

Settima esercitazione

Shell scripting

Agenda

Esempio

Esplorazione completa di una directory: script bash con ricorsione.

Esercizio 1

Esplorazione ricorsiva del file system

Esercizio 2

Esplorazione ricorsiva di più sottoalberi

Esempio – Script ricorsivi

Si scriva uno script bash avente interfaccia di invocazione

`recurse_dir.sh dir`

Il programma, dato un direttorio in ingresso **`dir`**, deve stampare su stdout l'elenco dei file contenuti nel direttorio e in tutti i suoi sottodirettori (analogamente al comando **`ls -R`**)

Schema di soluzione ricorsiva

`recurse_dir.sh arg1`

caso **base**

arg1 è un file

→ ne stampo il nome assoluto

caso generale espresso in termini **ricorsivi**

arg1 è una directory

→ mi muovo nella directory **arg1**;

per ogni file (normale o directory) **invoco**
nuovamente recurse_dir.sh

Bozza di soluzione

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

```
if ! test -d "$1" ; then  
    echo `pwd` /$1
```

**Caso
base**

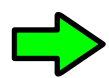
```
else  
    cd "$1"  
    for f in * ; do  
        "$0" "$f"  
    done  
fi
```

**Caso
generale**

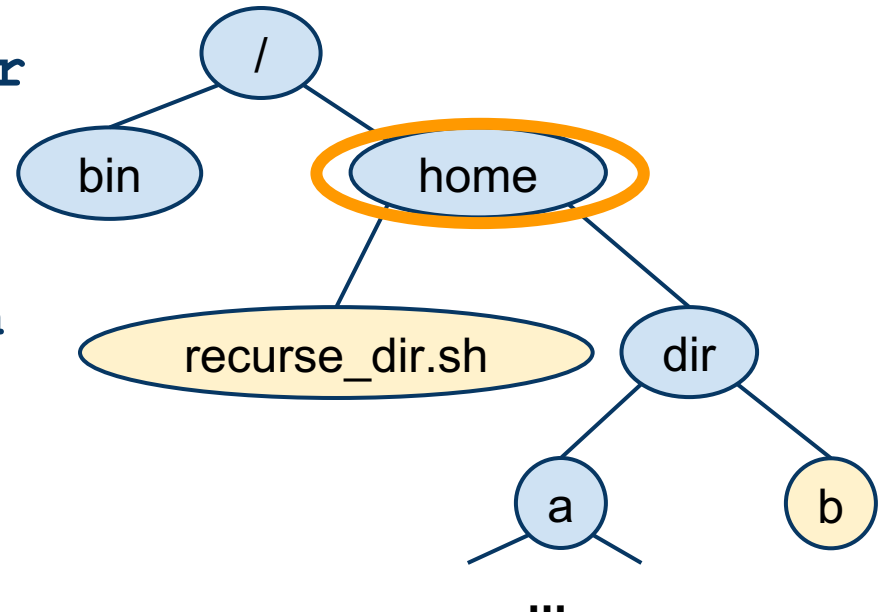
Chiamata ricorsiva

Ricorsione (1/6)

```
$ pwd  
/home  
$ /home/recurse_dir.sh dir
```



```
if ! test -d "$1" ; then  
    echo `pwd` /$1  
else  
    cd "$1"  
    for f in * ; do  
        "$0" "$f"  
    done  
fi
```



□ directory
□ file

VARIABILI:

\$PWD /home

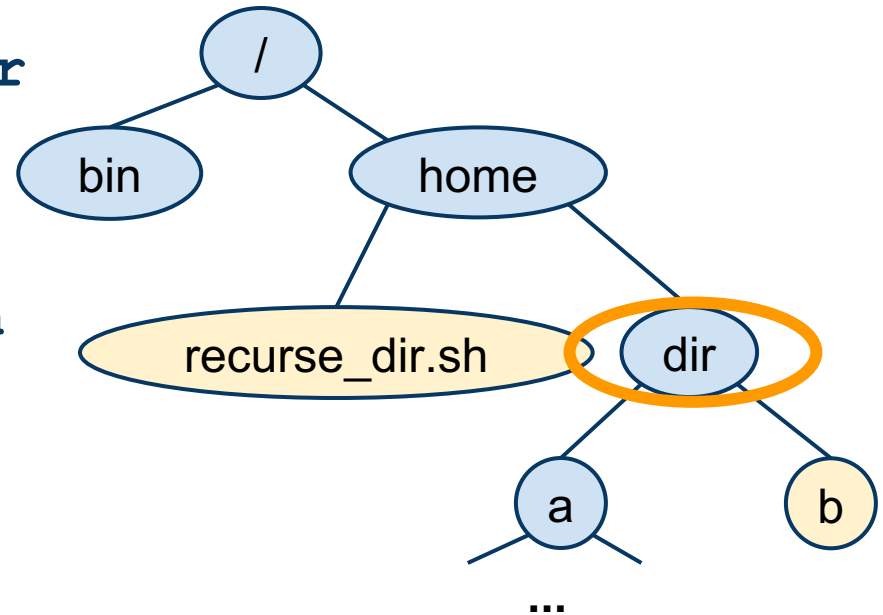
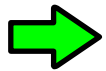
\$0 /home/recurse_dir.sh

\$1 dir

Ricorsione (2/6)

```
$ pwd  
/home  
$ /home/recurse_dir.sh dir
```

```
if ! test -d "$1" ; then  
    echo `pwd` /$1  
else  
    cd "$1"  
    for f in * ; do  
        "$0" "$f"  
    done  
fi
```



□ directory
□ file

VARIABILI:

\$PWD **/home/dir**

\$0 /home/recurse_dir.sh

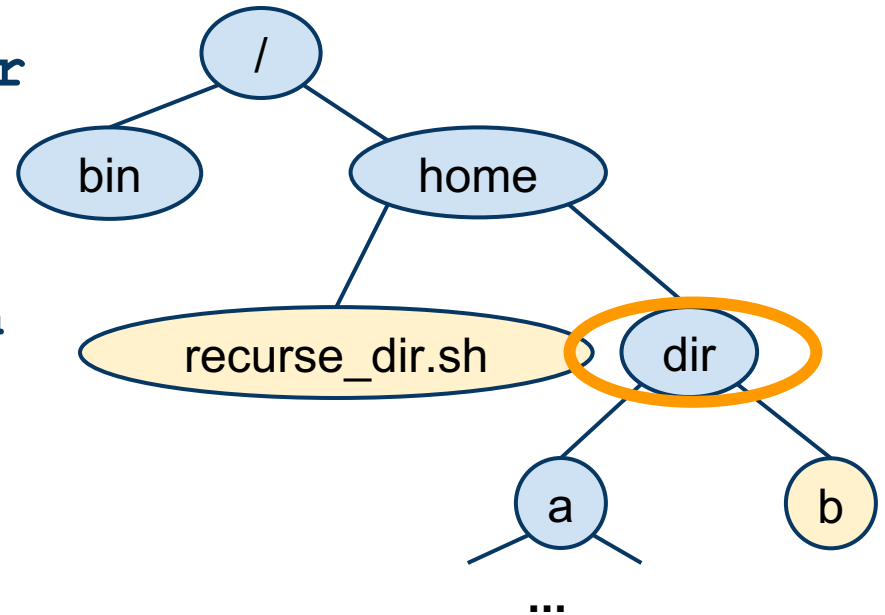
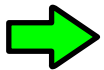
\$1 dir

Ricorsione (3/6)

```
$ pwd
/home
$ /home/recurse_dir.sh dir
```

```
$ /home/recurse_dir.sh a
```

```
if ! test -d "$1" ; then
    echo `pwd` /$1
else
    cd "$1"
    for f in * ; do
        "$0" "$f"
    done
fi
```



□ directory
□ file

VARIABILI:

\$PWD /home/dir

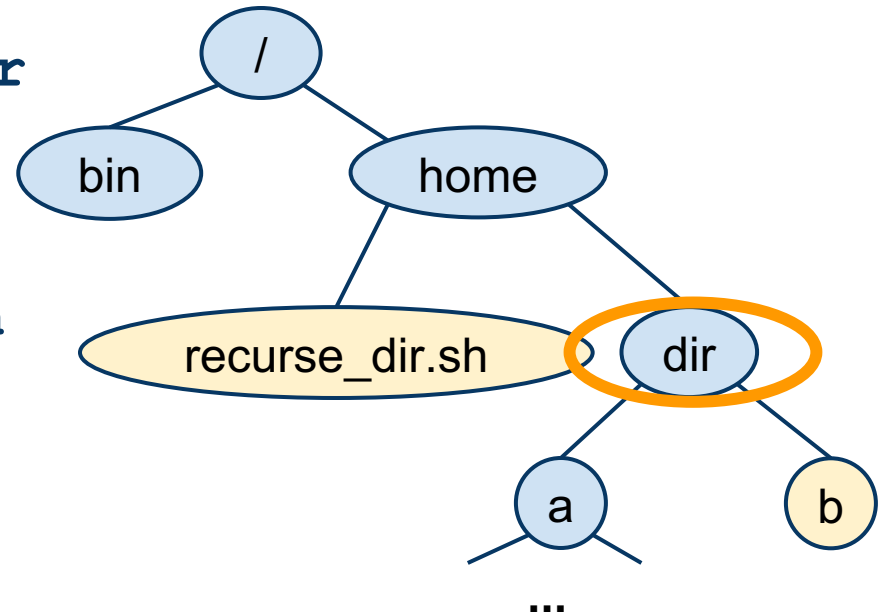
\$0 /home/recurse_dir.sh

\$1 dir

Ricorsione (4/6)

```
$ pwd  
/home  
$ /home/recurse_dir.sh dir
```

```
➔ if ! test -d "$1" ; then  
    echo `pwd` /$1  
else  
    cd "$1"  
    for f in * ; do  
        "$0" "$f"  
    done  
fi
```



□ directory
□ file

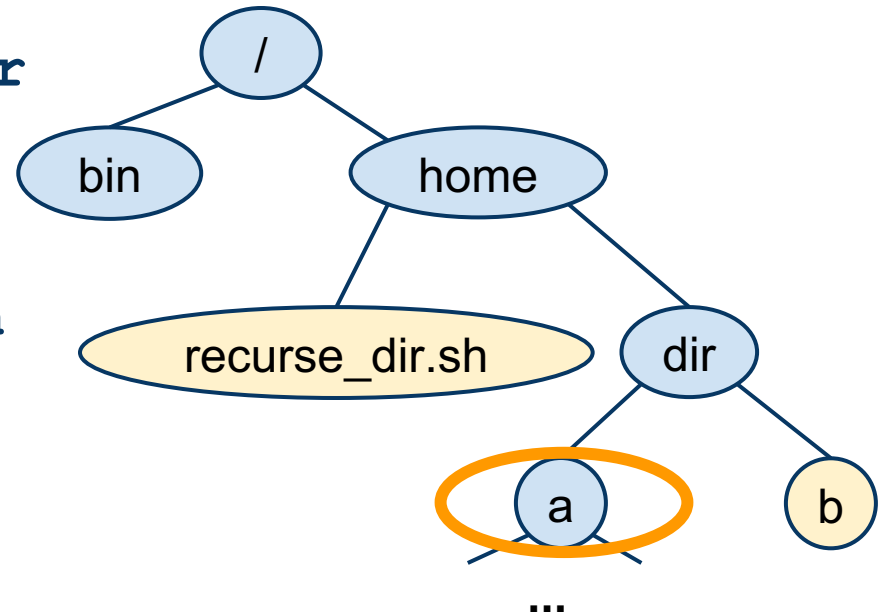
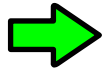
VARIABILI:

```
$PWD /home/dir  
$0 /home/recurse_dir.sh  
$1 a
```

Ricorsione (5/6)

```
$ pwd
/home
$ /home/recurse_dir.sh dir
```

```
if ! test -d "$1" ; then
    echo `pwd` /$1
else
    cd "$1"
    for f in * ; do
        "$0" "$f"
    done
fi
```



□ directory
□ file

VARIABILI:

\$PWD **/home/dir/a**

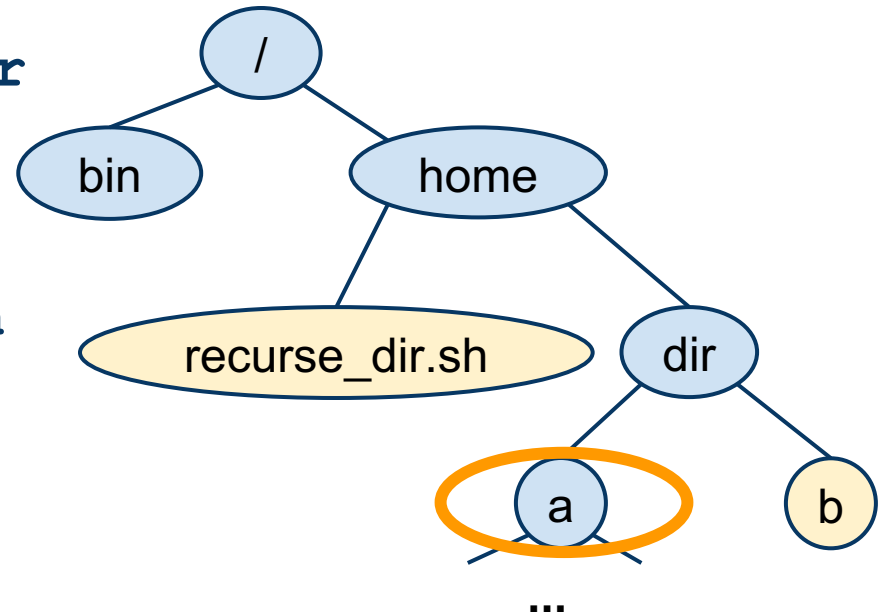
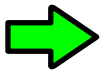
\$0 /home/recurse_dir.sh

\$1 a

Ricorsione (6/6)

```
$ pwd
/home
$ /home/recurse_dir.sh dir
```

```
if ! test -d "$1" ; then
    echo `pwd` /$1
else
    cd "$1"
    for f in * ; do
        "$0" "$f"
    done
fi
```



□ directory
□ file

VARIABILI:

\$PWD /home/dir/a

\$0 /home/recurse_dir.sh

\$1 ...

ATTENZIONE

Nell'esempio lo script è stato invocato **specificando il suo nome assoluto**:

```
$ /home/recurse_dir.sh dir
```

Cosa succederebbe invocandolo
con un **nome relativo**?

```
$ ./recurse_dir.sh dir
```

Ricorsione - alternativa (1/3)

```
$ pwd
```

```
/home
```

```
$ ./recurse_dir.sh dir
```

```
if ! test -d "$1" ; then
```

```
    echo `pwd` /$1
```

```
else
```

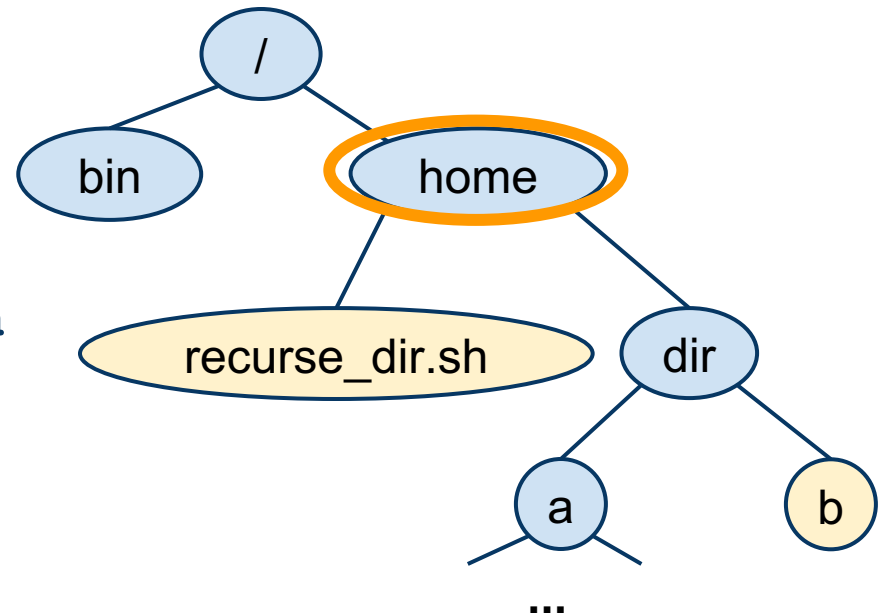
```
    cd "$1"
```

```
    for f in * ; do
```

```
        "$0" "$f"
```

```
    done
```

```
fi
```



□ directory
□ file

VARIABILI:

\$PWD /home

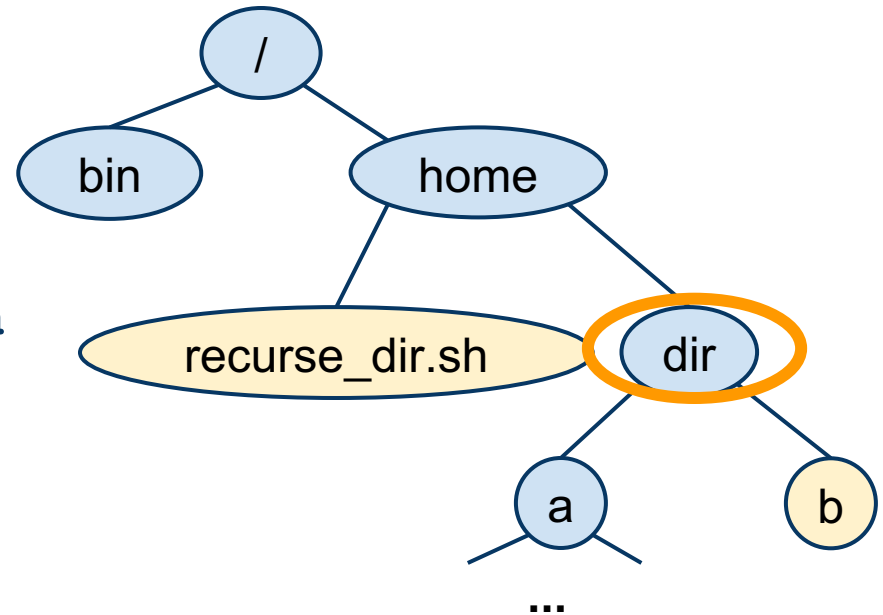
\$0 ./recurse_dir.sh

\$1 dir

Ricorsione - alternativa (2/3)

```
$ pwd  
/home  
$ ./recurse_dir.sh dir
```

```
if ! test -d "$1" ; then  
    echo `pwd` /$1  
else  
    ➔ cd "$1"  
    for f in * ; do  
        "$0" "$f"  
    done  
fi
```



□ directory
□ file

VARIABILI:

\$PWD /home/dir

\$0 ./recurse_dir.sh

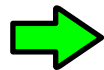
\$1 dir

Ricorsione - alternativa (3/3)

```
$ pwd
/home
$ ./recurse_dir.sh dir
```

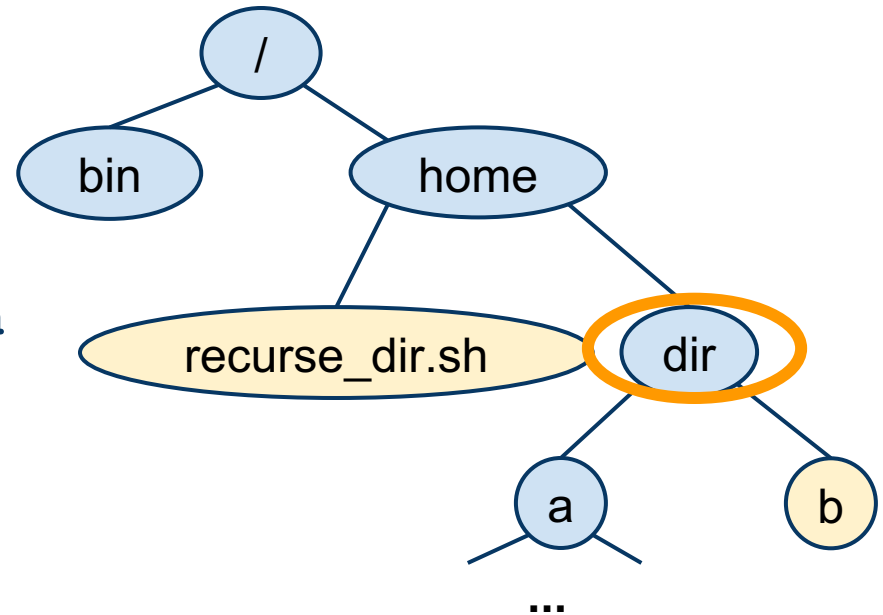
```
$ ./recurse_dir.sh a
```

```
if ! test -d "$1" ; then
    echo `pwd` /$1
else
    cd "$1"
    for f in * ; do
        "$0" "$f"
    done
fi
```



```
"$0" "$f"
```

non
funziona!



□ directory
□ file

VARIABILI:

```
$PWD /home/dir
$0 ./recurse_dir.sh
$1 dir
```

Come risolvere?

Problema: Un valore dipendente dalla directory di lavoro corrente (un percorso relativo) viene "**propagato**" da una invocazione ricorsiva all'altra (tramite la variabile \$0)

La directory di lavoro però cambia (perchè usiamo il comando `cd` nel codice)

Possibile soluzione: Prima di iniziare la ricorsione memorizzare la directory di partenza in una variabile che verrà usata per le invocazioni ricorsive

Occorre creare:

- **Script ricorsivo**
- **Script di invocazione:**
 - Controlla i parametri
 - Salva in maniera "stabile" il percorso dello script ricorsivo
 - Innesca la ricorsione

Struttura di un file comandi ricorsivo

invoker.sh

#!/bin/sh

Controllo degli argomenti

Invocazione del file comandi ricorsivo do_recursive.sh

do_recursive.sh

#!/bin/sh

Esecuzione del compito

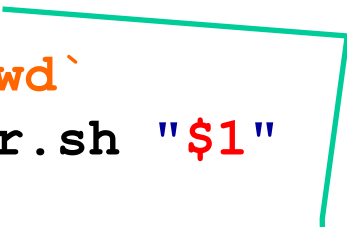
Invocazione del file comandi ricorsivo do_recursive.sh

Script di invocazione

recurse_dir.sh

```
#!/bin/bash
# ... controllo argomenti

oldpath=$PATH
PATH=$PATH:`pwd`
do_recurse_dir.sh "$1"
PATH=$oldpath
```



do_recurse_dir.sh

```
#!/bin/bash
if ! test -d "$1" ; then
    echo `pwd`/$1
else
    cd "$1"
    for f in * ; do
        "$0" "$f"
    done
fi
```

PATH è una variabile d'ambiente che contiene dei nomi di directory separati da ":".

Quando lancio un comando senza alcun path (né assoluto né relativo, es: invoco **ls** invece di **/bin/ls**), il SO cerca quel comando in tutte le directory contenute nella variabile **PATH**.

Script di invocazione

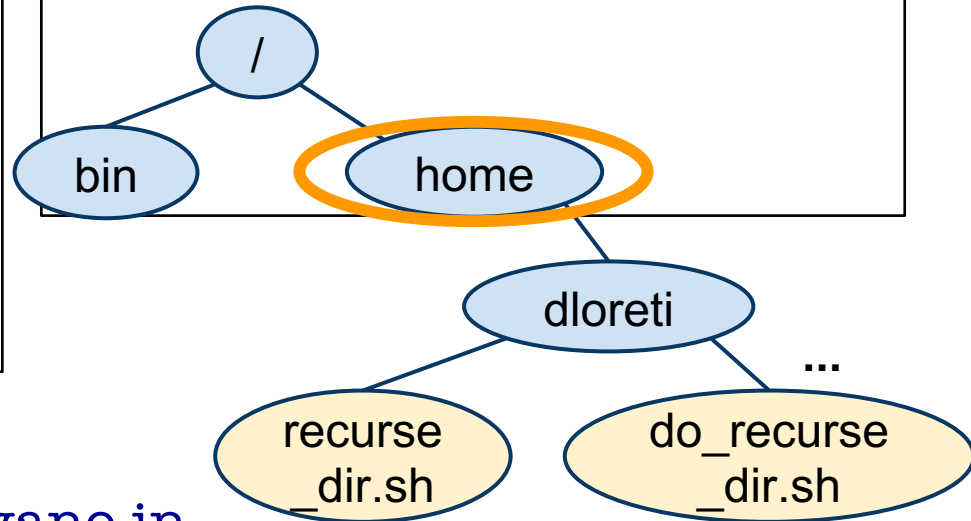
recurse_dir.sh

```
#!/bin/bash
# ... controllo argomenti

oldpath=$PATH
PATH=$PATH:`pwd`
do_recurse_dir.sh "$1"
PATH=$oldpath
```

do_recurse_dir.sh

```
#!/bin/bash
if ...
```



Problema :

Che succede se gli script si trovano in
/home/dloreti e l'utente li invoca dalla directory corrente
/home con il path relativo: ./dloreti/recurse_dir.sh

=> `pwd` viene espanso in "/home"

=> **PATH=\$PATH:/home**

=> do_recurse_dir.sh viene cercato in /home

→ **NON TROVATO!**

Soluzione generale

recurse_dir.sh

```
#!/bin/bash
# ... controllo argomenti
```

```
if [[ "$0" = /* ]] ; then
```

```
# se $0 è un path assoluto
```

```
dir_name=`dirname "$0"`
```

```
recursive_cmd="$dir_name/do_recurse_dir.sh"
```

```
elif [[ "$0" = */* ]] ; then
```

```
# se c'è uno slash, ma non inizia con /
```

```
# $0 è un path relativo
```

```
dir_name=`dirname "$0"`
```

```
recursive_cmd="`pwd`/$dir_name/do_recurse_dir.sh"
```

```
else
```

```
# Non si tratta nè di un path relativo, nè di uno
```

```
# assoluto, il comando $0 sarà cercato in $PATH.
```

```
recursive_cmd=do_recurse_dir.sh
```

```
fi
```

```
#Invoco il comando ricorsivo
```

```
"$recursive_cmd" "$1"
```

Restituisce \$0 tranne
l'ultimo / e ciò che segue

do_recurse_dir.sh

```
#!/bin/bash
if ...
```

Soluzione generale

recurse_dir.sh

```
#!/bin/bash
# ... controllo argomenti
```

```
if [[ "$0" = /* ]] ; then
    # se $0 è un path assoluto
    dir_name=`dirname "$0"`
    recursive_cmd="$dir_name/do_recurse_dir.sh"
```

```
elif [[ "$0" = */* ]] ; then
```

```
# se c'è un
```

```
# $0 è un p
```

```
dir_name=`d
```

```
recursive_c
```

```
else
```

```
# Non si tratta ne di un path relativo, ne di uno
```

```
# assoluto, il comando $0 sarà cercato in $PATH.
```

```
recursive_cmd=do_recurse_dir.sh
```

```
fi
```

```
#Invoco il comando ricorsivo
```

```
"$recursive_cmd" "$1"
```

Esempio:

\$0=/home/recurse_dir.sh

`dirname "\$0"` → /home

\$recursive_cmd=/home/do_recurse_dir.sh

Soluzione generale

recurse_dir.sh

```
#!/bin/bash
# ... controllo argomenti

if [[ "$0" = /* ]] ; then
    # se $0 è un path assoluto
    dir_name=`dirname "$0"`
    recursive_cmd="$dir_name/do_recurse_dir.sh"
elif [[ "$0" = */* ]] ; then
    # se c'è uno slash, ma non inizia con /
    # $0 è un path relativo
    dir_name=`dirname "$0"`
    recursive_cmd="`pwd`/$dir_name/do_recurse_dir.sh"
```

else

```
# Non
# asso
recurs
```

fi

#Invoco

```
"$recursive_cmd" "$1"
```

Esempio:

\$0=../folder/recurse_dir.sh

`dirname "\$0"` → ../folder

\$recursive_cmd=/home/../folder/do_recurse_dir.sh

Soluzione generale

recurse_dir.sh

```
#!/bin/bash
# ... controllo argomenti

if [[ "$0" = /* ]] ; then
    # se $0 è un path assoluto
    dir_name=`dirname "$0"`
    recursive_cmd="`pwd`/$dir_name/do_recurse_dir.sh"
```

Esempio:

\$0=recurse_dir.sh

\$recursive_cmd=do_recurse_dir.sh

```
dir_name=`dirname "$0"`
recursive_cmd="`pwd`/$dir_name/do_recurse_dir.sh"
```

else

Non si tratta nè di un path relativo, nè di uno
assoluto, il comando \$0 sarà cercato in \$PATH.

recursive_cmd=do_recurse_dir.sh

fi

#Invoco il comando ricorsivo

"\$recursive_cmd" "\$1"

Esercizio 1 – Esplorazione ricorsiva del file system (1/2)

Realizzare un file comandi (ricorsivo) che abbia la sintassi

`cerca dir G text`

dove:

- `dir` è il nome **assoluto** di un direttorio esistente nel file system
- `G` è una stringa che rappresenta il **groupname** di un gruppo di utenti
- `text` è una stringa

Esercizio 1 - (2/2)

Il compito del file comandi è quello di :

- Esplorare (ricorsivamente) il sottoalbero individuato da **dir**
- Individuare i file **ordinari** e **leggibili di proprietà del gruppo G**
- Per ogni file che rispetti tali caratteristiche **stampare** a video **il suo nome assoluto** e il **numero di occorrenze della stringa text** contenute in esso.

Esercizio 1 : suggerimenti

Prima di tutto realizzare la ricorsione e testare che funzioni correttamente!

Poi:

- Individuare il groupname del gruppo proprietario di un file. Due alternative:
 - → vedere il comando **stat** per ottenere gli attributi di un file (**man stat**): per il **groupname**: opzione **--format=%G**
 - → oppure usare **awk** per filtrare solo il groupname dall'out di **ls**
- contare il numero di occorrenze di una stringa in un file → vedere l'opzione **-o** del comando **grep** e l'opzione **-l** del comando **wc**

Esercizio 2 – Esplorazione

ricorsiva di N directory (1/2)

Realizzare un file comandi (ricorsivo) che abbia la sintassi

cerca G N OutFile dir1...dirN

dove:

G è una stringa che rappresenta il **groupname** di un gruppo di utenti;

N è un **intero** positivo;

OutFile è il nome **assoluto** di un file inizialmente non presente nel file system;

dir1...dirN sono nomi **assoluti** di direttori esistenti nel file system

Esercizio 2 - (2/2)

Il compito del file comandi è quello di eseguire una ricerca in tutte le directory **dir1...dirN**.

Per ogni directory **dir_i**:

- Esplorare (ricorsivamente) il sottoalbero individuato da **dir_i** allo scopo di individuare i file **ordinari** e **leggibili di proprietà del gruppo G**;
- Per ogni file che rispetti tali caratteristiche **aggiungere** al file **OutFile** **il suo nome assoluto** e stampare a video le **prime N righe** contenute in esso (vedere il comando **head**).

Al termine della ricerca su tutte le directory, il file comandi dovrà **stampare sullo standard output il numero di file** (che rispettano le caratteristiche date) **trovati nelle directory esplorate**.

Suggerimenti

- Invoker deve iterativamente attivare lo script ricorsivo sulle directory date. Ricordare:
 - "\$@" (o \$*) → lista delle variabili posizionali
 - **shift** → scorrimento a sinistra delle var posizionali

```
shift
shift
shift
for d in $@
do ... done
```
- alla fine occorre ricavare il numero totale dei file trovati → conteggio delle linee di **OutFile**. In che script fare questo conteggio, nello script ricorsivo o nell'invoker?