Λειτουργικά Συστήματα

2018 - 2019

Εργαστηριακή Άσκηση

Μέρος 1 [70 μονάδες]

Ερώτημα Α [5]: Εξηγήστε προσεκτικά τι κάνει το παρακάτω πρόγραμμα.

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

int main()
{
    int pid;
    int x,y;

    x = 10;
    y = 10;

    pid = fork();

    if (pid != 0)
    {
       x++;
       y--;
    }

    printf("x = %i y = %i\n",x,y);


    pid = fork();

    if (pid != 0)
    {
       x++;
       y--;
    }

    printf("x = %i y = %i\n",x,y);



    return (0);
}
```

Ερώτημα Β [5]:: Δημιουργείστε ένα πρόγραμμα στο οποίο μία διεργασία στο Linux/Unix παράγει άλλες 4 θυγατρικές της.

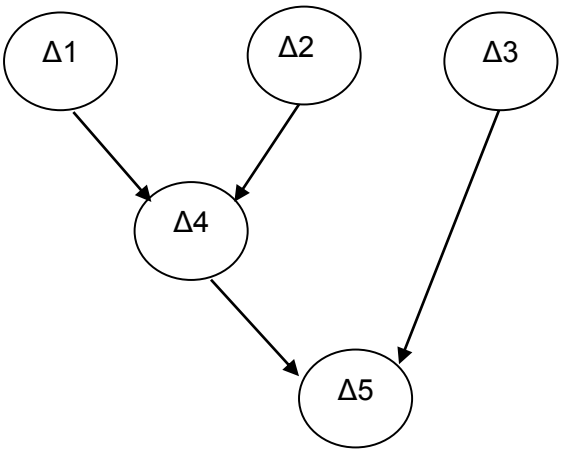
Ερώτημα Γ [5]: Να δημιουργήσετε ένα πρόγραμμα στο οποίο να δημιουργούνται 5 διεργασίες (αλυσίδα – κάθε μία να είναι παιδί της άλλης). Κάθε διεργασία να τυπώνει, το id του πατέρα της, το id της και το id του μοναδικού παιδιού που δημιουργεί.

Ερώτημα Δ [15]: Να δημιουργήσετε πρόγραμμα στο οποίο:

1. Θα ορίσετε μια συνάρτηση nothing() η οποία θα ορίζει μια μεταβλητή int x=0 και θα κάνει την πράξη x=x+1. Αυτή η συνάρτηση δεν κάνει τίποτα χρήσιμο, αλλά απλώς υπάρχει για να μας βοηθήσει στη μέτρηση.
2. Μέσα στη main() θα εκτελεί μια φορά τη συνάρτηση time() με τις κατάλληλες παραμέτρους, θα αποθηκεύεται ο αριθμός των δευτερολέπτων στη μεταβλητή start και αμέσως μετά θα εκτυπώνεται το μήνυμα “Αρχική τιμή δευτερολέπτων” ακολουθούμενο από τη start.
3. Στη συνέχεια θα υπάρχει ένας βρόχος while() ο οποίος θα δημιουργεί 100 διεργασίες (δοκιμάστε και για 5000, 10000) με τη fork() οι οποίες όλες θα εκτελούν τη nothing(). Προσοχή, ο πατέρας θα δημιουργήσει όλες τις διεργασίες και όχι η κάθε διεργασία θα δημιουργεί άλλη διεργασία.
4. Μόλις δημιουργηθούν οι 100 διεργασίες και μόνο τότε θα εκτελεί ο πατέρας 100 φορές τη waitpid() ώστε να περιμένει την επιτυχή ολοκλήρωση όλων των θυγατρικών.
5. Στη συνέχεια θα καλείται η time(), θα αποθηκεύεται ο αριθμός των δευτερολέπτων στη μεταβλητή end και αμέσως μετά θα εκτυπώνεται το μήνυμα “Τελική τιμή δευτερολέπτων” ακολουθούμενη από την end. Θα γίνεται η πράξη end-start, θα εκτυπώνεται το αποτέλεσμα ενώ θα εκτυπώνεται και το αποτέλεσμα «(end-start)/100» για να εμφανιστεί ο μέσος χρόνος δημιουργίας, εκτέλεσης και τερματισμού των 100 διεργασιών.

Πόσος είναι ο συνολικός χρόνος και ο μέσος χρόνος δημιουργίας εκτέλεσης και τερματισμού των 100 διεργασιών στο σύστημα σας (μη ξεχάσετε τη μονάδα μέτρησης);

Ερώτημα Ε [20]: Να γράψετε πρόγραμμα συγχρονισμού για τον παρακάτω γράφο προτεραιότητας κάνοντας χρήση δύο μόνο σημαφόρων. Θεωρείστε ότι κάθε διεργασία εκτελεί μία εντολή του συστήματος της αρεσκείας σας . π.χ. system("ls -l")ή system("ps -l") κ.τ.λ... Επίσης θεωρείστε ότι οι Δ1, Δ2, Δ3, Δ4 και Δ5 είναι θυγατρικές μιας μόνο διεργασίας.



Ερώτημα ΣΤ [20]: Δημιουργείστε n-διεργασίες (θυγατρικές) κάθε μία από τις οποίες να εκτελεί το παρακάτω κομμάτι ψευδο-κώδικα:

```
Char *p;
```

```
t = *p;
        CONCATENATE TO p A CHILD[i]--text;
        sleep (nSeconds[i]);
*p = t;
```

Δηλαδή, κάθε νέα (θυγατρική) διεργασία περιμένει ένα τυχαίο αριθμό δευτερολέπτων και στη συνέχεια αυξάνει το περιεχόμενο της κοινής *διαμοιραζόμενης μεταβλητή*`*p` προσαρτώντας σχετικό κείμενο. Να χρησιμοποιήσετε σημαφόρους έτσι ώστε να προστατέψετε τη κρίσιμη περιοχή κάθε διεργασίας, ώστε τελικά η μεταβλητή *p να έχει προσαρτημένα όλα τα κομμάτια κειμένων με οποιαδήποτε σειρά.

Τι αλλαγές θα πρέπει να κάνετε στον κώδικα, ώστε τα παραπάνω κομμάτια να έχουν συγκεκριμένη σειρά ώστε να βγαίνει νόημα κατά την ανάγνωση?