

Laboratorio di Elementi di Architetture e Sistemi Operativi

Esercizi del 28 Marzo 2012

Esercizio 1. *Scrivere uno script chiamato `menu.sh` che mostri il seguente menù a schermo:*

```
PROGRAMMI DISPONIBILI
1. Firefox
2. Openoffice
3. Gedit

0. Esci
```

e chieda all'utente quale programma eseguire. Dopo aver eseguito il programma, lo script termina.

```
#!/bin/bash
echo PROGRAMMI DISPONIBILI
echo 1. Firefox
echo 2. Openoffice
echo 3. Gedit
echo
echo 0. Esci
echo
echo -n "Quale programma devo eseguire? "
read Scelta
case $Scelta in
1) firefox ;;
2) ooffice ;;
3) gedit ;;
0) echo "Arrivederci!" ;;
*)echo "Scelta non prevista nel menu!"
  exit 1 ;;
esac
```

Esercizio 2. *Modificare lo script `creascript.sh` dell'Esercizio 6 della lezione scorsa aggiungendo il controllo dei parametri e la gestione degli errori. In particolare:*

- 1. controllare che il numero di parametri passati allo script sia corretto,*
- 2. se il file da creare esiste già, chiedere all'utente se desidera sovrascriverlo.*

```
#!/bin/bash
if [ $# -ne 1 ] ; then
  echo "Uso: creascript.sh nomescrpt"
  exit 1
fi
nomefile=$1.sh
if [ -e $nomefile ] ; then
  echo -n "Il file " $nomefile "esiste già. Sovrascrivere (s/n) ? "
  read Scelta
  if [ $Scelta != "s" -a $Scelta != "S" ] ; then
    exit 1
```

```
    fi
fi
echo "#!/bin/bash" > $nomefile
chmod 711 $nomefile
```

Esercizio 3. Scrivere uno script chiamato `quadrato.sh` che prenda in input un parametro intero con valore tra 2 e 15 e disegni sullo standard output un quadrato (utilizzando i caratteri `+`, `-` e `|`) come nel seguente esempio:

```
$ quadrato.sh 4
+--+
|  |
|  |
+--+
```

Lo script deve controllare che il numero di parametri ed il loro valore sia giusto.

```
#!/bin/bash
if [ $# -ne 1 ] ; then
    echo 'Utilizzo dello script: quadrato n'
    exit 1
fi
# controlla che il parametro sia un numero
test $1 -eq $1 2> /dev/null
if [ $? -ne 0 ] ; then
    echo 'Il parametro deve essere un numero'
    exit 2
fi
if [ $1 -lt 2 -o $1 -gt 15 ] ; then
    echo 'Il parametro deve essere un numero tra 2 e 15'
    exit 2
fi
n=$1
for i in $(seq $n) ; do
    for j in $(seq $n) ; do
        if [ $i -eq 1 -o $i -eq $n ] ; then
            # sto scrivendo la prima o l'ultima riga
            if [ $j -eq 1 -o $j -eq $n ] ; then
                # sto scrivendo il primo o l'ultimo carattere
                echo -n "+"
            else
                echo -n "-"
            fi
        else
            # sono nel corpo del quadrato
            if [ $j -eq 1 -o $j -eq $n ] ; then
                # sto scrivendo il primo o l'ultimo carattere
                echo -n "|"
            else
                echo -n " "
            fi
        fi
    done
done
```

```

        fi
    fi
done
# vai a capo
echo
done

```

Esercizio 4. Scrivere uno script che accetta come parametri una sequenza di nomi di file (ordinari o directory) e li rimuove spostandoli nella cartella `.cestino` della propria home (invece di cancellarli fisicamente). Nel caso in cui la directory `.cestino` non esista lo script dovrà crearla. Nel caso in cui i file da rimuovere abbiano dei nomi che corrispondono ad altri file già presenti in `.cestino`, lo script dovrà chiedere all'utente se desidera sovrascriverli. Si implementino inoltre le seguenti opzioni:

- `-l` che permette di visualizzare il contenuto della directory `.cestino`;
- `-e` che svuota la directory `.cestino`;
- `-r nomefile percorso` che ripristina il file `nomefile` spostandolo dalla directory `.cestino` a quella specificata da `percorso`. Nel caso in cui nella directory di destinazione esista già un file con lo stesso nome, lo script dovrà chiedere all'utente se desidera sovrascriverlo.

Si gestiscano opportunamente gli errori e si provi lo script su alcuni esempi significativi.

```

#!/bin/bash
# controllo argomenti
if [ $# -lt 1 ] ; then
    echo "Uso: cestina.sh file1 file2 ...."
    echo "      cestina.sh -l"
    echo "      cestina.sh -e"
    echo "      cestina.sh -r nomefile percorso"
    exit 1
fi

# controlla l'esistenza del cestino. Se non c'è lo crea
cestino=$HOME/.cestino
if [ ! -e $cestino ] ; then
    mkdir $cestino
    if [ $? != 0 ] ; then
        echo "ERRORE: impossibile creare il cestino."
        exit 1
    fi
fi

# controlla il primo argomento e vedi se è un'opzione
case $1 in
"-l") # Elenca i file in .cestino
    if [ $# -ne 1 ] ; then
        echo "Uso: cestina.sh -l"
        exit 1
    fi
    echo "CONTENUTO DEL CESTINO:"
    ls $cestino
    ;;
"-e") # Svuota il cestino

```

```

if [ $# -ne 1 ] ; then
    echo "Uso: cestina.sh -p"
    exit 1
fi
rm -rf $cestino/*
echo "CESTINO SVUOTATO."
;;
"-r") # recupera il file
if [ $# -ne 3 ] ; then
    echo "Uso: cestina.sh -r nomefile percorso"
    exit 1
fi
nomefile=$2
percorso=$3
if [ ! -e $cestino/$nomefile ] ; then
    echo "ERRORE: il file da recuperare non è presente nel cestino."
    exit 1
fi
if [ ! -d $percorso ] ; then
    echo "ERRORE: il secondo argomento deve essere una directory."
    exit 1
fi
if [ -e $percorso/$nomefile ] ; then
    echo -n "Il file esiste già! Sovrascrivere? (s/n): "
    read risp
    if [ $risp != "s" -a $risp != "S" ] ; then
        echo "recupero annullato."
        exit 1
    fi
fi
echo "recupero il file " $nomefile
mv -f $cestino/$nomefile $percorso
;;
*) # nessun argomento, sposta i file nel cestino
for f in $* ; do
    echo "Cancellazione di " $f
    if [ ! -e $f ] ; then
        echo "ERRORE: il file " $f " non esiste."
    elif [ -e $cestino/$(basename $f) ]
    then
        echo -n "Il file " $f "esiste già nel cestino! Sovrascrivere? (s/n): "
        read risp
        if [ $risp != "s" -a $risp != "S" ] ; then
            echo "Rimozione di " $f " annullata."
            exit 1
        else
            mv -f $f $cestino
        fi
    else
        mv -f $f $cestino
    fi
else

```

```
mv -f $f $cestino
fi
done
ii
esac
```