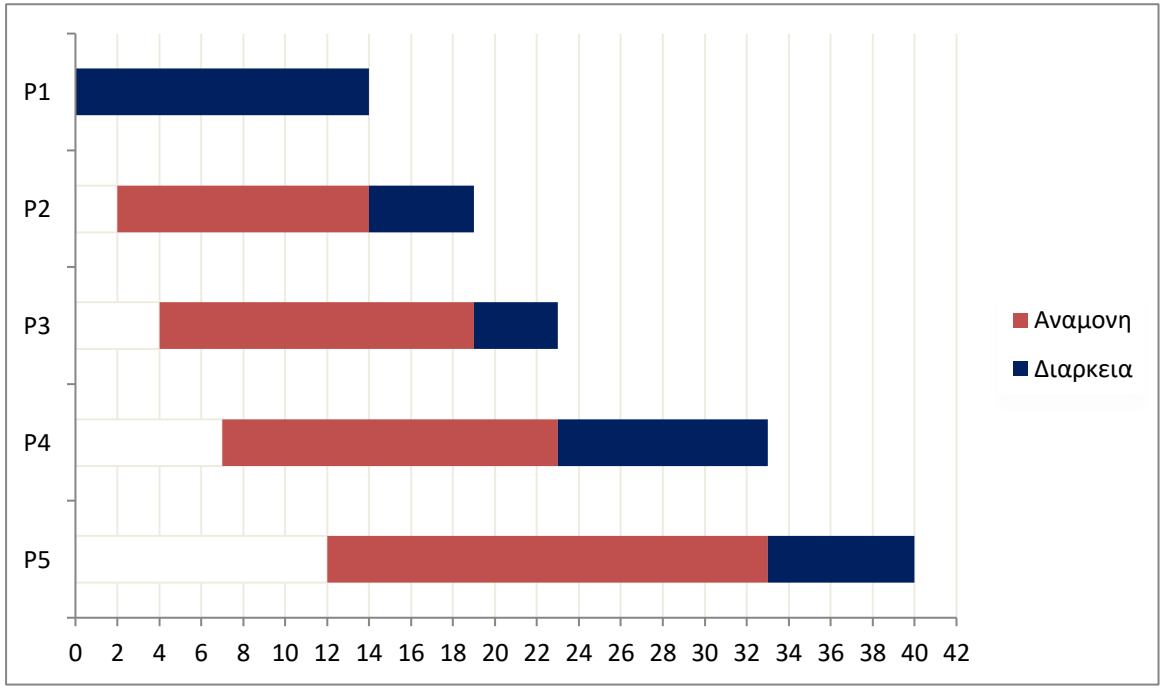
s1=1;

s2,..,sN=0;

```
 Process_i
while (TRUE) {
wait(si);
L:=K;
 K:=K+11;
print_num(L, L+10);
signal(si+1);
 }
```

Ερώτημα Ε:

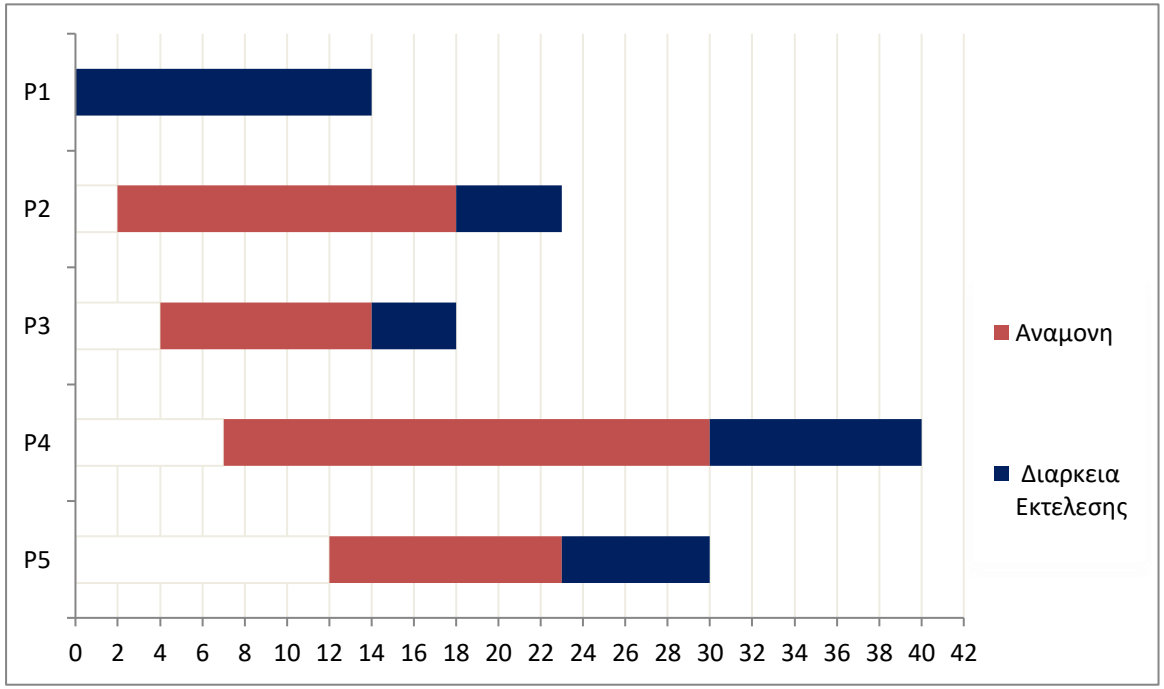
- *FCFS (First Come First Served):*



➔ Μέσος Χρόνος Διεκεραίωσης =104/5 =20,8
MXΔ(P1) =14-0 =14
MXΔ(P2) =19-2 =17
MXΔ(P3) =23-4 =19
MXΔ(P4) =33-7 =26
MXΔ(P5) =40-12 =28

➔ Μέσος Χρόνος Αναμονής=64/5=12,8
MXA(P1)=0
MXA(P2)=14-2=12
MXA(P3)=19-4=15
MXA(P4)=23-7=16
MXA(P5)=33-12=2

- *SJF (Short Job First):*

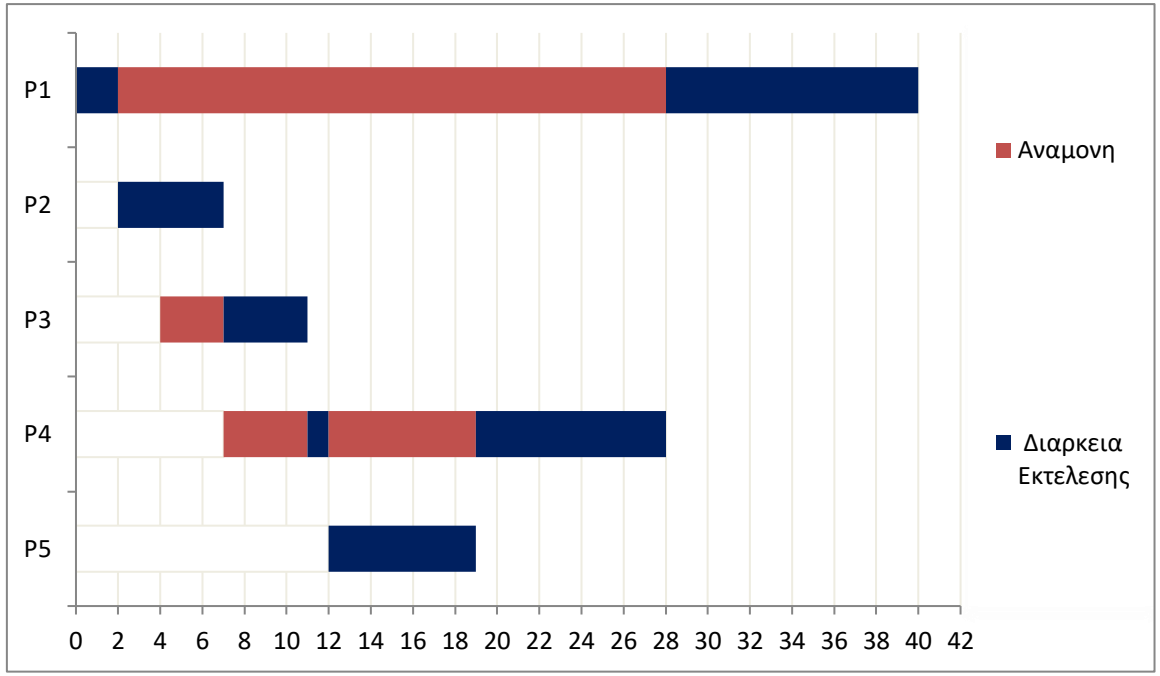


➔ Μέσος Χρόνος Διεκεραίωσης = 100/5=20
ΜΧΔ(P1) =14-0=14
ΜΧΔ(P2) =23-2=21
ΜΧΔ(P3) =18-4=14
ΜΧΔ(P4) =40-7=33
ΜΧΔ(P5) =30-12=18

➔ Μέσος Χρόνος Αναμονής=60/5= 12
ΜΧΑ(P1)=0
ΜΧΑ(P2)=18-2=16
ΜΧΑ(P3)=14-4=10
ΜΧΑ(P4)=30-7=23
ΜΧΑ(P5)=23-12=11

- *SRTF (Shortest Remaining Time First):*

1η ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ



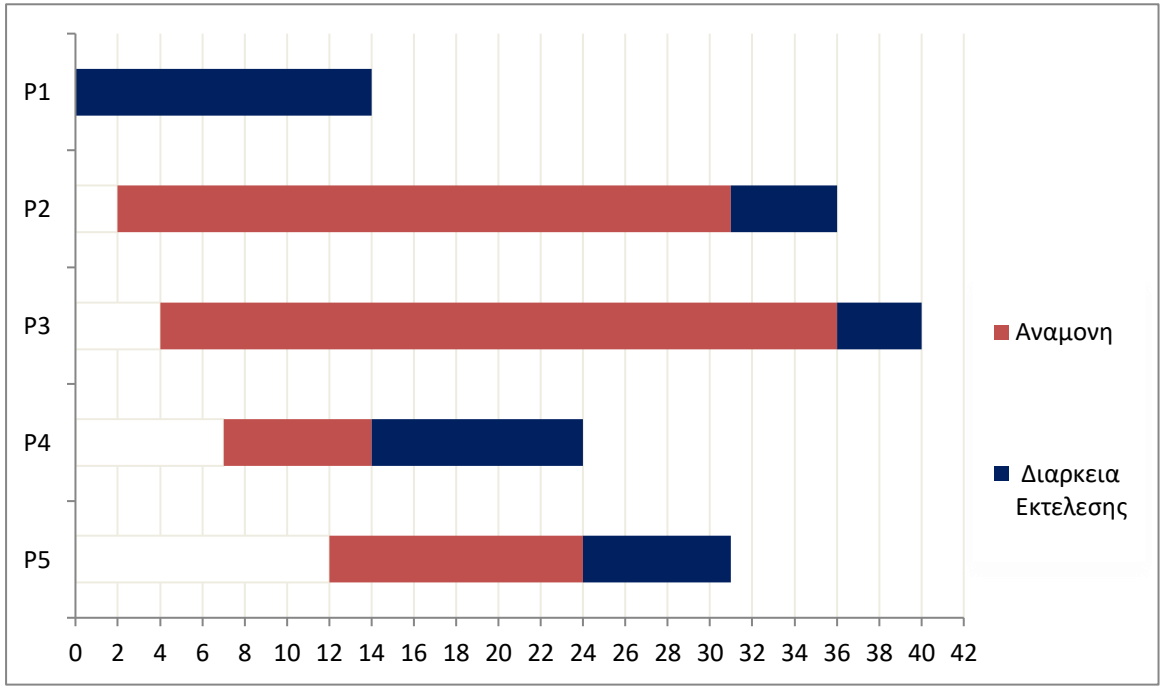
➔ Μέσος Χρόνος Διεκεραίωσης = 80/5=16
ΜΧΔ(P1) =40-0=40
ΜΧΔ(P2) =7-2=5

ΜΧΔ(P3) =11-4=7
ΜΧΔ(P4) =28-7=21
ΜΧΔ(P5) =19-12=7

➔ Μέσος Χρόνος Αναμονής= 40/5=8
ΜΧΑ(P1)=28-2=26
ΜΧΑ(P2)=0
ΜΧΑ(P3)=7-4=3
ΜΧΑ(P4)=(11-7)+(19-12)=11
ΜΧΑ(P5)=0

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

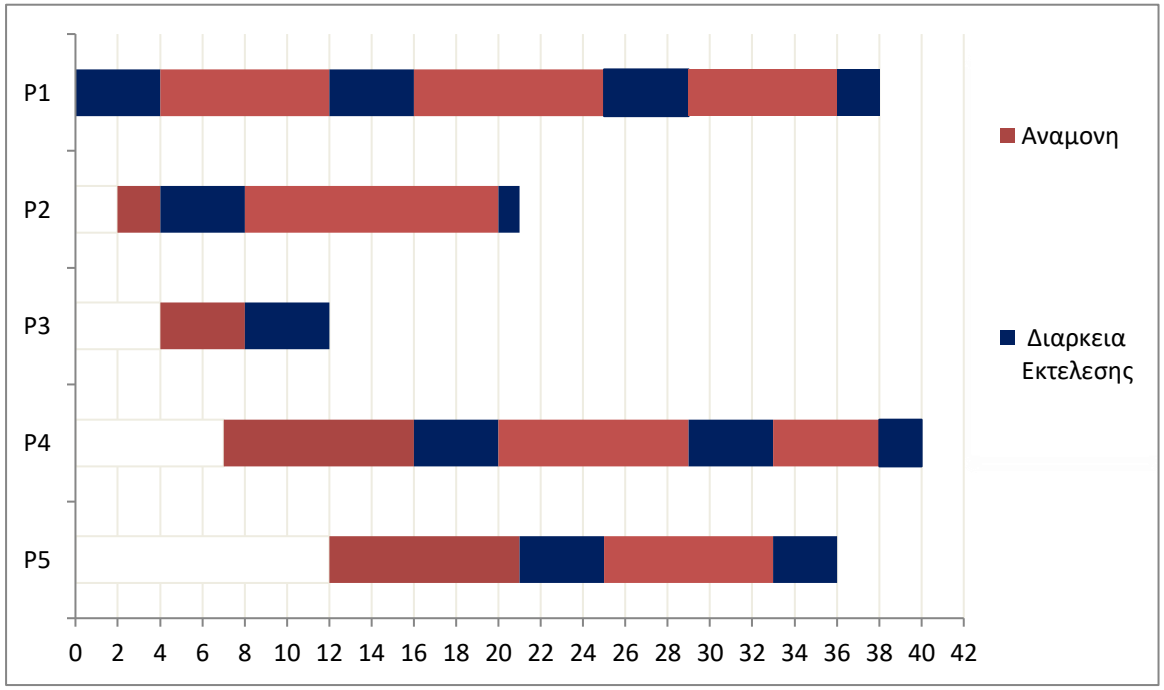
1η ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ



➔ Μέσος Χρόνος Διεκεραίωσης = 120/5=24
ΜΧΔ(P1) =14-0=14
ΜΧΔ(P2) =36-2=34
ΜΧΔ(P3) =40-4=36
ΜΧΔ(P4) =24-7=17
ΜΧΔ(P5) =31-12=19

➔ Μέσος Χρόνος Αναμονής= 80/5=16
MXA(P1)=0
MXA(P2)=31-2=29
MXA(P3)=36-4=32
MXA(P4)=14-7=7
MXA(P5)=24-12=12

- *RR (Round Robin):*



➔ Μέσος Χρόνος Διεκεραίωσης = 122/5=24.4
MΧΔ(P1) =38-0=38
MΧΔ(P2) =21-2=19
MΧΔ(P3) =12-4=8
MΧΔ(P4) =40-7=33
MΧΔ(P5) =36-12=24

➔ Μέσος Χρόνος Αναμονής= 82/5=16.4
MΧΑ(P1)=(12-4)+(25-16)+(36-29)=8+9+7=24
MΧΑ(P2)=(4-2)+(20-8)=2+12=14
MΧΑ(P3)=8-4=4
MΧΑ(P4)=(16-7)+(29-20)+(38-33)=9+9+5=23
MΧΑ(P5)=(21-12)+(33-25)=9+8=17