Λειτουργικά Συστήματα
2019 – 2020
1η Εργαστηριακή Άσκηση

Αλέξανδρος Νατσολλάρη **Α.Μ.: 1057769**
Αργύρης Μάλλιος **Α.Μ.: 1051937**
Παναγιώτης Μπαρμπούνης **Α.Μ.: 1054382**

Μέρος 1ο

Ερώτημα Α:

Η int μεταβλητή pid παίρνει τιμή από την συνάρτηση fork().Το πρόγραμμα έπειτα μπαίνει στην for όπου η fork για κάθε ένα parent (pid>0) δημιουργεί ένα νέο process child (pid=0). Οπότε τυπώνονται 400 μηνύματα συνολικά.

Ερώτημα Β:

Ένα πρόγραμμα (χρησιμοποιήσαμε c) το οποίο για μια διεργασία παράγει 10 θυγατρικές της είναι:

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
int i;
printf( "parent pid is: %d\n", getpid() );
for ( i = 0; i < 10; i++ )
   if ( fork() == 0 )
   {
      printf( "child pid is: %d parent pid is: %d\n", getpid(), getppid() );
      exit( 0 );
   }
   for (  i = 0; i < 10; i++ )
      wait( NULL );
}
```

ask1

```
parent pid is: 8263
child pid is: 8273 parent pid is: 8263
child pid is: 8271 parent pid is: 8263
child pid is: 8269 parent pid is: 8263
child pid is: 8268 parent pid is: 8263
child pid is: 8266 parent pid is: 8263
child pid is: 8264 parent pid is: 8263
child pid is: 8272 parent pid is: 8263
child pid is: 8270 parent pid is: 8263
child pid is: 8267 parent pid is: 8263
child pid is: 8265 parent pid is: 8263

Process returned 73 (0x49)   execution time : 0.024 s
Press ENTER to continue.
```

Ερώτημα Γ :

```
#include <stdio.h>
#include <stdlib.h>

int main (void)
{
   int pid;
   int i;
   int status;
   pid = fork();
   for (i=0; i <10; i++)
   {

      if (pid ==0) {
         printf("child: %d, my parent is: %d\n", getpid(), getppid());
         pid = fork();
      }
     else if (pid>0){

       wait( &status);
      }
   }

}
```

ask2

```
child: 9037, my parent is: 9036
child: 9038, my parent is: 9037
child: 9039, my parent is: 9038
child: 9040, my parent is: 9039
child: 9041, my parent is: 9040
child: 9042, my parent is: 9041
child: 9043, my parent is: 9042
child: 9044, my parent is: 9043
child: 9045, my parent is: 9044
child: 9046, my parent is: 9045

Process returned 255 (0xFF)   execution time : 0.014 s
Press ENTER to continue.
```

Ερώτημα Δ:

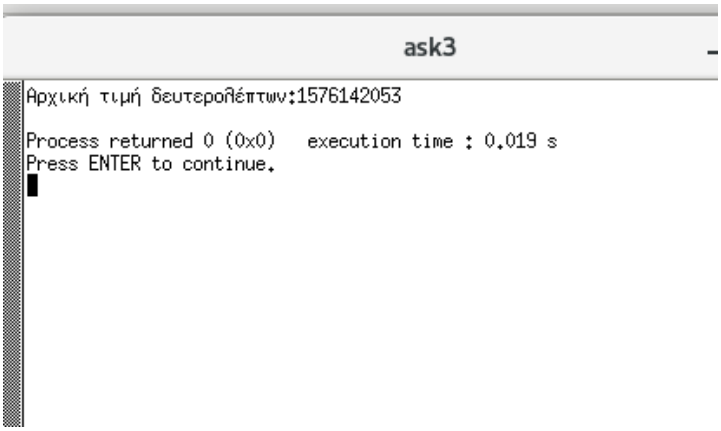
Δ1,2)

```
#include <stdio.h>
#include <time.h>

int foo(int x);

int main ()
{
   time_t seconds;
   int start;
   seconds = time(&start);
   printf("Αρχική τιμή δευτερολέπτων:%d\n", start);
   return(0);
}

int foo(int x)
{
   x=0;
   x=x+10;
}
```



Δ3)

```
#include <stdio.h>
#include <time.h>

int foo(int x);

int main ()
{
   time_t seconds;
   int start;
   int i;
   int synarthshFoo;
   int x=0;
   seconds = time(&start);
   printf("Αρχική τιμή δευτερολέπτων:%d\n", start);

   while(i<100)
   {
      if ( fork() == 0 )
         {
            printf( "child pid is: %d parent pid is: %d\n", getpid(), getppid() );
            exit( 0 );

         }
      x=foo(x);
      printf("%d synarthshFoo:%d\n",i,x);
      i++;
   }
         for (  i = 0; i < 100; i++ )
         wait( NULL );
   return(0);

}

int foo(int x)
{
   x=x+10;
   return x;
}
```

Applications Places XTerm

```
Αρχική τιμή δευτερολέπτων:1576142193
0 synarthshFoo:10
1 synarthshFoo:20
2 synarthshFoo:30
3 synarthshFoo:40
4 synarthshFoo:50
5 synarthshFoo:60
6 synarthshFoo:70
7 synarthshFoo:80
8 synarthshFoo:90
9 synarthshFoo:100
10 synarthshFoo:110
11 synarthshFoo:120
12 synarthshFoo:130
13 synarthshFoo:140
14 synarthshFoo:150
15 synarthshFoo:160
16 synarthshFoo:170
17 synarthshFoo:180
18 synarthshFoo:190
19 synarthshFoo:200
20 synarthshFoo:210
21 synarthshFoo:220
22 synarthshFoo:230
23 synarthshFoo:240
24 synarthshFoo:250
25 synarthshFoo:260
26 synarthshFoo:270
27 synarthshFoo:280
28 synarthshFoo:290
child pid is: 9957 parent pid is: 9950
child pid is: 9959 parent pid is: 9950
child pid is: 9961 parent pid is: 9950
child pid is: 9956 parent pid is: 9950
child pid is: 9954 parent pid is: 9950
child pid is: 9963 parent pid is: 9950
child pid is: 9964 parent pid is: 9950
child pid is: 9966 parent pid is: 9950
child pid is: 9953 parent pid is: 9950
child pid is: 9968 parent pid is: 9950
child pid is: 9951 parent pid is: 9950
child pid is: 9976 parent pid is: 9950
29 synarthshFoo:300
30 synarthshFoo:310
31 synarthshFoo:320
32 synarthshFoo:330
child pid is: 9971 parent pid is: 9950
33 synarthshFoo:340
child pid is: 9975 parent pid is: 9950
child pid is: 9969 parent pid is: 9950
child pid is: 9970 parent pid is: 9950
child pid is: 9960 parent pid is: 9950
child pid is: 9962 parent pid is: 9950
child pid is: 9965 parent pid is: 9950
child pid is: 9967 parent pid is: 9950
child pid is: 9972 parent pid is: 9950
child pid is: 9984 parent pid is: 9950
child pid is: 9977 parent pid is: 9950
child pid is: 9958 parent pid is: 9950
child pid is: 9979 parent pid is: 9950
child pid is: 9973 parent pid is: 9950
34 synarthshFoo:350
35 synarthshFoo:360
36 synarthshFoo:370
37 synarthshFoo:380
38 synarthshFoo:390
39 synarthshFoo:400
40 synarthshFoo:410
41 synarthshFoo:420
42 synarthshFoo:430
```

```
child pid is: 10004 parent pid is: 9950
child pid is: 10000 parent pid is: 9950
child pid is: 10012 parent pid is: 9950
child pid is: 10016 parent pid is: 9950
child pid is: 9988 parent pid is: 9950
child pid is: 10017 parent pid is: 9950
child pid is: 10023 parent pid is: 9950
child pid is: 9990 parent pid is: 9950
child pid is: 10025 parent pid is: 9950
child pid is: 10027 parent pid is: 9950
child pid is: 10028 parent pid is: 9950
child pid is: 9992 parent pid is: 9950
child pid is: 10030 parent pid is: 9950
child pid is: 9981 parent pid is: 9950
79 synarthshFoo:800
child pid is: 9994 parent pid is: 9950
80 synarthshFoo:810
81 synarthshFoo:820
82 synarthshFoo:830
83 synarthshFoo:840
84 synarthshFoo:850
child pid is: 9996 parent pid is: 9950
85 synarthshFoo:860
86 synarthshFoo:870
87 synarthshFoo:880
88 synarthshFoo:890
89 synarthshFoo:900
90 synarthshFoo:910
child pid is: 10015 parent pid is: 9950
91 synarthshFoo:920
92 synarthshFoo:930
93 synarthshFoo:940
94 synarthshFoo:950
95 synarthshFoo:960
child pid is: 10022 parent pid is: 9950
96 synarthshFoo:970
97 synarthshFoo:980
98 synarthshFoo:990
99 synarthshFoo:1000
child pid is: 10002 parent pid is: 9950
child pid is: 10003 parent pid is: 9950
child pid is: 10049 parent pid is: 9950
child pid is: 10043 parent pid is: 9950
child pid is: 10035 parent pid is: 9950
child pid is: 10036 parent pid is: 9950
child pid is: 10037 parent pid is: 9950
child pid is: 10038 parent pid is: 9950
child pid is: 10040 parent pid is: 9950
child pid is: 10046 parent pid is: 9950
child pid is: 10045 parent pid is: 9950
child pid is: 10034 parent pid is: 9950
child pid is: 9980 parent pid is: 9950
child pid is: 10033 parent pid is: 9950
child pid is: 10032 parent pid is: 9950
child pid is: 10048 parent pid is: 9950
child pid is: 10026 parent pid is: 9950
child pid is: 10024 parent pid is: 9950
child pid is: 10044 parent pid is: 9950
child pid is: 10047 parent pid is: 9950
child pid is: 10050 parent pid is: 9950
child pid is: 10029 parent pid is: 9950
child pid is: 10051 parent pid is: 9950
child pid is: 10031 parent pid is: 9950
child pid is: 10041 parent pid is: 9950
child pid is: 10042 parent pid is: 9950
child pid is: 10039 parent pid is: 9950

Process returned 0 (0x0)   execution time : 0.069 s
Press ENTER to continue.
```

Δ4)

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <unistd.h>
#include <sys/wait.h>
#include <time.h>

int foo(int x);

int main ()
{
   time_t seconds;
   int start;
   int i;
   int synarthshFoo;
   int x=0;
   int pid;
   seconds = time(&start);
   printf("Αρχική τιμή δευτερολέπτων:%d\n", start);


   while(i<100)
   {
      if ( fork() == 0 )
         {
            printf( "child pid is: %d parent pid is: %d\n", getpid(), getppid() );
            exit( 0 );
         }
      x=foo(x);
      printf("%d synarthshFoo:%d\n",i,x);
      i++;
   }
         for (  i = 0; i < 100; i++ )
         {
           waitpid(pid,NULL,0);
         }
   return(0);
}
int foo(int x)
{

   x=x+10;
   return x;
}
```

```
Αρχική τιμή δευτερολέπτων:1576142425
0 synarthshFoo:10
1 synarthshFoo:20
2 synarthshFoo:30
3 synarthshFoo:40
4 synarthshFoo:50
5 synarthshFoo:60
6 synarthshFoo:70
7 synarthshFoo:80
8 synarthshFoo:90
9 synarthshFoo:100
10 synarthshFoo:110
11 synarthshFoo:120
12 synarthshFoo:130
13 synarthshFoo:140
14 synarthshFoo:150
15 synarthshFoo:160
16 synarthshFoo:170
17 synarthshFoo:180
18 synarthshFoo:190
19 synarthshFoo:200
20 synarthshFoo:210
21 synarthshFoo:220
22 synarthshFoo:230
23 synarthshFoo:240
24 synarthshFoo:250
25 synarthshFoo:260
26 synarthshFoo:270
27 synarthshFoo:280
28 synarthshFoo:290
29 synarthshFoo:300
child pid is: 10708 parent pid is: 10697
child pid is: 10710 parent pid is: 10697
child pid is: 10712 parent pid is: 10697
child pid is: 10714 parent pid is: 10697
child pid is: 10723 parent pid is: 10697
child pid is: 10705 parent pid is: 10697
child pid is: 10707 parent pid is: 10697
child pid is: 10703 parent pid is: 10697
child pid is: 10718 parent pid is: 10697
child pid is: 10702 parent pid is: 10697
child pid is: 10709 parent pid is: 10697
child pid is: 10701 parent pid is: 10697
child pid is: 10711 parent pid is: 10697
child pid is: 10713 parent pid is: 10697
child pid is: 10699 parent pid is: 10697
child pid is: 10700 parent pid is: 10697
child pid is: 10715 parent pid is: 10697
child pid is: 10716 parent pid is: 10697
child pid is: 10720 parent pid is: 10697
child pid is: 10724 parent pid is: 10697
child pid is: 10725 parent pid is: 10697
child pid is: 10706 parent pid is: 10697
child pid is: 10727 parent pid is: 10697
child pid is: 10719 parent pid is: 10697
child pid is: 10717 parent pid is: 10697
child pid is: 10722 parent pid is: 10697
child pid is: 10704 parent pid is: 10697
child pid is: 10721 parent pid is: 10697
child pid is: 10726 parent pid is: 10697
30 synarthshFoo:310
31 synarthshFoo:320
32 synarthshFoo:330
33 synarthshFoo:340
34 synarthshFoo:350
child pid is: 10698 parent pid is: 10697
35 synarthshFoo:360
36 synarthshFoo:370
37 synarthshFoo:380
38 synarthshFoo:390
```

```
79 synarthshFoo:800
80 synarthshFoo:810
81 synarthshFoo:820
child pid is: 10732 parent pid is: 10697
82 synarthshFoo:830
83 synarthshFoo:840
84 synarthshFoo:850
child pid is: 10730 parent pid is: 10697
85 synarthshFoo:860
86 synarthshFoo:870
child pid is: 10734 parent pid is: 10697
87 synarthshFoo:880
88 synarthshFoo:890
child pid is: 10753 parent pid is: 10697
89 synarthshFoo:900
90 synarthshFoo:910
child pid is: 10755 parent pid is: 10697
91 synarthshFoo:920
child pid is: 10746 parent pid is: 10697
child pid is: 10748 parent pid is: 10697
child pid is: 10749 parent pid is: 10697
child pid is: 10750 parent pid is: 10697
child pid is: 10774 parent pid is: 10697
child pid is: 10757 parent pid is: 10697
child pid is: 10760 parent pid is: 10697
child pid is: 10775 parent pid is: 10697
child pid is: 10762 parent pid is: 10697
child pid is: 10778 parent pid is: 10697
child pid is: 10777 parent pid is: 10697
child pid is: 10764 parent pid is: 10697
child pid is: 10782 parent pid is: 10697
child pid is: 10780 parent pid is: 10697
child pid is: 10779 parent pid is: 10697
child pid is: 10784 parent pid is: 10697
child pid is: 10781 parent pid is: 10697
child pid is: 10776 parent pid is: 10697
child pid is: 10771 parent pid is: 10697
child pid is: 10772 parent pid is: 10697
child pid is: 10773 parent pid is: 10697
child pid is: 10770 parent pid is: 10697
child pid is: 10788 parent pid is: 10697
child pid is: 10783 parent pid is: 10697
child pid is: 10769 parent pid is: 10697
child pid is: 10785 parent pid is: 10697
child pid is: 10786 parent pid is: 10697
child pid is: 10768 parent pid is: 10697
child pid is: 10787 parent pid is: 10697
child pid is: 10789 parent pid is: 10697
92 synarthshFoo:930
93 synarthshFoo:940
94 synarthshFoo:950
child pid is: 10767 parent pid is: 10697
95 synarthshFoo:960
96 synarthshFoo:970
97 synarthshFoo:980
98 synarthshFoo:990
99 synarthshFoo:1000
child pid is: 10790 parent pid is: 10697
child pid is: 10793 parent pid is: 10697
child pid is: 10792 parent pid is: 10697
child pid is: 10766 parent pid is: 10697
child pid is: 10797 parent pid is: 10697
child pid is: 10794 parent pid is: 10697
child pid is: 10795 parent pid is: 10697
child pid is: 10796 parent pid is: 10697
child pid is: 10791 parent pid is: 10697

Process returned 0 (0x0)   execution time : 0.079 s
Press ENTER to continue.
```

Δ5)

Χρησιμοποιήσαμε την clock() γιατί με την time() δεν έβγαζε σωστό αποτέλεσμα στις 100 επαναλήψεις διότι οι χρόνοι εκτελέσεις είναι πολύ μικροί. Τυπώνεται Teliki timi deuteroleptwn:0.010000 second και Mesos xronos dimiourgias:0.001000 second.

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <unistd.h>
#include <sys/wait.h>
#include <time.h>

int foo(int x);

int main (){
   int i, x=0, pid;
   int synarthshFoo;
   clock_t start = clock();
   printf("Αρχική τιμή δευτερολέπτων:%d\n", start);

   while(i<100)
   {
      if ( fork() == 0 )
         {
            printf( "child pid is: %d parent pid is: %d\n", getpid(), getppid() );
            exit( 0 );
         }
      x=foo(x);
      printf("%d synarthshFoo:%d\n",i,x);
      i++;
   }
         for (  i = 0; i < 100; i++ )
         {
           waitpid(pid,NULL,0);
         }
   clock_t end = clock();
           printf("Teliki timi deuteroleptwn: %f\n",(double)(end - start) / CLOCKS_PER_SEC);
           printf("Mesos xronos dimiourgias: %f\n",((double)(end - start) /
CLOCKS_PER_SEC)/100);
   return 0;
}
int foo(int x)
{
   x=x+10;
   return x;
}
```

```
Αρχική τιμή δευτερολέπτων:0
0 synarthshFoo:10
1 synarthshFoo:20
2 synarthshFoo:30
3 synarthshFoo:40
child pid is: 11367 parent pid is: 11366
4 synarthshFoo:50
5 synarthshFoo:60
child pid is: 11370 parent pid is: 11366
6 synarthshFoo:70
7 synarthshFoo:80
8 synarthshFoo:90
9 synarthshFoo:100
10 synarthshFoo:110
11 synarthshFoo:120
12 synarthshFoo:130
13 synarthshFoo:140
14 synarthshFoo:150
15 synarthshFoo:160
16 synarthshFoo:170
17 synarthshFoo:180
18 synarthshFoo:190
19 synarthshFoo:200
20 synarthshFoo:210
21 synarthshFoo:220
22 synarthshFoo:230
23 synarthshFoo:240
24 synarthshFoo:250
25 synarthshFoo:260
26 synarthshFoo:270
27 synarthshFoo:280
28 synarthshFoo:290
29 synarthshFoo:300
30 synarthshFoo:310
31 synarthshFoo:320
32 synarthshFoo:330
33 synarthshFoo:340
34 synarthshFoo:350
child pid is: 11372 parent pid is: 11366
35 synarthshFoo:360
36 synarthshFoo:370
37 synarthshFoo:380
38 synarthshFoo:390
39 synarthshFoo:400
40 synarthshFoo:410
child pid is: 11369 parent pid is: 11366
41 synarthshFoo:420
42 synarthshFoo:430
43 synarthshFoo:440
44 synarthshFoo:450
child pid is: 11368 parent pid is: 11366
child pid is: 11373 parent pid is: 11366
child pid is: 11371 parent pid is: 11366
45 synarthshFoo:460
46 synarthshFoo:470
47 synarthshFoo:480
48 synarthshFoo:490
49 synarthshFoo:500
50 synarthshFoo:510
51 synarthshFoo:520
52 synarthshFoo:530
53 synarthshFoo:540
54 synarthshFoo:550
55 synarthshFoo:560
56 synarthshFoo:570
57 synarthshFoo:580
58 synarthshFoo:590
59 synarthshFoo:600
60 synarthshFoo:610
61 synarthshFoo:620
```

```
child pid is: 11432 parent pid is: 11366
child pid is: 11404 parent pid is: 11366
child pid is: 11437 parent pid is: 11366
child pid is: 11405 parent pid is: 11366
child pid is: 11441 parent pid is: 11366
child pid is: 11409 parent pid is: 11366
child pid is: 11433 parent pid is: 11366
child pid is: 11392 parent pid is: 11366
child pid is: 11434 parent pid is: 11366
child pid is: 11380 parent pid is: 11366
child pid is: 11435 parent pid is: 11366
child pid is: 11378 parent pid is: 11366
75 synarthshFoo:760
76 synarthshFoo:770
77 synarthshFoo:780
78 synarthshFoo:790
79 synarthshFoo:800
child pid is: 11442 parent pid is: 11366
80 synarthshFoo:810
81 synarthshFoo:820
child pid is: 11443 parent pid is: 11366
82 synarthshFoo:830
child pid is: 11444 parent pid is: 11366
83 synarthshFoo:840
84 synarthshFoo:850
85 synarthshFoo:860
child pid is: 11445 parent pid is: 11366
86 synarthshFoo:870
child pid is: 11446 parent pid is: 11366
87 synarthshFoo:880
88 synarthshFoo:890
child pid is: 11452 parent pid is: 11366
89 synarthshFoo:900
90 synarthshFoo:910
child pid is: 11453 parent pid is: 11366
91 synarthshFoo:920
92 synarthshFoo:930
child pid is: 11448 parent pid is: 11366
93 synarthshFoo:940
child pid is: 11438 parent pid is: 11366
child pid is: 11454 parent pid is: 11366
child pid is: 11440 parent pid is: 11366
child pid is: 11449 parent pid is: 11366
child pid is: 11450 parent pid is: 11366
94 synarthshFoo:950
95 synarthshFoo:960
child pid is: 11456 parent pid is: 11366
96 synarthshFoo:970
97 synarthshFoo:980
child pid is: 11457 parent pid is: 11366
98 synarthshFoo:990
99 synarthshFoo:1000
child pid is: 11458 parent pid is: 11366
child pid is: 11460 parent pid is: 11366
child pid is: 11461 parent pid is: 11366
child pid is: 11462 parent pid is: 11366
child pid is: 11464 parent pid is: 11366
child pid is: 11463 parent pid is: 11366
child pid is: 11465 parent pid is: 11366
child pid is: 11466 parent pid is: 11366
child pid is: 11459 parent pid is: 11366
child pid is: 11455 parent pid is: 11366
child pid is: 11451 parent pid is: 11366
child pid is: 11447 parent pid is: 11366
Teliki timi deuteroleptwn: 0.010000
Mesos xronos dimiourgias: 0.000100

Process returned 0 (0x0)   execution time : 0.068 s
Press ENTER to continue.
```

Μέρος 2

Ερώτημα Α:

i)

```
var s1,s2,s3.s4.s5:Semaphore;
begin
s1:=s2:=s3:=s4:=s5:=0;
cobegin
```

process1

```
for k = 1 to 10 do
begin
down(s1);
E1.1;
up(s2);

E1.2;
up(s4);
end
```

process2

```
for j = 1 to 10 do
begin
down(s2);
E2.1;
up(s3);
down(s4);
E2.2;
up(s5);
end
```

process3

```
for l = 1 to 10 do
begin
down(s3);
E3.1;

down(s5);
E3.2;
up(s1);
end
```

```
coend
end
```

ii)

```
var s1,s2,s3 :Semaphore;
int counter=0;
begin
s1:=1; s2:=s3:=0;
cobegin
```

process1

```
for k = 1 to 10 do
begin
down(s1)
if(counter==0)
{
  E1.1;
  counter+1;
}
if(counter==3)
{
  E1.2;
  counter+1;
}
up(s2)
end
```

process2

```
for j = 1 to 10 do
begin
down(s2)
if(counter==1)
{
  E2.1;
  counter+1;
}
if(counter==4)
{
  E2.2;
  counter+1;
}
up(s3)
end
```

process3

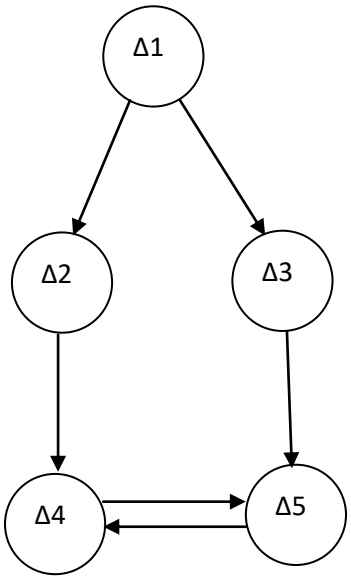
```
for l = 1 to 10 do
begin
down(s3)
if(counter==2)
{
  E3.1;
  counter+1;
}
if(counter==5)
{
  E3.2;
  counter=0;
}
up(s1)
end
```

```
coend
end
```

Ερώτημα Β:

1)

```
begin
        Δ1;
        cobegin
                begin
                        Δ2;
                        Δ4;
                end
                begin
                        Δ3;
                        Δ5;
                end
        coend
end
```



2)

```
var s1,s2,s3,s5,s6 :Semaphore;
begin
s1:=1; s2:=s3:=s4:=s5:=s6:=0;
cobegin
```

Δ1

```
down(s1);
d1=random(1...10)
;
write(buf1,d1);
up(s2);
end
```

Δ2

```
down(s2);
d2=read(buf1,d1);
write(buf2,d2);
up(s3);
end
```

Δ3

```
down(s3);
d3=read(buf1,d1);
write(buf3,d3);
up(s4)
end
```

Δ4

```
down(s4);
d4=read(buf2,d2);
w=random(1...10);
apotelesmaD4=w+d4;
up(s5);
down(s6);
if(apotelesmad4>apotelesmad5)
{
  printf("Apotelesma d4");
}
end
```

Δ5

```
down(s5);
a=random(1...10);
apotelesmad5=a+d5;
if(apotelesmad5>=apotelesmad4)
{
  printf("Apotelesma d5");
}
up(s6);
end
```

```
coend
 end
```

Ερώτημα Γ:

Δ0	Δ1	Flag0	Flag1	turn
		False	False	0
Flag0=true		True	False	0
Εκτελεί το εξωτερικό while		True	False	0
Εισέρχεται στο κρίσιμο τμήμα		True	False	0
	Flag1-true	True	True	0
	Εκτελεί το εξωτερικό while	True	True	0
	Εκτελεί το εσωτερικό while	True	True	0
	Turn=1;	True	True	1
	Βγαίνει από το εξωτερικό while	True	True	1
	Εισέρχεται στο κρίσιμο τμήμα	True	True	1

Ερώτημα Δ:

Η χρήση του δυαδικού σεμαφόρου mutex είναι σημαντική διότι εξασφαλίζει πως ο supervisor θα περιμένει να ολοκληρώσουν την εργασία τους οι 3 workers. Μπλοκάροντας έτσι οποιαδήποτε άλλη διεργασία προσπαθήσει να μπει στην κρίσιμη περιοχή προτού βγει η ιδία από αυτήν. Αφού λοιπόν χρησιμοποιείται ο σεμαφόρος mutex αυτό σημαίνει ότι μονό η διεργασία supervisor έχει τον έλεγχο στο να <<μπει>> και να <<βγει>> από την κρίσιμη περιοχή χωρίς αυτή να διακοπεί από κάποια άλλη διεργασία.

Ένα παράδειγμα είναι ότι αν δεν υπάρχει ο σεμαφόρος mutex και για κάποιο λόγο η διεργασία supervisor διακοπεί ενώ η διεργασία worker έχει στείλει 3 σήματα singal(w) δεν θα τερματίσει ποτέ. Αυτό θα γίνει διότι ο supervisor περιμένει να φύγουν οι workers έτσι ώστε να αποχωρίσει και αυτός για να τερματίσει την λειτουργιά του. Πράγμα που δεν θα γίνει ποτέ αφού θα διακοπεί η λειτουργιά του πρώτου επεξεργαστεί όλα τα σήματα που θα στείλει η διεργασία worker.

Χωρίς το mutex όταν ολοκληρωθεί μια διεργασία worker και σταλεί το singal(w) θα ολοκληρωθεί και ένα wait(w) οπότε θα ολοκληρωθεί και η διεργασία supervisor.

Ερώτημα Ε:

Διεργασία	**Χρόνος Άφιξης**	**Χρόνος Εκτέλεσης**	**Προτεραιότητα**
P1	0	12	3
P2	5	19	3
P3	8	21	5
P4	11	13	2
P5	15	5	3

α) FCFS (First Come First Served)

P1	P2	P3	P4	P5

12 31 52 65 70

Η εκτέλεση των διεργασιών θα ολοκληρωθεί την χρονική στιγμή του 70^{ου} ms.Αν t1 ο χρόνος άφιξης και t2 ο χρόνος εξόδου, ο χρόνος διεκπεραίωσης είναι ΧΟ=t2-t1 και ΧΑ=ΧΟ-tinput ο χρόνος αναμονής.

$ΧΔ_{P1}$=12-0=12 , $ΧΔ_{P2}$=31-5=26, $ΧΔ_{P3}$=52-8=44, $ΧΔ_{P4}$=65-11=54, $ΧΔ_{P5}$=70-15=55

$ΧΑ_{P1}$=12-12=0, $ΧΑ_{P2}$=26-19=7, $ΧΑ_{P3}$=44-21=23, $ΧΑ_{P4}$=54-13=41, $ΧΑ_{P5}$=55-5=50

ΜΧΔ=(12+26+44+54+55)/5=38.2 ΜΧΑ=(0+7+23+41+50)/5=24.2

β) SJF (Shortest Job First)

Non Preemptive

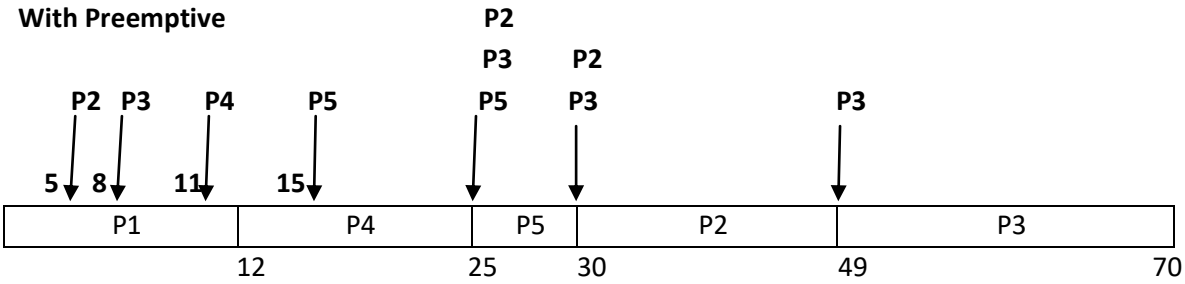
P5	P1	P4	P2	P3

5 17 30 49 70

MXΔ=(12+19+21+13+5)/5=70/5=14

MXA=(5+17+30+49+70)/5=34,2

With Preemptive



$XΔ_{P1}$=12-0=12 XA_{P1}=12-12=0

$XΔ_{P2}$=49-5=44 XA_{P2}=30-5=25

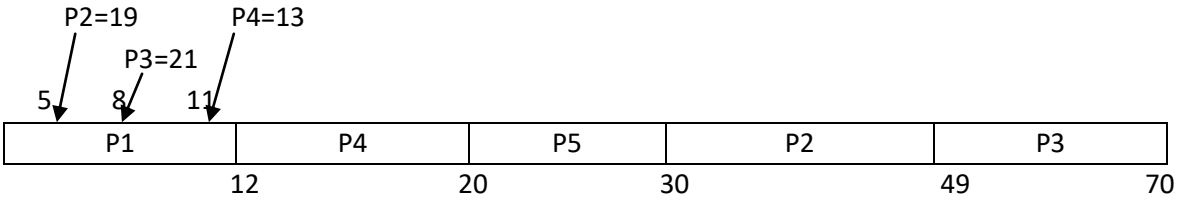
$XΔ_{P3}$=70-8=62 XA_{P3}=49-8=41

$XΔ_{P4}$=25-11=14 XA_{P4}=12-11=1

$XΔ_{P5}$=30-15=15 XA_{P5}=25-15=10

MXΔ=(12+44+62+14+15)/5=29,4 MXA=(0+25+41+1+10)/5=15,4

γ) SRTF (Shortest Remaining Time First)



$XΔ_{P1}$=12-0=12 XA_{P1}=0

$XΔ_{P2}$=49-5=44 XA_{P2}=25

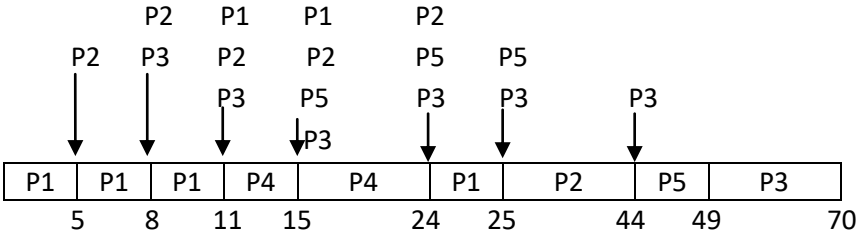
$XΔ_{P3}$=70-8=62 XA_{P3}=41

$XΔ_{P4}$=30-12=18 XA_{P4}=7

$XΔ_{P5}$=30-15=15 XA_{P5}=0

MXΔ=(12+44+62+18+15)/5=28.2 MXA=(0+25+41+7+0)/5=14.6

δ) Προεκχωρητικός Προτεραιότητας (Preemptive Priority Scheduling)



$ΧΔ_{P1}$=13+12=25 $ΧΑ_{P1}$=24-11=13

$ΧΔ_{P2}$=20+19=39 $ΧΑ_{P2}$=25-5=20

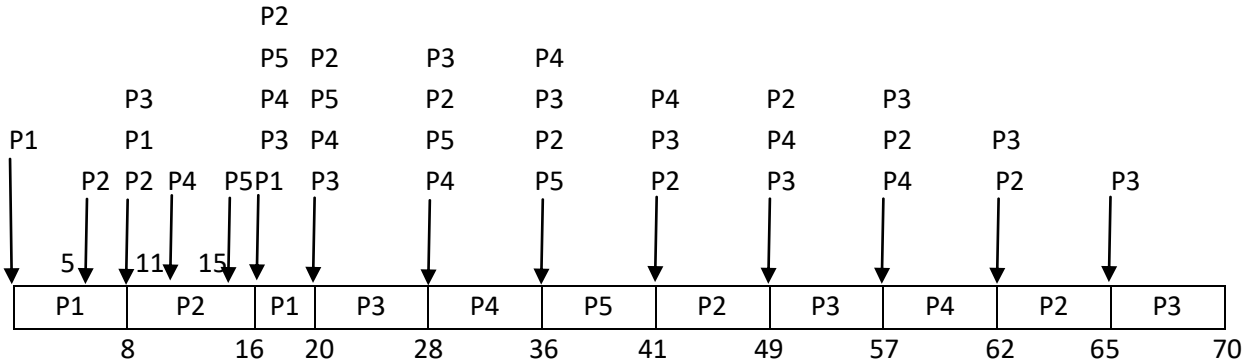
$ΧΔ_{P3}$=41+21=62 $ΧΑ_{P3}$=49-8=41

$ΧΔ_{P4}$=4+13=17 $ΧΑ_{P4}$=15-11=4

$ΧΔ_{P5}$=29+5=34 $ΧΑ_{P5}$=44-15=29

ΜΧΔ=(25+39+62+17+34)/5=35.4 ΜΧΑ=(13+20+41+4+29)/5=21.4

ε) RR (Round Robin) με κβάντο χρόνου 8 msec



$ΧΔ_{P1}$=20 $ΧΑ_{P1}$=0+8=8

$ΧΔ_{P2}$=60 $ΧΑ_{P2}$=3+4+8+8+5+8+5=41

$ΧΔ_{P3}$=60 $ΧΑ_{P3}$=8+4+8+5+8+5+3=41

$ΧΔ_{P4}$=51 $ΧΑ_{P4}$=5+4+8+5+8+8=38

$ΧΔ_{P5}$=26 $ΧΑ_{P5}$=1+4+8+8=21

ΜΧΔ=(20+60+60+51+26)/5=43.4 ΜΧΑ=(8+41+41+38+21)/5=29.8