Εργαστήριο 4: Προγραμματισμός Κελύφους Bash

Στο εργαστήριο θα μελετηθούν:

- *Εργαστηριακή Άσκηση 2*
- *Παραδείγματα Προγραμματισμού Κελύφους*
  - *Υπολογισμός του παραγοντικού ενός ακέραιου αριθμού*
  - *I/O redirection*

Διδάσκων: Γιώργος Χατζηπολλάς

Εργαστηριακή Άσκηση 2
Γενικά

- **Θέμα:** η υλοποίηση ενός προγράμματος ανάλυσης ιστοσελίδων HTML (Hypertext Markup Language) ενός δομημένου καταλόγου

- **Στόχος:** η εξοικείωση με προχωρημένες τεχνικές προγραμματισμού στο κέλυφος Bash, και η εκτίμηση της ευκολίας με την οποία μπορεί κανείς να δημιουργήσει ένα σύνθετο προγράμματα, μέσω system utilities.
  - *πίνακες, διαχείριση σημάτων (signals), συνθήκες ελέγχου, δομές επανάληψης, regular expressions, stream editors (sed, awk) και χρήση συναρτήσεων με τα προαναφερθέντα.*

- **Σκοπός:** Κυρίως σκοπός αυτής της άσκησης είναι η δημιουργία ενός λεξικού με τη συχνότητα εμφάνισης λέξεων στο σύνολο ιστοσελίδων που δίδονται σε δομημένο κατάλογο. Πέραν του λεξικού θα πρέπει να δημιουργηθούν αρχεία (ένα για κάθε αρχείο-ιστοσελίδα της δομημένης ιεραρχίας) τα οποία θα περιέχουν τα links και την περιγραφή τους καθώς και ένα αρχείο το οποίο θα περιέχει όλα τα ονόματα των αρχείων που δεν είναι ιστοσελίδες.

Δημιουργία Λεξικού

- η **δημιουργία ενός φίλτρου** το οποίο επεξεργάζεται κάθε ιστοσελίδα και εξάγει όλες τις λέξεις.
  - Στις λέξεις αυτές, δεν περιλαμβάνονται τα HTTP headers, τα HTML tags, και οι ειδικοί χαρακτήρες HTML
  - **HTML TAG:** Οτιδήποτε περικλείεται μεταξύ των συμβολών < >.
    - π.χ. <html> , <td bgcolor="red" width="100%">
  - **HTML Ειδικοί Χαρακτήρες:** Οτιδήποτε περικλείεται μεταξύ & και ;
    - π.χ. **& &nsbp; Á**
  - **HTTP headers:** Οτιδήποτε περικλείεται μεταξύ των συμβολών HTML TAGS <head> και </head>
- Το σύστημα δημιουργεί πάντοτε το λεξικό, ανεξάρτητα εάν διακοπεί η λειτουργία του προγράμματος από κλείσιμο του κελύφους.
- Το λεξικό είναι case-insensitive.

- **Σημείωση**: Το βάθος της διερεύνησης DEPTH, καθώς επίσης ο κατάλογος εκκίνησης A πρέπει να δίδονται από τον χρήστη υπό μορφή command-line parameters.

Δημιουργία αρχείων συνδέσμων

- Για κάθε δοθείσα ιστοσελίδα, πρέπει να βρείτε όλους τους συνδέσμους που βρίσκονται σε αυτή.
  - Δημιουργία του δομημένου καταλόγου κάτω από το ~/tmp/data/, όπου θα δημιουργούνται και θα φυλάγονται στην αντίστοιχη θέση τα αρχεία με τους συνδέσμους
  - Οι σύνδεσμοι αυτοί μπορεί να είναι απόλυτοι (π.χ. http://www.cs.ucy.ac.cy/People/graduate.html) ή σχετικοί, με βάση το όνομα του διαθέτη (π.χ. People/graduate.html).
  - Σημειώστε ότι δεν είναι απαραίτητο ο σύνδεσμος να εσωκλείετε μέσα σε μονά ή διπλά εισαγωγικά.
  - Επίσης, δεν είναι απαραίτητο να βρίσκετε σε μια γραμμή ένας σύνδεσμος (μπορεί να βρίσκετε σε περισσότερες γραμμές. (αλλά αυτό είναι προαιρετικό για την άσκηση)

Graduate

<A HREF="**http://www.ucy.ac.cy/cc/index.html**"><B><FONT COLOR="#FFFFFF" FACE="Times New Roman,Times,Times NewRoman">**Information Systems Services**</FONT></B></A>

<a href="**prices.htm**" target="_top"
onClick="MM_nbGroup('down','group1','prices','images/buttons/price_or.gif',1)"
onMouseOver="MM_nbGroup('over','prices','images/buttons/price_or.gif','images/buttons/price_or.gif',1)"
onMouseOut="MM_nbGroup('out')"><img src="images/buttons/price.gif" alt="Hotel Apts Price List"
name="prices" border="0" onload="">

- Δημιουργία του αρχείου **nodata.txt** το οποίο περιέχει τα ονόματα όλων των αρχείων που δεν είναι ιστοσελίδες (*.html ή *.htm).

Εργαστηριακή Άσκηση 2
Γενικές Οδηγίες

- Το σύστημα δεν αφήνει ποτέ άχρηστα και μεταβατικά αρχεία στον δίσκο, ανεξάρτητα εάν διακοπεί η λειτουργία του προγράμματος από κλείσιμο του κελύφους;
- Το σύστημα πρέπει να χρησιμοποιεί τεχνικές δομημένου προγραμματισμού με την χρήση συναρτήσεων
- Το σύστημα πρέπει να ελαχιστοποιεί την χρήση πόρων του συστήματος (αρχεία, μνήμης, κτλ);
- Το σύστημα πρέπει να μειώνει όσο το δυνατό περισσότερο τον χρόνο επεξεργασίας των δεδομένων.
- Σημειώστε ότι η περιγραφή της άσκησης αποτελεί την ελάχιστη προδιαγραφή του συστήματος που πρέπει να παραδώσετε.

Παραδείγματα Προγραμματισμού Bash
Υπολογισμός παραγοντικού ενός αριθμού

Να γραφεί ένα πρόγραμμα για το κέλυφος Bash που να υπολογίζει το παραγοντικό (x!) ενός ακεραίου αριθμού x, **επαναληπτικά**

```
[hpollas@cs4118 bash ]$ cat paragondiko
#!/bin/bash
echo -n "Give input number: "
read n

((m=n)) # copy variable
result=1

until [ $m -eq 0 ]
do
  ((result*=m))
  ((m--))
done

echo "Factorial of $n is $result"
```

Υπολογισμός παραγοντικού ενός αριθμού
Αναδρομική Λύση

```
#!/bin/bash

fact () {
# Variable "number" must be declared as local,
# otherwise this doesn't work.
local number=$1

if [ "$number" -eq 0 ]; then
   factorial=1 # Factorial of 0 = 1.
else
   ((decrnum=number-1))
   fact $decrnum # Recursive function call
   ((factorial = number * $?))
fi

return $factorial
}

echo -n "Give input number: "
read n
((m=n)) # copy variable.still an integer

result=1

fact $n

echo "Factorial of $n is $?."
```

Παραδείγματα Προγραμματισμού Bash
Redirecting stdin using exec

```
#!/bin/bash

exec 6<&0
exec < data-file # stdin replaced by file "data-file"

read a1 # Reads first line of file "data-file".
read a2 # Reads second line of file "data-file."

echo
echo "Following lines read from file."
echo "-------------------------------"
echo $a1
echo $a2
echo; echo; echo

exec 0<&6 6<&-

echo -n "Enter data "

read b1 # Now "read" functions as expected, reading from normal stdin.

echo "Input read from stdin."
echo "----------------------"
echo "b1 = $b1"
echo

exit 0
```

Παραδείγματα Προγραμματισμού Bash
Redirecting stdout using exec

```
#!/bin/bash
# reassign-stdout.sh

LOGFILE=logfile.txt
exec 6>&1 # Link file descriptor #6 with stdout.

# Saves stdout.
exec > $LOGFILE # stdout replaced with file "logfile.txt".
# ------------------------------------------------------------ #
# All output from commands in this block sent to file $LOGFILE.

echo -n "Logfile: "
date
echo "-------------------------------------"
echo
echo "Output of \"ls -al\" command"
echo
ls -al
echo; echo
echo "Output of \"df\" command"
echo
df
# ------------------------------------------------------------ #
exec 1>&6 6>&- # Restore stdout and close file descriptor #6.
echo
echo "== stdout now restored to default == "
echo
ls -al
echo

exit 0
```