Μερικές άλλες χρήσιμες εντολές

whoami (Εμφανίζει το όνομα του τρέχοντος χρήστη)

pwd (Εμφανίζει το όνομα του τρέχοντος καταλόγου)

less (Το ίδιο με την εντολή «**cat**», αλλά επιτρέπει την κύλιση στο κείμενο.

more (Το ίδιο με την εντολή «less»)

echo 'Welcome' (Εμφανίζει στην οθόνη ό,τι πληκτρολογήσουμε μέσα σε εισαγωγικά)

echo 'OK !!!' > test.txt (Εγγράφει στο αρχείο κειμένου «test.txt» το κείμενο «OK !!!». Εάν το αρχείο «test.txt» δεν υπάρχει τότε δημιουργείται.

echo $(whoami) > test.txt (Εγγράφει στο αρχείο κειμένου «test.txt» ό,τι προκύπτει από την εκτέλεση της εντολής «**whoami**».

echo 'END' >> test.txt (Προσαρτεί στο τέλος του αρχείου «test.txt» μια σειρά με την λέξη «END».

finger (Δείχνει μια λίστα με τους συνδεδεμένους χρήστες)

talk «**username άλλου χρήστη**» (Επιτρέπει την συνομιλία μεταξύ δύο χρηστών)

ps ax (Δείχνει μια λίστα με όλα τα φορτωμένα processes και τα PID)

kill (Μπορείτε να σταματήσετε το process ή το πρόγραμμα με process ID π.χ. 122 πληκτρολογώντας «**kill 122**».

man «command» (Δείχνει τον τρόπο χρήσης της «**command**»). π.χ.
man cp (δείχνει τον τρόπο χρήσης της εντολής «**cp**»)
«command» --help (Βοήθεια για τον τρόπο χρήσης της «**command**»). π.χ.
cp --help (Βοήθεια για τον τρόπο χρήσης της «**cp**»)

Αναζήτηση-εντοπισμός αρχείων με την εντολή «find».

find /home -user pc17 (Εντοπισμός όλων των αρχείων του καταλόγου «home», τα οποία έχουν κτήτορα (ιδιοκτήτη) τον χρήστη «**pc17**».)

find /usr -name *stat (Εντοπισμός όλων των αρχείων του καταλόγου «usr», των οποίων το όνομα λήγει σε «stat».)

find . -perm 755 (Εντοπισμός όλων των αρχείων του τρέχοντος καταλόγου «.», στα οποία ο κτήτωρ έχει δικαιώματα `rwx` , ενώ η ομάδα και οι άλλοι έχουν δικαιώματα `r-x`.)

read –p "Please give an answer: " ans (Ό,τι πληκτρολογηθεί στο prompt που εμφανίζεται γίνεται τιμή της μεταβλητής ans, π.χ. «Yes»)

```
echo $ans
Yes
```

Αναζήτηση-εντοπισμός κειμένου μέσα σε αρχεία με την εντολή «grep»

Η εντολή «`grep`» αναζητεί σε δοσμένο αρχείο γραμμές που περιέχουν συγκεκριμένη σειρά χαρακτήρων ή λέξεις.

grep 'ακολουθία χαρακτήρων' «όνομα αρχείου»

Εάν κατά την αναζήτηση δεν θέλουμε να διακρίνονται πεζά από κεφαλαία τότε χρησιμοποιούμε το flag «-i» (insensitive).

grep -i 'ακολουθία χαρακτήρων' «όνομα αρχείου»

Εάν η προς εντοπισμό ακολουθία χαρακτήρων αποτελεί ολόκληρη λέξη τότε συμπληρώνουμε το flag με «**w**»:

grep -iw 'λέξη' «όνομα αρχείου»

Εάν επιδιώκουμε αναζήτηση μιας ακολουθίας χαρακτήρων σε όλα τα αρχεία όλων των υποκαταλόγων του τρέχοντος καταλόγου τότε χρησιμοποιούμε το flag «**-r**»:

grep -r 'ακολουθία χαρακτήρων' *

Η εντολή **pgrep** επιτρέπει τον εντοπισμό ενός process ή προγράμματος που είναι ενεργό εκείνη την στιγμή χρησιμοποιώντας το όνομά του και επιστρέφει το Process ID (PID), π.χ.:

pgrep xterm

Συνδυασμός των εντολών «find» και «grep»

Εάν θέλουμε η εντολή «**grep**» να αναζητήσει μια ακολουθία χαρακτήρων μέσα σε αρχεία που εντοπίζουμε με την εντολή «**find**», τότε συνδυάζουμε τις δύο εντολές:

Οι δύο εντολές που ακολουθούν αναζητούν την ακολουθία χαρακτήρων «**done >**» σε αρχεία του τρέχοντος καταλόγου και των υποκαταλόγων του.

```
find . -type f -exec grep 'done >' {} \;
```

```
find . -type f | xargs grep 'done >'
```

Πολύ χρήσιμες είναι οι εντολές **sed** και **awk** που όμως απαιτούν μεγαλύτερο βαθμό εξοικείωσης. Πιο συγκεκριμένα με την εντολή sed μπορούμε χειρισθούμε κείμενα με την βοήθεια regular expressions. Η εντολή awk είναι πιο ισχυρή και αποτελεί γλώσσα αναζήτησης και επεξεργασίας μοτίβων.

sed (**s**tream **ed**itor), s/ (substitute), /g (global)

Πολιτική δικαιωμάτων (permissions)

Κάθε αρχείο ή κατάλογος στο Linux συνδέεται με δικαιώματα πρόσβασης (τί επιτρέπεται να γίνει με το αρχείο), τα οποία είναι τριών τύπων:

- Ανάγνωσης (**r**)
- Τροποποίησης (**w**)
- Εκτέλεσης (**x**)

Τα δικαιώματα αυτά ορίζονται για τρεις τύπους χρηστών:

- Τον κτήτορα (owner) του αρχείου
- Την ομάδα (group) που ανήκει ο κτήτωρ
- Τους άλλους χρήστες (other users)

Έτσι προκύπτουν 9 bits πληροφορίας καθένα από τα οποία μπορεί να πάρει δύο τιμές: επιτρέπεται ή απαγορεύεται.

Αυτόματα «κτήτωρ» ενός αρχείου γίνεται ο χρήστης που το δημιουργεί και έχει πλήρη δικαιώματα (**r**ead, **w**rite, e**x**ecute). Ο υπερχρήστης root έχει όλα τα δικαιώματα. Σε κάθε αρχείο ή κατάλογο στο Linux μπορεί να αποδοθεί ένας κτήτωρ (owner). Με κάθε αρχείο ή κατάλογο συνδέεται μια πολιτική δικαιωμάτων. Δηλαδή, ποιοι χρήστες ή ομάδες χρηστών θα έχουν δικαιώματα ανάγνωσης (r), τροποποίησης (w) και εκτέλεσης (x). Η τροποποίηση περιλαμβάνει και το σβήσιμο ή την μετακίνηση. Ο κτήτωρ μπορεί να αλλάξει τις ιδιότητες ενός αρχείου του π.χ. να το μετατρέψει σε αναγνώσιμο, αλλά μη εγγράψιμο και μη εκτελέσιμο. Ο υπερχρήστης μπορεί να δώσει σε κάποιον χρήστη δικαιώματα πρόσβασης π.χ.

στην ανάγνωση και στην εκτέλεση ενός προγράμματος αλλά να του στερήσει το δικαίωμα τροποποίησης.

chmod (Αλλάζει τα δικαιώματα ενός αρχείου [r, w, x])

chown (Αλλάζει τον κτήτορα ενός αρχείου)

Δικαιώματα σε μορφή κειμένου

Αποτελούνται από 10 χαρακτήρες. Ο πρώτος χαρακτήρα φανερώνει τον τύπο του αρχείου, δηλαδή κατάλογος (d), σύνδεσμος (l) ή συνηθισμένο αρχείο (-). Οι επόμενοι 9 χαρακτήρες αντιστοιχούν σε δικαιώματα, χωρισμένα σε τρεις ομάδες: κτήτωρ (owner), ομάδα (group), άλλοι (others). Κάθε ομάδα αποτελείται από τρία σύμβολα: **rwx** (με αυτήν την σειρά) . Εάν κάποιο δικαίωμα δεν παρέχεται, τότε στην θέση του αντίστοιχου χαρακτήρα μπαίνει μια παύλα «-». Π.χ.

-rwxr--r--

0123456789

- Το σύμβολο στην θέση 0 (-) δείχνει ότι πρόκειται για συνηθισμένο αρχείο.
- Τα σύμβολα στις θέσεις 1 έως 3 (rwx) αποτελούν δικαιώματα του κτήτορα του αρχείου. Δηλαδή έχει πρόσβαση ανάγνωσης, εγγραφής-τροποποίησης και εκτέλεσης.
- Τα σύμβολα στις θέσεις 4 έως 6 (r--) αναφέρονται στα αντίστοιχα δικαιώματα της ομάδας στην οποία ανήκει ο κτήτωρ. Εδώ η ομάδα έχει μόνο δικαίωμα ανάγνωσης.
- Τα σύμβολα στις θέσεις 7 έως 9 (r--) αναφέρονται στα αντίστοιχα δικαιώματα των «άλλων».

dr-x------

Εδώ πρόκειται για κατάλογο (d) στον οποίον ο κτήτωρ έχει δικαιώματα ανάγνωσης και εκτέλεσης, η ομάδα και οι άλλοι δεν έχουν κανένα δικαίωμα.

Δικαιώματα σε αριθμητική (οκταδική) μορφή

Παράδειγμα παράστασης αριθμού στο δεκαδικό σύστημα:

$$92_{10} = 9x10^1 + 2x10^0$$

Παράδειγμα παράστασης του ίδιου αριθμού οκταδικό σύστημα:

$$134_8 = 1x8^2 + 3x8^1 + 4x8^0$$

Τα δικαιώματα μπορούν να παρασταθούν και σε αριθμητική μορφή (π.χ. κατά την χρήση της εντολής «chmod») με 4 ψηφία από το 0 έως το 7. Το πρώτο ψηφίο «0» σημαίνει ότι ακολουθεί οκταδική παράσταση (συνήθως παραλείπεται). Το δεύτερο

ψηφίο αναφέρεται στα δικαιώματα του κτήτορα, το τρίτο στα δικαιώματα της ομάδας και το τέταρτο στα δικαιώματα των άλλων.

Οκταδικό ψηφίο	Μορφή κειμένου	Δυαδική τιμή	Σημασία
0	`---`	000	Απαγόρευση κάθε πρόσβασης
1	`--x`	001	Επιτρέπεται μόνο εκτέλεση
2	`-w-`	010	Επιτρέπεται μόνο εγγραφή
3	`-wx`	011	Επιτρέπονται μόνο εγγραφή και εκτέλεση
4	`r--`	100	Επιτρέπεται μόνο ανάγνωση
5	`r-x`	101	Επιτρέπονται μόνο ανάγνωση και εκτέλεση
6	`rw-`	110	Επιτρέπονται μόνο ανάγνωση και εγγραφή
7	`rwx`	111	Επιτρέπονται όλα

Παραδείγματα:

644	*κτήτωρ*: δικαιώματα ανάγνωσης και εγγραφής *ομάδα* : μόνο δικαίωμα ανάγνωσης *άλλοι* : μόνο δικαίωμα ανάγνωσης.
755	*κτήτωρ*: δικαιώματα ανάγνωσης, εγγραφής και εκτέλεσης *ομάδα* : δικαιώματα ανάγνωσης και εκτέλεσης *άλλοι* : δικαιώματα ανάγνωσης και εκτέλεσης

Παραδείγματα χρήσης chmod, chown

Έστω ότι δημιουργήσατε το script «`dokimi.sh`» και θέλετε να το μετατρέψετε σε εκτελέσιμο. Πληκτρολογήστε**:**

```
chmod 755 dokimi.sh
```

Επίσης με:

```
chmod +x dokimi.sh
```

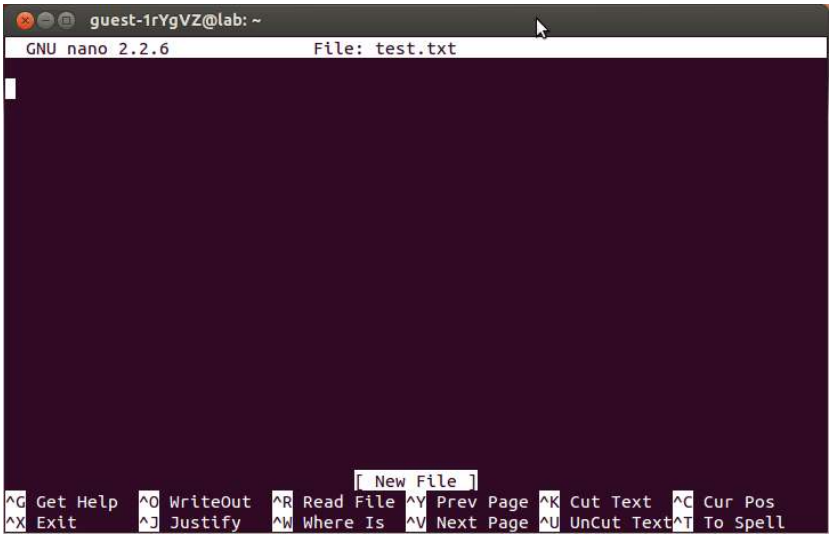
Προκειμένου να αλλάξετε τον κτήτορα του αρχείου «**`file`**» σε «**`ego`**» πρέπει να έχετε δικαιώματα root (**`chown ego file)`.**

Ο απλός κειμενογράφος «nano»

Για την σύνταξη κειμένων μπορούμε να χρησιμοποιήσουμε τον κειμενογράφο «nano». Έστω ότι θέλετε να δημιουργήσετε το αρχείο κειμένου `test.txt` και να συντάξετε σ' αυτό ένα κείμενο. Πληκτρολογήστε:

```
nano test.txt
```

Το τερματικό μετατρέπεται τότε σε οθόνη επεξεργασίας κειμένου. Πληκτρολογήστε δοκιμαστικά δύο σειρές κειμένου.



Στο κάτω μέρος της οθόνης υπάρχουν συνδυασμοί πλήκτρων με την λειτουργία τους. Είναι χρήσιμο να θυμάστε τους εξής συνδυασμούς πλήκτρων:

`ALT + A` (επιλογή κειμένου με τα βέλη)

`ALT + 6` (αντιγραφή επιλογής στο πρόχειρο (clipboard)

`CTRL + U` (επικόλληση επιλογής) `(^U)`

Δοκιμάστε αντιγραφή και επικόλληση τμημάτων του κειμένου που συντάξατε. Δώστε εντολή για έξοδο από τον κειμενογράφο με «`^X`» και πληκτρολογήστε «`y`» για να σώσετε το κείμενο. Πατήστε μετά «`Enter`» για να σώσετε με το ίδιο όνομα αρχείου με αυτό που ξεκινήσατε.

Χρήση των εντολών του bash για δημιουργία script

Είναι δυνατόν να ομαδοποιήσουμε την εκτέλεση μιας σειράς εντολών του shell συντάσσοντας ένα script. Π.χ. γράφουμε με έναν editor τις ακόλουθες εντολές

```
#!/bin/bash
```

```
ls
mkdir test1
echo 'OK !' > test1/test1.txt
cat test1/test1.txt
```

και σώζουμε το script με το όνομα «script.sh». Αφού το μετατρέψουμε σε εκτελέσιμο (**chmod 755 script.sh**) το εκτελούμε με:

```
./script.sh
```

Οι εντολές μπορούν να γραφούν σε μια γραμμή, αρκεί να διαχωρίζονται μεταξύ τους με «;».

```
#!/bin/bash
cd ..; ls; mkdir test1; echo 'OK !' > test1/test1.txt; cat test1/test1.txt
```

Ένα script μπορεί να συμπεριλάβει διαδοχικές εντολές που συνθέτουν ένα πρόγραμμα. Τα πιο συνηθισμένα στοιχεία ενός τέτοιου προγράμματος είναι το «for-do loop» και η εντολή «if». Το «for-do loop» χρησιμοποιείται για επαναλαμβανόμενη εκτέλεση μιας ή περισσοτέρων εντολών.

```
#!/bin/bash
for i in {1..5}
do
  echo 'OK !'
done > test.txt
```

Το **for i in {1..5}** ισοδυναμεί με **for i in 1 2 3 4 5**

Δοκιμάστε να βάλετε μετά το **"OK !"** την τιμή της **"i"**.

```
#!/bin/bash
while read line
do
  name=$line
  echo "Text from file: $name"
done < test.txt
```

Η εντολή «**if**» χρησιμοποιείται για να ελεγχθεί εάν αληθεύει μια συνθήκη. Εάν αληθεύει η συνθήκη τότε εκτελούνται κάποιες εντολές. Στην αντίθετη περίπτωση εκτελούνται κάποιες άλλες ή και καμία.

```
#!/bin/bash
count=100
if [ $count -eq 100 ]
then
```

```
  echo 'count is 100'
else
  echo 'count is not 100'
fi
```

```
#!/bin/bash
if [ '$(whoami)' != 'root' ]; then
    echo 'No permission to reboot'
fi
```

Το κέλυφος bash προσφέρει μεγάλη ποικιλία λογικών τελεστών, τόσο για αριθμητικές συγκρίσεις όπως **-eq** (ίσο), **-gt** (μεγαλύτερο από), **-lt** (μικρότερο από), όσο και για συγκρίσεις ακολουθιών χαρακτήρων, όπως **=** (ίδιο), **!=** (διάφορο) και πολλοί άλλοι.

```
#!/bin/bash
cd Desktop/PDBs
for pdb in 1A8U.pdb1 1ADW.pdb1 1AI9.pdb1
do
   sed 's/HIS/HSE/g' $pdb > H_$pdb.pdb
done
```

Το **s** στην εντολή **sed** σημαίνει να αντικατασταθεί (substitute) ό,τι υπάρχει μεταξύ του πρώτο και δευτέρου **/** με αυτό που υπάρχει μεταξύ του δευτέρου και του τρίτου **/**. Το **g** στο τέλος σημαίνει παντού (global).

Αριθμητικές πράξεις με Shell Scripts

Οι μεταβλητές σε ένα shell script αντιμετωπίζονται εξ ορισμού ως χαρακτήρες και **όχι ως αριθμοί**. Αυτό εισάγει μια δυσκολία στο να εκτελούμε αριθμητικές πράξεις. Οι γλώσσες προγραμματισμού με scripts όπως είναι η Tcl, η Perl ή η Python είναι καταλληλότερες όταν χρειάζεται μαθηματικός προγραμματισμός. Εντούτοις, είναι δυνατόν να κάνουμε μαθηματικές πράξεις με shell scripts.

Η εντολή declare

Με την εντολή αυτή μπορούμε π.χ. να δηλώσουμε ότι ο τύπος μιας μεταβλητής είναι ακέραιος αριθμός. Το παράδειγμα που ακολουθεί δείχνει σε τί εξυπηρετεί η εντολή declare:

```
$ n=6/3
$ echo $n
6/3
$ declare -i n
# Το -i δηλώνει ότι η μεταβλητή που ακολουθεί είναι ακέραιος (integer)
# αριθμός.
$ echo $n
2
```

Η εντολή expr

ΠΡΟΣΟΧΗ στα διάκενα !

```
$ z=5
$ z=$(expr $z+1)
$ echo $z
5+1
#  Χρειάζονται κενά γύρω από το "+" για γίνει η πράξη.
$ z=$(expr $z + 1)
$ echo $z
6
```

Η εντολή let

Πιο ειδική για αριθμητικές πράξεις είναι η εντολή **let**. Εδώ, σε αντίθεση με την **expr,** πρέπει να αποφεύγονται τα διάκενα. Έχει ενδιαφέρον ότι η **let** δείχνει ανοχή στην παράληψη του "$" μπροστά από τις μεταβλητές.

```
$ let z=5
$ echo $z
5
$ let z=$z+1
$ echo $z
6

# Τα διάκενα γύρω από το "+" δεν δουλεύουν με την let.
$ let z=$z + 1
-bash: let: +: syntax error: operand expected (error token is
"+")

# Ενώ:
$let z=z+1
$echo $z
7
```

Εναλλακτικά, αντί να χρησιμοποιήσουμε την **let** μπορούμε να βάλουμε την μαθηματική έκφραση μέσα σε διπλή παρένθεση.

```
# Δουλεύει χωρίς διάκενα
$ ((e=5))
$ echo $e
5

# Δουλεύει και με διάκενα
$ (( e = e + 3 ))
$ echo $e
8

$ (( e=e+4 ))
$ echo $e
12
```

Η εντολή bc

ΕΠΙΤΡΕΠΕΙ σε συνδυασμό με την **scale** υπολογισμούς floating point.

```
A=12 ; bc <<< "scale=2; $A/5"
2.40
```