- Script shell -

- Controllo di flusso: for -

- Esempi d'uso -

- Ciclo **for**

```
for var in lista
   do
      comando/i che usano $var
done
```

Esercizio n° 1

- Scrivere uno script shell (shell program) di nome **cercaFileReg** che, nella directory corrente, (di lancio) crea un file di nome **fileReg** contenente l' elenco di tutti i file regolari.

  Nota: (Creare una sottodirectory **bin** all'interno della propria work directory in cui mettere gli script)

  Suggerimenti:

  Usare $(comando) per assegnare il risultato del comando lista

Esempio di lancio dello script:

$ chmod +x cercaFileReg.sh (permessi per esecuzione)

$ ./cercaFileReg.sh

Soluzione Esercizio n° 1

Script Shell:

```
#!/bin/bash

for file in $(ls)
do
if [ -f $file ]
then
    echo $file >> fileReg
fi
done
```

La prima linea dello script deve iniziare con #!, che indica al kernel che lo script è direttamente eseguibile, poi nome dell'interprete dei comandi shell (Bourne shell).

fornisce il contenuto della nostra directory

variabile da testare

vero se il file esiste ed è un file regolare

espressione condizionale if

Per effettuare un ciclo tra una lista di valori di tipo stringa si può usare il comando for

Esercizio n° 2

- Scrivere uno script shell di nome **copiaFileC** che, compia le seguenti operazioni:
  - Cerca tutti i file con estensione **.c** nella home directory e in tutte le sottodirectory
  - Crea una cartella **fileC** nella propria directory di lavoro e copia i file trovati in questa cartella
    - Lo script dovrà anche produrre il numero di file trovati

- Suggerimenti:
  - Usare comando find
  - Usare comando grep
  - Usare $(comando) per assegnare il risultato della ricerca

Soluzione Esercizio n° 2

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

interprete dei comandi shell

```
myhome=$HOME

echo "Numero file trovati $(find $myhome -name "*.c" |
grep -c "\.c")"
cd ..
mkdir fileC

for file in $(find $home -name "*.c")
do
if [ -f $file ];
then
    cp  $file fileC
fi
done
```

vero se il file esiste ed è un file regolare

Esercizio n° 3

- Scrivere uno script shell di nome **copiaFile** che prende come argomento una stringa **<str>** e
  - Cerca tutti i file con estensione **.<str>** nella home directory e in tutte le sottodirectory
  - Crea una cartella **file<str>** nella propria directory di lavoro e copia i file trovati in questa cartella
    - Lo script dovrà anche produrre il numero di file trovati

- Suggerimenti:
  - Usare comando find
  - Usare comando grep
  - Usare $(comando) per assegnare il risultato della ricerca
  - $ ./copiaFile.sh <str>

Soluzione Esercizio n° 3

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

interprete dei comandi shell

```
myhome=$HOME
echo "Numero file trovati $(find $myhome -name "*.$1" |
grep -c "\.$1$")"
cd ..
mkdir file$1

for file in $(find $home -name "*.$1")
do
if [ -f $file ];
then
     cp  $file file$1
fi
done
```

vero se il file esiste ed è un file regolare

Esercizio n° 4

- Scrivere uno script shell **cancellaFile** che prende una stringa **<str>** come argomento, e nella directory corrente, (di lancio) cancella tutti i file che terminano con **.<str>**

  (cancellare interattivamente con rm –i)

Soluzione

```
#!/bin/bash
for file in $(ls *.$1)
do
   if [ -f $file ];
   then
        rm -i  $file
   fi
done
```

Esercizio n° 5

- Scrivere uno script shell **cercaDirectory** che, nella directory corrente, (di lancio) crea un file di nome **fileDir** contenente l' elenco di tutte le directory il cui nome inizia per lettera maiuscola

Soluzione Esercizio n° 5

```
#!/bin/bash
for file in $(ls)
do
if [ -d $file ]
then
    a=$(echo $file | cut -c1)
    b=-$(echo $a | grep "[A-Z]")
    if [ $b != - ]
    then
        echo $file >> fileDir
    fi
fi
done
```

vero se il file esiste ed è una directory

in questo modo si seleziona solamente il primo caratteri di ogni linea. (la prima lettera del nome della directory)

Espressione regolare

File contenente l'elenco delle directory il cui nome inizia per lettera maiuscola

- Controllo di flusso: while -

- Esempi d'uso -

- Ciclo **while** esegue la lista di comandi finchè la condizione è vera

```
while condition;
do
    comandi
done
```

- Creazione file **touch**:
  - **touch** nomeFile

Esercizio n° 6 - while -

- Si realizzi uno script "**scriviNumeri.sh**" che stampa in stdout numeri da 0 a N: **0,1,2,…….,N-1**

  Il valore di N viene passato allo script da riga di comando.
- Esempio di lancio: $ ./scriviNumeri.sh N

Soluzione

```
#!/bin/bash
COUNTER=0
while [  $COUNTER -lt $1 ];
do
  echo il valore di counter è
$COUNTER
   COUNTER=$((COUNTER+1))
done
```

Esercizio n° 7 - while -

- Si realizzi uno script che chiameremo "**creaFiles.sh**" che genera n file vuoti denominati:

  node1.txt, **node2.txt**,…**nodeN.txt**

  nella directory di lancio contenenti il proprio nome.
  Il valore di N viene passato allo script da riga di comando.

- Esempio di lancio:

  $ ./creaFiles.sh N

Soluzione Esercizio n° 7 - while -

```
#!/bin/bash
if  test $# -ne 1
then
   echo "Wrong number of parameters $#"
   echo "Usage: $0 param"
fi

i=0
while [ $i -lt $1 ];
do
     i=$((i+1))
     touch node$i.c
     echo node$i.c > node$i.c
done
```

Inizializzazione della variabile di ciclo

Entra nel ciclo fintanto che la variabile i è minore di n (less than)

Creazione file

- Controllo di flusso: until -

- Esempi d'uso -

- Ciclo **until** esegue la lista di comandi finchè la condizione è falsa

```
until condition;
do
        comandi
done
```

- Alcuni test relativi alle proprietà dei file :
  - **-e** file esiste
  - **-d** file directory
  - **-f** il file esiste ed è regolare

Esercizio n° 8 - until -

- Si realizzi uno script che chiameremo "**scriviNumeri.sh**" che scrive in stdout i numeri da 20 a 10: **20,19,18,……..,10**
- Esempio di lancio: $ ./scriviNumeri.sh

Soluzione

```
#!/bin/bash
COUNTER=20
until [ $COUNTER -lt 10 ];
do
  echo COUNTER: $COUNTER
  COUNTER=$((COUNTER-1))
done
```

Esercizio n° 9 – until -

- Scrivere uno script shell **copiaFile.sh** che riceve da riga di comando due parametri il file da copiare e la directory di destinazione. Lo script dovrà copiare il file nella directory controllando il numero di parametri passati, l'esistenza del file ed il fatto che la directory di copia sia una sottodirectory della work directory.

- Esempio di lancio:

  $ ./copiaFile.sh <file> <dir>

Soluzione Esercizio n° 9 - until -

```
#!/bin/bash
if [ "$#" -ne 2 ]; then
echo "bash: $0: wrong number of arguments."; exit 1
fi
file=$1; destdir=$2
if  ! [ -d "$destdir" ]  ||  ! [ -f "$file" ];
then
   echo  "bash:  copiaFile:  Usage  copiaFile  <file>
          <dir>."; exit 1
else
until cp $file $destdir; do
   echo "Attempt to copy failed. Waiting ...";
   sleep 5
done
fi
```

Condizione comando cp

ciclo until

Se la copia fallisce attende 5 secondi prima di riprovare

- Fine Esercitazione -