ΠΑΝΕΠΙΣΤΗΜΙΟ
ΔΥΤΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ

Λειτουργικά Συστήματα

Ενότητα: ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ No:05

Δρ. Μηνάς Δασυγένης

mdasyg@ieee.org

Τμήμα Μηχανικών Πληροφορικής και Τηλεπικοινωνιών

Εργαστήριο Ψηφιακών Συστημάτων και Αρχιτεκτονικής Υπολογιστών

http://arch.icte.uowm.gr/mdasyg

Άδειες Χρήσης

- Το παρόν εκπαιδευτικό υλικό υπόκειται σε άδειες χρήσης Creative Commons.
- Για εκπαιδευτικό υλικό, όπως εικόνες, που υπόκειται σε άλλου τύπου άδειας χρήσης, η άδεια χρήσης αναφέρεται ρητώς.

Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «**Ανοικτά Ψηφιακά Μαθήματα του Πανεπιστημίου Δυτικής Μακεδονίας**» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Περιεχόμενα

1. Σκοπός της άσκησης

- Διαχείριση Διεργασιών στο ΛΣ Linux.
- Κανονικές Εκφράσεις.
- Καταγραφή των εντολών με το script.
- Έλεγχος παραμέτρων γραμμής εντολών.

2. Παραδοτέα

(A) 43 ερωτήσεις
(C και G) 5+1 ασκήσεις

3. Διαχείριση Διεργασιών στο ΛΣ Linux

Όλα τα σύγχρονα λειτουργικά συστήματα μπορούν και εκτελούν ταυτόχρονα διάφορα προγράμματα, τα οποία ονομάζονται διεργασίες. Σε αυτή την εργαστηριακή άσκηση θα γνωρίσουμε τις διεργασίες στο ΛΣ Linux.

Συνδεθείτε στο ΛΣ Linux σύμφωνα με τις οδηγίες που σας έχουν δοθεί.

Βρείτε από τη σελίδα βοήθειας του ΛΣ τι κάνουν οι παρακάτω εντολές:

Εντολή	Λειτουργία	Ερώτηση
ps :		**(A1)**
kill :		**(A2)**
killall :		**(A3)**
top:		**(A4)**

Εκτός από τις ανωτέρω εντολές υπάρχουν και προγράμματα ενσωματωμένα μέσα στο φλοιό της γραμμής εντολών. Τα παρακάτω ανήκουν στο φλοιό sh *(bash)*. Χρησιμοποιείστε το `man` για να βρείτε τι κάνουν αυτές οι εντολές. Επειδή η σελίδα βοήθειας έχει πάρα πολλές πληροφορίες, μπορείτε να χρησιμοποιήσετε την αναζήτηση αφού μπείτε μέσα στη σελίδα βοήθειας με το να γράψετε `/` και το όνομα της λέξης που θέλετε να ψάξετε. Αν το επαναλάβετε θα προωθηθείτε στην επόμενη εύρεση. Για παράδειγμα αφού εκτελέσετε το `man sh` δώστε `/jobs` και επαναλάβετε μέχρι να φτάσετε στην κατάλληλη παράγραφο.

Εντολή	Λειτουργία	Ερώτηση
jobs :		**(A5)**
fg :		**(A6)**
bg :		**(A7)**

Ένα από τα βασικά προγράμματα του Linux είναι το πρόγραμμα που εμφανίζει τις τρέχουσες διεργασίες. Αυτό το πρόγραμμα είναι το ps.

Εκτελέστε το:

```
ps axuw
```

Οι παράμετροι που χρησιμοποιήσαμε στην εντολή ps τι αποτέλεσμα προκαλούν; Εξηγείστε τι κάνει η κάθε παράμετρος:

Παράμετρος	Λειτουργία	Ερώτηση
a :		**(A8)**
x :		**(A9)**
u :		**(A10)**
w :		**(A11)**

θα πάρετε ως έξοδο του προγράμματος ps αρκετές γραμμές του τύπου.

```
USER PID %CPU %MEM VSZ RSS TTY STAT START TIME COMMAND

root 3179 0.0 0.0 1624 520 tty1 Ss+ Oct12 0:00 /sbin/getty
```

Αυτά που μας ενδιαφέρουν είναι οι παρακάτω στήλες:

1η στήλη: Χρήστης που εκτελεί τη διεργασία

2η στήλη: Process ID

4η στήλη: Πόση μνήμη καταναλώνει αυτή η διεργασία

11η στήλη: Όνομα προγράμματος

Βρείτε τα παρακάτω:

Ερώτηση	Απάντηση	Αριθμός
Ποιος χρήστης έχει εκτελέσει τη διεργασία με PID 1;		**(A12)**
Ποια διεργασία έχει το PID 1;		**(A13)**
Ποια διεργασία έχει το μεγαλύτερο PID;		**(A14)**
Ποια είναι αυτή;		**(A15)**
Ποια διεργασία καταναλώνει την περισσότερη μνήμη;		**(A16)**
Τι PID έχει η διεργασία με την περισσότερη μνήμη;		**(A17)**

Υπάρχει διεργασία *(και αν ναι με τι PID)* στην οποία στην 1η στήλη εμφανίζεται μόνο ένας αριθμός και όχι ένα όνομα χρήστη; Αν υπάρχει, θα πρέπει να εξεταστεί από κάποιον διαχειριστή, γιατί σηματοδοτεί κάποια ανωμαλία.

_____ **(A18)**

Στο Linux υπάρχουν διεργασίες που εκτελούνται στο προσκήνιο και διεργασίες που βρίσκονται στο παρασκήνιο. Για να το καταλάβετε κάντε τα παρακάτω:

Εκτελέστε το πρόγραμμα **gdb**

Θα δείτε ότι έχει εκτελεστεί και περιμένει τις εντολές σας αφού έχει βγει η προτροπή **(gdb).**

Πατήστε **CTRL+Z**

Θα εμφανιστεί (αν έχετε φλοιό CSH/TCSH , δείτε και την επόμενη φωτογραφία).

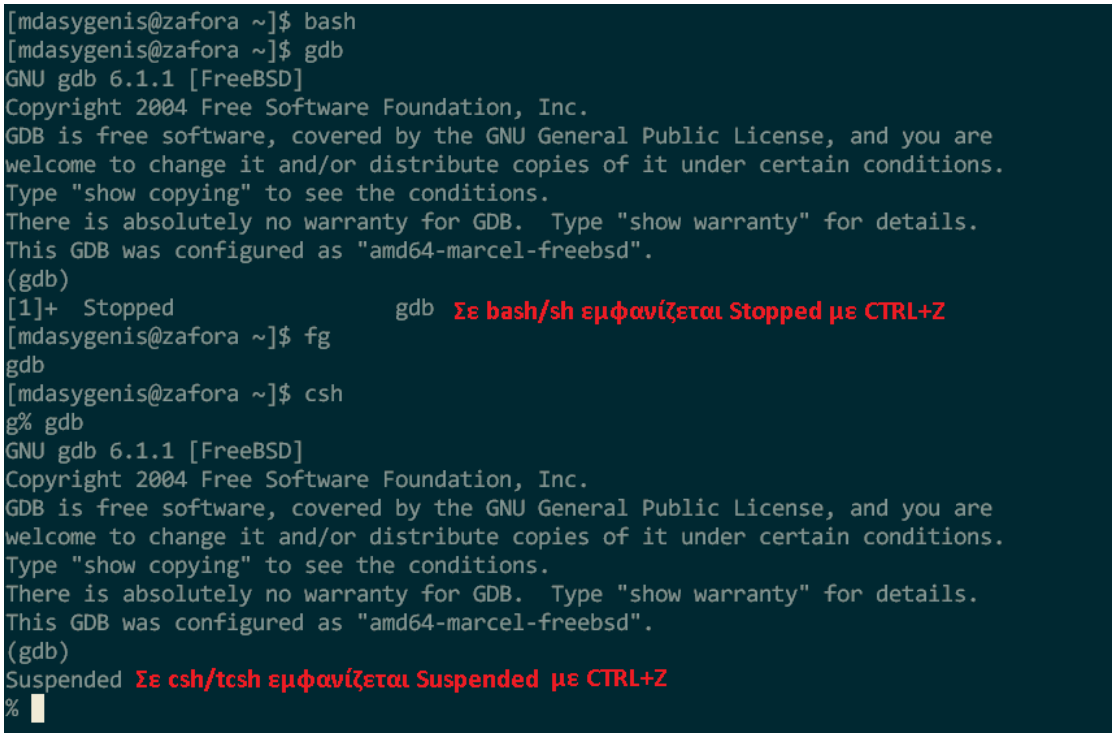
Suspended

Αν έχετε φλοιό BASH/SH θα εμφανιστεί το μήνυμα

Stopped

Μπορείτε να δείτε το φλοιό σας με

```
echo $SHELL
```



Το suspend ή stopped σημαίνει ότι η διεργασία τοποθετήθηκε στο παρασκήνιο *(background)* και δεν εκτελείται.

Σας έχει βγει η προτροπή για την επόμενη εντολή.

Δώστε την εντολή που σας εμφανίζει ΟΛΕΣ τις διεργασίες _____ **(A19)**

Αν δε βρείτε το πρόγραμμα gdb να εκτελείται τότε έχετε κάνει λάθος.

Το gdb τι PID έχει; _____ **(A20)**

Μπορείτε να δείτε τι διεργασίες έχετε εσείς στο παρασκήνιο με το να δώσετε

```
jobs
```

Αν δώσετε jobs η διεργασία gdb σε τι κατάσταση *(state)* είναι; _____ **(A21)**

Εκτελέστε την παρακάτω εντολή η οποία δημιουργεί ένα αρχείο με το όνομα myfile στον κατάλογο που βρίσκεστε με μηδενικά μεγέθους `10000000 * 512 bytes` , μόλις πατήσετε enter πατήστε CTRL+Z

```
dd if=/dev/zero of=myfile count=10000000
```

Μόλις πατήσετε CTRL+Z επιστρέφετε στη γραμμή εντολών. Βρείτε η διεργασία dd σε τι κατάσταση είναι: _____ **(A22)**

Πόσες διεργασίες είναι σε κατάσταση suspended τώρα; _____ **(A23)**

Για να επιστρέψουμε στην παρασκηνιακή διεργασία gdb και να συνεχίσουμε την εργασία μας δίνουμε:

```
fg %1
```

όπου **fg** είναι να έρθει η διεργασία στο προσκήνιο και με %1 προσδιορίζουμε την πρώτη διεργασία. Δώστε:

```
quit
```

και θα επιστρέψετε πάλι στη γραμμή εντολών.

Προκειμένου να εκτελέσουμε τη διεργασία **dd** στο παρασκήνιο θα δώσουμε

```
bg %2
```

ώστε να συνεχιστεί η εκτέλεση στο παρασκήνιο.

Αμέσως μετά δώστε

```
top
```

το οποίο μας εμφανίζει τις διεργασίες που εκτελούνται και σημειώστε πόσο χρησιμοποιείται η %CPU από τη συγκεκριμένη διεργασία: _____ **(A24)**

Τι PID έχει η συγκεκριμένη διεργασία; _____ **(A25)**

Πατήστε `q` και επιστρέψτε στη γραμμή εντολών.

Προκειμένου να τερματίσουμε τη διεργασία θα δώσουμε

```
kill απάντηση_ερώτησης_25
```

Μπορούμε να τοποθετήσουμε μια διεργασία κατευθείαν στην εκτέλεση στο παρασκήνιο με το να τοποθετήσουμε το σύμβολο & ύστερα από τη γραμμή εντολών. Για παράδειγμα:

```
dd   if=/dev/zero   of=myfile      count=10000000 &
```

Τερματίστε την εκτέλεση της παραπάνω διεργασίας με τη χρήση του προγράμματος `killall`. Εντολή που χρησιμοποιήθηκε: _____ **(A26)**

4. Κανονικές εκφράσεις (regular expressions)

Μια κανονική έκφραση είναι ένα συγκεκριμένο είδος πρότυπου κείμενου που μπορούμε να χρησιμοποιήσουμε σε πολλές γλώσσες προγραμματισμού. Με τις κανονικές εκφράσεις μπορούμε εύκολα να βρούμε κείμενο που συμφωνεί με κάποιο πρότυπο μέσα σε δεκάδες ή εκατοντάδες *(ή και πολύ περισσότερες)* γραμμές. Τις κανονικές εκφράσεις τις συναντάμε σε πολλές μορφές. Η σύνταξη που χρησιμοποιούμε στο μάθημα είναι πιο απλή απ' αυτή που χρησιμοποιείται, για παράδειγμα, στη `grep` εντολή του UNIX, ωστόσο και οι δύο συντάξεις έχουν την ίδια περιγραφική δύναμη.

Οι κανονικές εκφράσεις *(RE)* αποτελούνται από συνδυασμό κανονικών χαρακτήρων με έναν ή περισσότερους μεταχαρακτήρες. Οι μεταχαρακτήρες είναι χαρακτήρες με ειδική σημασία που καθορίζεται από το φλοιό που τους χρησιμοποιεί *(π χ Bourne Korn ή C shell)*. Οι κανονικές εκφράσεις μπορούν να χρησιμοποιηθούν σε εντολές και προγράμματα του UNIX για:

- Να προσδιορίσουν ονόματα αρχείων
- Να ανιχνεύσουν περιεχόμενα αρχείων
- Να μεταβάλλουν περιεχόμενα αρχείων

Κάθε εντολή του UNIX διαφέρει όσον αφορά τον τύπο των RE που υποστηρίζει. Ο προσδιορισμός των RE που υποστηρίζονται από κάθε εντολή είναι προτιμότερο να γίνεται ελέγχοντας την εντολή μέσω της εντολής **man** *(manual pages)*.

Μια κανονική έκφραση είναι μια σειρά από κανονικούς χαρακτήρες και ειδικούς τελεστές.

Οι κανονικοί χαρακτήρες περιλαμβάνουν το σύνολο των πεζών και κεφαλαίων γραμμάτων, τα ψηφία, και άλλους συχνά χρησιμοποιούμενους χαρακτήρες : ~, ', !, @, #, _, -, =, :, ;, ,/
Οι ειδικοί τελεστές είναι \, ., *, [, ^, $,].

Εκτός από τους παραπάνω ειδικούς τελεστές, μερικές φορές συναντάμε και τους επεκταμένους ειδικούς τελεστές **?** και **+** οι οποίοι βρίσκονται στις επεκταμένες κανονικές εκφράσεις *(extended regular expressions)*. Η σημασία των ειδικών τελεστών είναι:

Τελεστής	Σημασία
\	Ακυρώνει την ειδική σημασία του χαρακτήρα που ακολουθεί
.	Ταιριάζει οποιοδήποτε χαρακτήρα εκτός από την αλλαγή γραμμής
*	Η έκφραση που προηγείται θα ταυτιστεί μηδέν ή περισσότερες φορές
[ ]	Ταιριάζει έναν χαρακτήρα από ένα συγκεκριμένο εύρος. Π.χ. [0-9a-z]
^	Ταιριάζει την αρχή της γραμμής. Αν είναι μέσα σε [ ] και είναι ο πρώτος χαρακτήρας **[^]** σημαίνει εκτός από τους χαρακτήρες που είναι στο εύρος.
$	Ταιριάζει στο τέλος της γραμμής.
+	Ταιριάζει τον προηγούμενο χαρακτήρα μια ή περισσότερες φορές.
&	Ταιριάζει τον προηγούμενο χαρακτήρα μηδέν ή μια φορές.

Παρακαλείστε τώρα να μελετήσετε το συνοδευτικό αρχείο που έχει ανέβει σχετικά με τις κανονικές εκφράσεις, για να τις κατανοήσετε καλύτερα.

5. Το πρόγραμμα grep

Η τυπική χρήση των κανονικών εκφράσεων είναι στο **grep**. Με το grep μπορείτε:

- Να φιλτράρετε τις γραμμές που συμφωνούν με κάποιο πρότυπο.

- Να φιλτράρετε τις γραμμές που δε συμφωνούν με κάποιο πρότυπο *(με την παράμετρο* `-v` *)*

Μια χρήσιμη παράμετρος είναι η `--color` για να χρωματίζει το σημείο της γραμμής που έχει ταιριάξει το πρότυπο. Συνιστάται να τοποθετείται η κανονική έκφραση μέσα σε μονά εισαγωγικά `'` για μην προκαλείται μπέρδεμα στο φλοιό αν ερμηνευτούν διαφορετικά οι ειδικοί χαρακτήρες. Επίσης αν θέλουμε να χρησιμοποιήσουμε τις επεκταμένες κανονικές εκφράσεις τότε θα πρέπει στο grep να τοποθετήσουμε την παράμετρο `-E` .

5.1 Παραδείγματα:

Εντολή για να βρίσκει στο ls τη γραμμή που έχει ιδιοκτήτη root:

```
ls -l | grep root
```

Εντολή για να βρίσκει στο ls τη γραμμή που δεν έχει ιδιοκτήτη root:

```
ls -l | grep -v root
```

5.2 Ερωτήσεις

Ποια είναι η εντολή που θα απομακρύνει από την έξοδο τη γραμμή του `ls -l` που αναφέρει total και έναν αριθμό; _____ **(A27)**

Ποια είναι η εντολή που θα εμφανίζει μόνο τους κατάλογους από το `ls -l` _____ **(A28)**

Ποια είναι η εντολή που θα εμφανίζει μόνο τα αρχεία από το `ls -l` _____ **(A29)**

Ποια είναι η εντολή που θα εμφανίζει μόνο τους δεσμούς *(link)* από το `ls -l` _____ **(A30)**

Ποια είναι η εντολή που θα εμφανίζει μόνο τις γραμμές από το αρχείο /etc/passwd που δεν έχουν # στον πρώτο χαρακτήρα; _____**(A31)**

Ποια είναι η εντολή που θα εμφανίζει μόνο τις γραμμές από το αρχείο /etc/passwd που δεν έχουν : *(οπουδήποτε);* _____ **(A32)**

Ποια είναι η εντολή που θα εμφανίζει μόνο τις γραμμές από την εντολή `ps -axu` που δεν ανήκουν στο χρήστη root *(προσοχή, ο χρήστης είναι το πρώτο πεδίο μόνο)* _____ **(A33)**

Ποια είναι η εντολή που θα εμφανίζει μόνο τις γραμμές από την εντολή `ps -axu` που έχουν *(τουλάχιστον)* κατάσταση I *(idle)*.

Η κατάσταση βρίσκεται στη στήλη STAT._____**(Α34)**

Ποια είναι η εντολή που θα εμφανίζει μόνο τις γραμμές από την εντολή `ps -axu` που ΔΕΝ έχουν κατάσταση I *(idle)*.

Η κατάσταση βρίσκεται στη στήλη STAT._____**(Α35)**

Μπορούμε αν θέλουμε να τοποθετούμε διαδοχικά φίλτρα grep αν θέλουμε κάτι πολύπλοκο. Για παράδειγμα, αν θέλουμε τις γραμμές `του ps -axu` *(είναι ίδιο με το* `ps axuw`*)* που είναι του χρήστη mysql και έχουν ημερομηνία έναρξης **Mar** τότε μπορούμε να κάνουμε :

```
ps -axuw | grep '^mysql' | grep ' Mar '
```

Αν μας ενοχλούν τα πολλά κενά γιατί μπερδεύουν την cut , μπορούμε να τα συμπιέσουμε σε ένα. Δηλαδή, όταν είναι 2 και περισσότερα να γίνονται ένα με το tr και την παράμετρο -s .

Παράδειγμα: `ps axuw | tr -s " " | cut -f 3`

Το grep επιστρέφει τιμή επιτυχίας αν βρεθεί το συγκεκριμένο πρότυπο ή αποτυχίας αν δε βρεθεί. Δηλαδή, μπορούμε να το χρησιμοποιήσουμε άμεσα μέσα σε μια `if[...]` με τα ανάποδα μονά εισαγωγικά *(όλα τα προγράμματα του linux μπορούμε να τα χρησιμοποιούμε μέσα σε if)* ως εξής:

```
if [ ` grep root /etc/passwd ` ] ; then
     echo "Root word located at /etc/passwd"
else
     echo "Root word does not exist in /etc/passwd"
fi
```

Με την ίδια λογική μπορούμε να κάνουμε έλεγχο παραμέτρων αν ταιριάζουν σε κάποιο πρότυπο. Για παράδειγμα αν θέλουμε η πρώτη παράμετρος να είναι αριθμός θα πρέπει να γίνει έλεγχος για

```
if [ `echo $1 | grep '^[0-9][0-9]*$' ` ] ; then
```

5.3 Ασκήσεις (C1), (C2) και (C3)

(C1) Να δημιουργήστε σενάριο φλοιού sh με το όνομα **c1.sh** το οποίο θα δέχεται μια και μόνο μια παράμετρο *(αν δοθούν περισσότερες ή μια τότε να σταματάει η εκτέλεση ύστερα από σχετικό μήνυμα)*. Η παράμετρος αυτή θα είναι μια λέξη με μικρούς λατινικούς χαρακτήρες *(αν δεν είναι να υπάρχει σχετικό μήνυμα και να σταματάει το script)*. Το σενάριο στη συνέχεια θα ψάχνει αυτή τη λέξη μέσα στο αρχείο /etc/passwd Αν υπάρχει τότε θα εμφανίζεται σχετικό μήνυμα διαφορετικά θα εμφανίζεται ότι δε βρέθηκε η λέξη.

(C2) Να δημιουργήστε σενάριο φλοιού sh με το όνομα **c2.sh** το οποίο θα δέχεται μια και μόνο μια παράμετρο *(αν δοθούν περισσότερες ή μια τότε να σταματάει η εκτέλεση ύστερα από σχετικό μήνυμα)*. Η παράμετρος αυτή θα είναι μια λέξη με μικρούς λατινικούς χαρακτήρες. Αν δεν είναι λέξη με μικρούς χαρακτήρες τότε θα ερωτάται ο χρήστης να δώσει μια λέξη με μικρούς λατινικούς. Η διαδικασία θα επαναλαμβάνεται μέχρι ο χρήστης να δώσει μια λέξη με μικρούς λατινικούς και τότε θα συνεχίζει παρακάτω. Το σενάριο στη συνέχεια θα ψάχνει αυτή τη λέξη μέσα στο αρχείο /etc/passwd Αν υπάρχει τότε θα εμφανίζεται σχετικό μήνυμα διαφορετικά θα εμφανίζεται ότι δε βρέθηκε η λέξη.

Ποια είναι η εντολή που θα εμφανίζει μόνο τις γραμμές που ξεκινάνε από **#** *(δίεση)* στο αρχείο /etc/passwd _____ **(A36)**

Ποια είναι η εντολή που θα εμφανίζει μόνο τις γραμμές που ξεκινάνε από τέσσερις μικρούς λατινικούς χαρακτήρες (a-z) και μετά υπάρχει άνω κάτω τελεία; _____ **(A37)**

Υπενθυμίζεται ότι μπορούμε να χρησιμοποιήσουμε σε φίλτρο το πρόγραμμα wc *(word count)* για καταμέτρηση των γραμμών με την παράμετρο **-l** *(παύλα ελ)*.

Ποια είναι η εντολή που καταμετράει τις γραμμές από την εντολή **ps -axu** που ΔΕΝ έχουν κατάσταση I *(idle)*.

Η κατάσταση βρίσκεται στη στήλη STAT._____**(A38)**

Ποια είναι η εντολή που καταμετράει τις γραμμές που ξεκινάνε από τέσσερις μικρούς λατινικούς χαρακτήρες (a-z) και μετά υπάρχει άνω κάτω τελεία;_____ **(A39)**

Υπενθυμίζεται ότι μπορείτε να χρησιμοποιήσετε το πρόγραμμα find για να βρείτε κάποια αρχεία που ταιριάζουν με κάποια πρότυπα του φλοιού *(όχι κανονικές εκφράσεις)*. Για παράδειγμα η εντολή **find / -iname "*.sh"** βρίσκει όλα τα αρχεία από τη ρίζα που έχουν οποιοδήποτε όνομα και κατάληξη sh. Το * *(αστέρι)* δεν είναι κανονική έκφραση γιατί το find δεν καταλαβαίνει κανονικές εκφράσεις, αλλά εκφράσεις φλοιού. Επίσης μπορούμε να δώσουμε την εντολή **-print** για να εκτυπώσει στην οθόνη τη λίστα των αρχείων που ταιριάζουν στο πρότυπο.

(C3) Να δημιουργήσετε σενάριο φλοιού **c3-updatedb.sh** το οποίο χρησιμοποιώντας το find δημιουργεί ένα αρχείο file.db στον κατάλογο ~ το οποίο έχει όλα τα αρχεία από τη διαδρομή /home . Στη συνέχεια να δημιουργήσετε ένα αρχείο **c3-locate.sh** το οποίο θα δέχεται μια και μόνο μια παράμετρο *(αν δοθούν περισσότερες ή μια τότε να σταματάει η εκτέλεση ύστερα από σχετικό μήνυμα)*. Η παράμετρος αυτή θα είναι μια λέξη με λατινικούς χαρακτήρες, αριθμούς και τελεία *(αν δεν είναι να υπάρχει σχετικό μήνυμα και να σταματάει το script)*. Το σενάριο **c3-locate.sh** στη συνέχεια θα ψάχνει αυτή τη λέξη μέσα στο αρχείο ~file.db Αν

υπάρχει τότε θα εμφανίζεται η συγκεκριμένη γραμμή διαφορετικά θα εμφανίζεται το μήνυμα “File Not Found”.

Ένα βοηθητικό πρόγραμμα για μεταφόρτωση σελίδων από το Internet είναι το wget. Το wget μπορεί να κατεβάσει και να αποθηκεύσει στο δίσκο μια σελίδα. Μπορεί να χρησιμοποιηθεί η παράμετρος `-O` για να προσδιοριστεί το αρχείο που θα αποθηκευτεί. Για παράδειγμα `wget http://www.in.gr -O /tmp/test` θα κατεβάσει την πρώτη σελίδα από το ww w.in.gr και θα την αποθηκεύσει στο αρχείο /tmp/test.

Ποιες είναι οι εντολές που θα καταμετράνε τις γραμμές που έχουν τη λέξη META ή meta από τη σελίδα ww w.in.gr _____**(A40)**

Όταν δημιουργούμε ένα script πολλές φορές θέλουμε να χρησιμοποιήσουμε προσωρινά αρχεία. Για λόγους ασφαλείας δε μπορούμε να έχουμε πάντα το ίδιο προσωρινό αρχείο, αφού αν εκτελεστεί πολλές φορές ταυτόχρονα το script μας θα δημιουργηθεί πρόβλημα. Η λύση δίνεται με το πρόγραμμα mktemp το οποίο δημιουργεί ένα προσωρινό αρχείο σύμφωνα με τις παραμέτρους που δίνουμε. Συνήθως λοιπόν σε ένα script που χρειαζόμαστε προσωρινά αρχεία, στην αρχή του σεναρίου δίνουμε `tmpfile1=`mktemp /tmp/$0.XXXX`` για κάθε προσωρινό αρχείο που θέλουμε να χρησιμοποιήσουμε. Στη συνέχεια όταν θέλουμε να χρησιμοποιήσουμε το προσωρινό αρχείο χρησιμοποιούμε το $tmpfile1 (για παράδειγμα `ls -la > $tmpfile1`).

Στο τέλος του σεναρίου, όταν δε χρειαζόμαστε το προσωρινό αρχείο το διαγράφουμε με

```
rm $tmpfile1 .
```

Το πρόγραμμα diff συγκρίνει δύο αρχεία και βρίσκει τις διαφορές.

Σύνταξη:

```
diff file1 file2
```

5.4 Ασκήσεις C4 και C5

(C4) Να δημιουργήσετε σενάριο φλοιού `c4.sh` το οποίο θα ελέγχει για αλλαγές μια συγκεκριμένη σελίδα που θα έχει τοποθετηθεί στη μεταβλητή url. Για παράδειγμα:

`url="http:/ /feeds.feedburner.com/grinsomnia"` Το script θα επισκέπτεται τη σελίδα και χρησιμοποιώντας το πρόγραμμα **`wget`** θα την αποθηκεύει σε ένα προσωρινό αρχείο *(tmpfile1).* Στη συνέχεια θα εκτελεί την εντολή sleep με όρισμα ένα χρονικό διάστημα *(π.χ. 60 sec, δηλαδή* `sleep 60`*)* για να μπει σε κατάσταση αδράνειας για αυτό το χρονικό διάστημα. Ύστερα από αυτό το χρονικό διάστημα θα

επισκέπτεται πάλι τη σελίδα και θα αποθηκεύει σε ένα άλλο προσωρινό αρχείο *(tmpfile2)* τη σελίδα και χρησιμοποιώντας το diff θα κάνει σύγκριση των δύο αρχείων (μπορείτε να χρησιμοποιήσετε τη δομή `if [ `diff $tmpfile1 $tmpfile2`];`) και αν είναι διαφορετικά τότε να εμφανίζει το μήνυμα "Είναι διαφορετικά" και προαιρετικά τις αλλαγές, ενώ ταυτόχρονα θα αντιγράφει το **`$tmpfile2`** στο **`$tmpfile1`** για την επόμενη επανάληψη. Αν είναι ίδια δε θα εμφανίζει τίποτα. Θα επαναλαμβάνεται η διαδικασία από το sleep έως αυτό το σημείο. Το script θα τερματίζει μόνο αν πατήσει ο χρήστης CTRL+C.

Όταν εκτελείται η εντολή sleep σε ποια κατάσταση μεταβαίνει η διεργασία του script από τις 7 καταστάσεις που παρουσιάστηκαν στη θεωρία;_____**(A41)**

Σε μια εκτέλεση του b4 script πόσες κλήσεις συστήματος πιστεύετε κατ' ελάχιστον ότι εκτελούνται; *)*_____ **(A42)**
(κλήσεις συστήματος και όχι προγράμματα. Π.χ. το να αναφέρει κάποιος το wget είναι λάθος

(C5) Να δημιουργήσετε σενάριο φλοιού **`c5.sh`** το οποίο θα μετράει σε μια ιστοσελίδα τους αριθμούς τηλεφώνων της Κοζάνης. Συγκεκριμένα, θα επισκέπτεται τη σελίδα http:/ /www.icte.uowm.gr, θα αποθηκεύει τα δεδομένα σε ένα προσωρινό αρχείο και με το grep θα βρίσκει τις γραμμές που έχουν το τηλεφωνικό πρόθεμα της Κοζάνης *(24610)* και θα μετράει πόσες γραμμές έχουν τηλέφωνο.
ΓΙΑ ΠΡΟΧΩΡΗΜΕΝΟΥΣ: Χρησιμοποιώντας όποιο πρόγραμμα βρίσκεται στη βασική εγκατάσταση του linux δοκιμάστε να εκτυπώνεται αμέσως μετά η λίστα με τα τηλέφωνα και μόνο αυτά *(π.χ. cut ή sed).*

6. Το πρόγραμμα script

Πολλές φορές θέλουμε να καταγράφουμε κάθε εντολή που δίνουμε στο τερματικό καθώς και την έξοδο του προγράμματος μας. Το πρόγραμμα που χρησιμοποιούμε είναι το script. Ξεκινάμε το πρόγραμμα γράφοντας **`script όνομα_αρχείου`**.

Ποια παράμετρος στο script ενεργοποιεί την καταγραφή χρόνου;_____ **(A43)**

{Αν χρησιμοποιείτε FreeBSD δεν υπάρχει αυτή η παράμετρος καταγραφής χρόνου}.

Η καταγραφή του χρόνου, σύμφωνα με το εγχειρίδιο χρήσης γίνεται στο κανάλι εξόδου τυπικού σφάλματος *(Output timing data to standard error)*. Προκειμένου να ανακατευθύνουμε το κανάλι standard error σε κάποιο αρχείο με το όνομα timingfile θα χρησιμοποιήσουμε την ανακατεύθυνση εξόδου stdout *(δηλαδή το κανάλι εξόδου 2)* ως εξής:

```
script -t recordfile 2>timingfile
```

Μόλις δώσετε αυτή την εντολή θα δείτε:

`Script started, file is recordfile` (ή κάποιο παρόμοιο μήνυμα, αναλόγως της έκδοσης του προγράμματος script), δηλαδή ότι έχει αρχίσει η καταγραφή. Ότι γράφετε από εδώ και πέρα καθώς και η έξοδος της οθόνης μεταφέρεται στο αρχείο recordfile ενώ οι πληροφορίες χρονισμού *(πότε δώσατε την κάθε εντολή και πότε εμφανίστηκε η κάθε έξοδος)* στο αρχείο timingfile . Μόλις θέλετε να σταματήσετε την καταγραφή πατήστε exit και θα σταματήσει. Θα παρατηρήσετε ότι στον κατάλογο θα υπάρχουν τα 2 αρχεία *(recordfile , timingfile)* . Αν έχετε εγκατεστημένο το πρόγραμμα scriptreplay μπορείτε να δώσετε το **`scriptreplay timingfile recordfile`** και θα δείτε να αναπαράγεται στην οθόνη ότι εντολή δώσατε, στο χρόνο που τη δώσατε και την έξοδο που είχε.

(G1) Χρησιμοποιώντας το script να καταγράψετε σε ένα αρχείο *(μαζί με το timing)* μια ορθή χρήση για κάθε μια από τις παρακάτω εντολές *(δηλαδή να μην εμφανίζεται σφάλμα κατά την εκτέλεση)*:

`mkdir,rmdir,rm,mv,cp,touch,cd,ps,kill,bg,fg,ls,grep,dd,diff,wget,find`

και μια δύο χρήσεις κάποιου επεξεργαστή τερματικού *(π.χ. ανοίξτε ένα αρχείο, διαγράψτε την πρώτη σειρά και αποθηκεύστε το αρχείο)*. Θα στείλετε και τα δύο αρχεία που δημιουργήθηκαν από το script.