

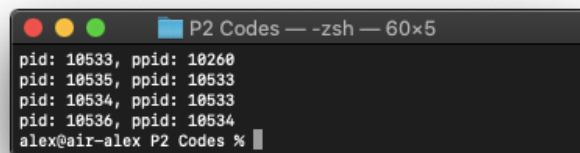
ΑΝΑΦΟΡΑ 2^{ΟΥ} PROJECT ΛΕΙΤΟΥΡΓΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΜΕΡΟΣ ΠΡΩΤΟ

Ερώτημα Α

Το πρόγραμμα που μας δίνεται αρχικοποιεί δύο ακέραιες μεταβλητές `pid1`, `pid2`. Η `pid1` καλεί την `fork()` και αν δημιουργήσει μια θυγατρική της επιτυχημένα, αντίστοιχα δημιουργείται μια θυγατρική και για την `pid2`. Ως αποτέλεσμα, 10 δευτερόλεπτα μετά την έναρξη του προγράμματος, υπάρχουν 4 διεργασίες σε κατάσταση `sleeping`, δηλαδή δύο ζευγάρια θυγατρικής-γονέα. Σε περίπτωση που έχουμε διπλή αποτυχία εκτέλεσης της `fork()`, στέλνεται μήνυμα μέσω κλήσης της `kill()` στο λειτουργικό σύστημα για να σταματήσει η εκτέλεση του προγράμματος.

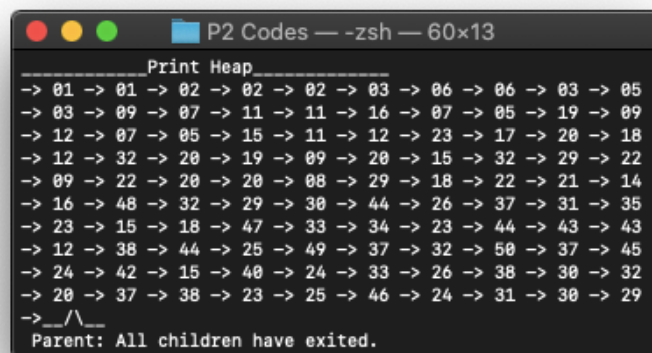
Τροποποιούμε κατάλληλα τον κώδικα έτσι ώστε και οι 4 διεργασίες να τυπώνουν το `id` τους και το `id` του γονέα τους μέσω των εντολών `getpid()` και `getppid()` αντίστοιχα λίγο πριν βρεθούν στην κατάσταση `sleeping`. Βάσει των δεδομένων που τυπώνονται, επαληθεύεται ο αρχικός ισχυρισμός μας πως έχουμε συνολικά δύο διεργασίες ζευγάρια γονέα - θυγατρικής, όπου το `id` γονέα της μιας είναι το `id` της άλλης διεργασίας. Οι κώδικες επισυνάπτονται στο `.zip` αρχείο, προφανώς.



```
P2 Codes — -zsh — 60x5
pid: 10533, ppid: 10260
pid: 10535, ppid: 10533
pid: 10534, ppid: 10533
pid: 10536, ppid: 10534
alex@air-alex P2 Codes %
```

Ερώτημα Β

Προκειμένου να πετύχουμε την προσπέλαση της `heap shared` δομής και από τις 100 θυγατρικές διεργασίες κάνουμε χρήση των συναρτήσεων της βιβλιοθήκης `sys/shm.h`. Δημιουργούμε τις διεργασίες μέσω της `fork()` και παράγουμε τυχαίες προτεραιότητες για αυτές μέσω της συνάρτησης `rand()` η οποία τις αποθηκεύει στον πίνακα `rand_priority`. Αυτές, τις κάνουμε `insert` στη `heap` δομή, την οποία τυπώνουμε κάθε φορά που κάποια διεργασία εισέρχεται στην κρίσιμη περιοχή της.



```
P2 Codes — -zsh — 60x13
Print Heap
-> 01 -> 01 -> 02 -> 02 -> 02 -> 03 -> 06 -> 06 -> 03 -> 05
-> 03 -> 09 -> 07 -> 11 -> 11 -> 16 -> 07 -> 05 -> 19 -> 09
-> 12 -> 07 -> 05 -> 15 -> 11 -> 12 -> 23 -> 17 -> 20 -> 18
-> 12 -> 32 -> 20 -> 19 -> 09 -> 20 -> 15 -> 32 -> 29 -> 22
-> 09 -> 22 -> 20 -> 20 -> 08 -> 29 -> 18 -> 22 -> 21 -> 14
-> 16 -> 48 -> 32 -> 29 -> 30 -> 44 -> 26 -> 37 -> 31 -> 35
-> 23 -> 15 -> 18 -> 47 -> 33 -> 34 -> 23 -> 44 -> 43 -> 43
-> 12 -> 38 -> 44 -> 25 -> 49 -> 37 -> 32 -> 50 -> 37 -> 45
-> 24 -> 42 -> 15 -> 40 -> 24 -> 33 -> 26 -> 38 -> 30 -> 32
-> 20 -> 37 -> 38 -> 23 -> 25 -> 46 -> 24 -> 31 -> 30 -> 29
-> __/\__
Parent: All children have exited.
```

Ερώτημα Γ

Για την υλοποίηση του προβλήματος ανάγνωσης-εγγραφής, χρησιμοποιήσαμε 9 αναγνώστες και 4 εγγραφείς με τη μορφή νημάτων `pthread`s. Επίσης, προκειμένου να εξασφαλιστεί ο ζητούμενος συγχρονισμός χρησιμοποιήσαμε έναν δυαδικό σημαφόρο `mutex` ο οποίος εξασφαλίζει μοναδική πρόσβαση για κάθε αναγνώστη στη μεταβλητή `readcount`, η οποία μετράει τον αριθμό των αναγνωστών και ενημερώνει αντίστοιχα τον εγγραφέα μέσω του σημαφόρου `wsem`. Επιπλέον χρησιμοποιούμε μια `public` μεταβλητή `counter` που εξυπηρετεί τον ρόλο του διαμοιραζόμενου αρχείου, με αρχική τιμή 1, το περιεχόμενο της οποίας διπλασιάζεται κάθε φορά που επιχειρείται μια νέα εγγραφή και οι αναγνώστες αντίστοιχα διαβάζουν το περιεχόμενό της.

Παρατηρούμε κατά την εκτέλεση του προγράμματος πως οι αναγνώσεις όντως γίνονται πολλές ταυτόχρονα, ενώ στην περίπτωση της εγγραφής για να επιχειρηθεί μια νέα εγγραφή πρέπει να έχει ολοκληρωθεί η προηγούμενη.

```
Writer 1 is created
Writer 2 is created
Writer 3 is created
Writer 4 is created
Reader 4 is reading

Reader 4 is reading
Reader 2 is reading
Reader 3 is reading

Reader 4 finished reading,
counter value: 1

Reader 2 finished reading,
counter value: 1

Reader 3 finished reading,
counter value: 1

Writer 2 is writing

Writer 2 finished writing,
counter value: 2

Writer 3 is writing

Writer 3 finished writing,
counter value: 4
```

Ερώτημα Δ

Για την υλοποίηση των δύο ζητούμενων συγχρονισμών διεργασιών, χρησιμοποιήσαμε νήματα `pthread`s για την παράλληλη εκτέλεση κάθε διεργασίας. Οι διεργασίες δημιουργούνται μέσω της `pthread_create()`, η οποία δημιουργεί `threads` που τρέχουν οι P1, P2, P3, P4, P5 (και P6 στο δεύτερο σκέλος), και μέσω της `pthread_join()`, η οποία επιτρέπει την παράλληλη εκτέλεση τους. Κάθε διεργασία κλειδώνει και ξεκλειδώνει τους αντίστοιχους σημοφόρους, σύμφωνα με το γράφημα και τους περιορισμούς, όπου για κάθε σχέση προτεραιότητας Δi -> Δj αντιστοιχίζεται και ένας σημαφόρος.

(1)

```
P2 Codes — -zsh — 60x28

** Entered P1
total 128
-rwxr-xr-x 1 alex staff 12772 13 Iav 17:37 erA
-rw-r--r--@ 1 alex staff 396 13 Iav 17:36 erA.c
-rwxr-xr-x 1 alex staff 13664 13 Iav 17:38 erC
-rw-r--r--@ 1 alex staff 1698 13 Iav 17:38 erC.c
-rwxr-xr-x 1 alex staff 13196 13 Iav 17:54 erD1
-rw-r--r--@ 1 alex staff 1943 13 Iav 17:54 erD1.c
-rw-r--r--@ 1 alex staff 2531 13 Iav 17:49 erD2.c
** Exiting P1
** Entered P2
  UID  PID  PPID      F CPU PRI NI      SZ  RSS  WCHAN
  S      ADDR TTY      0  31  0  4297436 2284 -
  S      0 ttys000 0:00.24 -zsh
  S01 12056 10260 4006 0 31 0 4277888 676 -
  S+      0 ttys000 0:00.00 ./erD1
** Exiting P2
** Entered P3
17:54 up 2:50, 2 users, load averages: 1,92 1,84 1,79
** Exiting P3
** Entered P4
erA  erA.c  erC  erC.c  erD1  erD1.c  erD2.c
** Exiting P4
** Entered P5
alex
** Exiting P5
alex@air-alex P2 Codes %
```

(2)

```
P2 Codes — -zsh — 60x34

** Entered P1
total 160
-rwxr-xr-x 1 alex staff 12772 13 Iav 17:37 erA
-rw-r--r--@ 1 alex staff 396 13 Iav 17:36 erA.c
-rwxr-xr-x 1 alex staff 13664 13 Iav 17:38 erC
-rw-r--r--@ 1 alex staff 1698 13 Iav 17:38 erC.c
-rwxr-xr-x 1 alex staff 13196 13 Iav 17:54 erD1
-rw-r--r--@ 1 alex staff 1943 13 Iav 17:54 erD1.c
-rwxr-xr-x 1 alex staff 13228 13 Iav 17:56 erD2
-rw-r--r--@ 1 alex staff 2393 13 Iav 17:56 erD2.c
** Exiting P1
** Entered P2
  UID  PID  PPID      F CPU PRI NI      SZ  RSS  WCHAN
  S      ADDR TTY      0  31  0  4297436 2284 -
  S      0 ttys000 0:00.26 -zsh
  S01 12174 10260 4006 0 31 0 4268672 656 -
  S+      0 ttys000 0:00.01 ./erD2
** Exiting P2
** Entered P3
17:56 up 2:52, 2 users, load averages: 1,88 1,83 1,78
** Exiting P3
** Entered P4
erA  erC  erD1  erD2
erA.c erC.c erD1.c erD2.c
** Exiting P4
** Entered P5
alex
** Exiting P5
** Entered P6
/Users/alex/Documents/CEID/5o ΕΕΑΜΗΝΟ/ΛΕΙΤΟΥΡΓΙΚΑ ΣΥΣΤΗΜΑΤΑ/
PROJECT 2/P2 Codes
** Exiting P6
alex@air-alex P2 Codes %
```

ΜΕΡΟΣ ΔΕΥΤΕΡΟ**Ερώτημα Α**

α)

Γνωρίζουμε ότι η λογική διεύθυνση είναι των 32 bits και ότι τα πρώτα 18 bits αναπαριστούν τον αριθμό σελίδας p . Συνεπώς, τα bits της μετατόπισης d είναι $32 - 18 = 14$. Άρα το μέγεθος σελίδας είναι 2^{14} bytes = 16384 bytes.

Από τα δεδομένα της εκφώνησης, μία διεργασία που είναι ολόκληρη φορτωμένη στη μνήμη αποτελείται από 39500_{16} bytes = 234.752 bytes. Έχουμε $234.752 = 14 \cdot 16.384 + 5.376$.

Συνεπώς, η διεργασία καταλαμβάνει 14 σελίδες (των 16.384 bytes η κάθε μία) και 5.376 bytes επιπλέον, άρα συνολικά 15 σελίδες. Οπότε θα διεκδικήσει από τη μονάδα διαχείρισης μνήμης (MMU) 15 πλαίσια για να χωρέσουν όλες οι σελίδες της. Στην 15η σελίδα, μόνο τα 5.376 bytes από τα 16.384 bytes θα χρησιμοποιηθούν. Τα υπόλοιπα 11.008 bytes μένουν αχρησιμοποίητα, που είναι και το μέγεθος της εσωτερικής κλασματοποίησης που προκαλεί η διεργασία.

β)

Βάσει των δεδομένων, δημιουργούμε τον παρακάτω πίνακα σελίδων:

Αριθμός Σελίδας	Αριθμός Πλαισίου
...	...
10	16
11	225
12	170
13	35
14	51

- **00031958₁₆**

Μετατρέπουμε τη διεύθυνση στο δυαδικό και διαχωρίζουμε τον αριθμό σελίδας και τη μετατόπιση ακολούθως:

$00031958 = 0000\ 0000\ 0000\ 0011\ 0001\ 1001\ 0101\ 1000$

Αριθμός σελίδας (πρώτα 18 bits) $\rightarrow 0000\ 0000\ 0000\ 0011\ 00 = 12$

Μετατόπιση (τελευταία 14 bits) $\rightarrow 01\ 1001\ 0101\ 1000 = 1958_{16}$

Με βάση των πίνακα σελίδων, η λογική σελίδα 12 αντιστοιχεί στο πλαίσιο 170, που στο δεκαεξαδικό σύστημα είναι ο αριθμός AA_{16} .

Άρα η τελική φυσική διεύθυνση είναι η $AA1958_{16}$.

- **0001E800₁₆**

Ακολουθούμε την ίδια διαδικασία με προηγουμένως:

$0001E800 = 0000\ 0000\ 0000\ 0001\ 1110\ 1000\ 0000\ 0000$

Αριθμός σελίδας $\rightarrow 0000\ 0000\ 0000\ 0001\ 11 = 7$

Μετατόπιση $\rightarrow 10\ 1000\ 0000\ 0000 = 2800_{16}$

Με βάση τον πίνακα σελίδων, η λογική σελίδα 7 δεν είναι φορτωμένη στη μνήμη, άρα έχουμε σφάλμα σελίδας (page fault).

Ερώτημα Β

α)

Με βάση τα δεδομένα της εκφώνησης έχουμε:

Λογική διεύθυνση $n = 32$ bits

Μέγεθος τμήματος $16 \text{ Mbytes} = 16 \cdot 2^{20} \text{ bytes} = 2^4 \cdot 2^{20} \text{ bytes} = 2^{24} \text{ bytes}$

Συνεπώς, μπορούμε να συμπεράνουμε πως η μετατόπιση d αποτελείται από $k = 24$ bits και ο αριθμός τμήματος s αποτελείται από $n - k = 32 - 24 = 8$ bits.

Άρα ο μέγιστος υποστηριζόμενος αριθμός τμημάτων για μια διεργασία ισούται με $2^{n-k} = 2^8 = 256$ τμήματα.

β)

- **0B00042A₁₆**

Μετατρέπουμε τη διεύθυνση στο δυαδικό και διαχωρίζουμε τον αριθμό τμήματος και τη μετατόπιση ακολούθως:

$0B00042A = 0000 \ 1011 \ 0000 \ 0000 \ 0000 \ 0100 \ 0010 \ 1010$

Αριθμός τμήματος $\rightarrow 0000 \ 1011 = 11$

Μετατόπιση $\rightarrow 0000 \ 0000 \ 0000 \ 0100 \ 0010 \ 1010 = 1066$

Για να βρούμε τη φυσική διεύθυνση, προσθέτουμε στη διεύθυνση βάσης του αντίστοιχου τμήματος τη μετατόπιση:

[11, 1066] : $9050 + 1066 = 10116$, που στο δεκαεξαδικό σύστημα είναι η διεύθυνση με αριθμό 2784_{16} .

- **02000B6D₁₆**

Ακολουθούμε την ίδια διαδικασία με προηγούμενως:

$02000B6D = 0000 \ 0010 \ 0000 \ 0000 \ 0000 \ 1011 \ 0110 \ 1101$

Αριθμός τμήματος $\rightarrow 0000 \ 0010 = 2$

Μετατόπιση $\rightarrow 0000 \ 0000 \ 0000 \ 1011 \ 0110 \ 1101 = 2925$

Άρα, **[2, 2925]** : $10310 + 2925 = 13235$, που στο δεκαεξαδικό είναι η διεύθυνση $33B3_{16}$.

Παρατηρούμε πως ο διεύθυνση που υπολογίσαμε υπερβαίνει το όριο του τμήματος, άρα έχουμε σφάλμα τμήματος.

Ερώτημα Γ

α)

Με βάση τα δεδομένα της εκφώνησης, γνωρίζουμε ότι το μέγεθος σελίδας είναι $512 \text{ bytes} = 2^9 \text{ bytes}$. Από αυτό συμπεραίνουμε πως η μετατόπιση σελίδας d' αποτελείται από $k' = 9$ bits.

Συνεπώς, ο αριθμός σελίδας p αποτελείται από

$j = k - k' = 24 - 9 = 15$ bits.

Άρα η μορφή κάθε λογικής διεύθυνσης είναι η εξής:

Αριθμός τμήματος s 8 bits	Αριθμός σελίδας p 15 bits	Μετατόπιση σελίδας d' 9 bits
--------------------------------	--------------------------------	-----------------------------------

Λογική διεύθυνση 32 bits

β)

Πλήθος σελίδων $= 2^{n-k'} = 2^{32-9} = 2^{23}$ σελίδες.

γi)

Αρχικά μετατρέπουμε τη διεύθυνση στο δυαδικό και διαχωρίζουμε τον αριθμό τμήματος, τον αριθμό σελίδας και την μετατόπιση:

$010004CF = 0000 \ 0001 \ 0000 \ 0000 \ 0000 \ 0100 \ 1100 \ 1111$

Αριθμός τμήματος $\rightarrow 0000\ 0001 = 1$

Αριθμός σελίδας $\rightarrow 0000\ 0000\ 0000\ 010 = 2$

Μετατόπιση $\rightarrow 0\ 1100\ 1111 = 0CF_{16}$

Με βάση τον πίνακα σελίδων του τμήματος 1, η λογική σελίδα 2 αντιστοιχεί στο πλαίσιο 0B0B₁₆, άρα η τελική φυσική διεύθυνση είναι η 0B0B0CF₁₆.

γii)

Η λογική διεύθυνση 010009FF₁₆ προκαλεί σφάλμα σελίδας

Αφού προκαλεί σφάλμα σελίδας, σημαίνει πως η συγκεκριμένη σελίδα δεν είναι φορτωμένη στη μνήμη. Στη συνέχεια, τη μετατρέπουμε στο δυαδικό:

010009FF = 0000 0001 0000 0000 0000 1001 1111 1111

Αριθμός τμήματος $\rightarrow 0000\ 0001 = 1$

Αριθμός σελίδας $\rightarrow 0000\ 0000\ 0000\ 100 = 4$

Η λογική διεύθυνση 000003F0₁₆ αντιστοιχεί στη φυσική E0E1F0₁₆

000003F0 = 0000 0000 0000 0000 0000 0011 1111 0000

Αριθμός τμήματος $\rightarrow 0000\ 0000 = 0$

Αριθμός σελίδας $\rightarrow 0000\ 0000\ 0000\ 001 = 1$

Μετατόπιση $\rightarrow 1\ 1111\ 0000 = 1F0_{16}$

Από την αντίστοιχη φυσική διεύθυνση που μας δίνεται, έχουμε ότι η μετατόπιση 1F0 είναι ο αριθμός που μόλις υπολογίσαμε, άρα συμπεραίνουμε πως ο αριθμός πλαισίου είναι ο E0E.

Παρακάτω, παρουσιάζουμε τους δύο Π.Σ. που προκύπτουν βάσει των προηγούμενων υπολογισμών:

Πίνακας Σελίδων Τμήματος 0	
Αριθμός Σελίδας	Αριθμός Πλαισίου
0	151F
1	E0E
2	34FE
3	7E11
4	2345
5	-
...	...

Πίνακας Σελίδων Τμήματος 1	
Αριθμός Σελίδας	Αριθμός Πλαισίου
0	-
1	2EE1
2	0B0B
3	0C11
4	-
5	1BA2
...	...

Ερώτημα Δ

	3	5	8	1	8	7	5	1	8	2	4	2	7	3	6	4	7	5	3	7
0	3	3	3	3	3	7	7	7	7	2	2	2	2	2	2	4	4	4	4	4
1	-	5	5	5	5	5	5	5	5	5	4	4	4	4	6	6	6	6	3	3
2	-	-	8	8	8	8	8	8	8	8	8	8	8	3	3	3	3	5	5	5
3	-	-	-	1	1	1	1	1	1	1	1	1	7	7	7	7	7	7	7	7
	F	F	F	F		F				F	F		F	F	F	F		F	F	

Στοιχεία ομάδας:

Αλέξανδρος Ξιάρχος
 Νικηφόρος – Γεώργιος Παπαγεωργίου
 Παναγιώτης Συριόπουλος

1059619
 1059633
 1059664

st1059619@ceid.upatras.gr
 st1059633@ceid.upatras.gr
 st1059664@ceid.upatras.gr