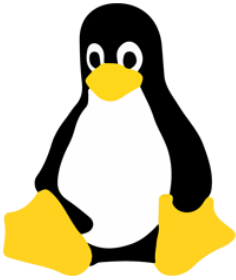

Systems Programming

Shell scripting

Most of the slides in this lecture are either from or adapted from the slides provided by Dr. Ahmad Barghash

Piping and redirection in shell scripting

Piping and redirection

Each program has these 3 streams

Piping and redirection is the means by which we may connect these streams.

Redirecting to a file

Don’t send the output to the terminal but to a file of your choice using (>)

Terminal

```
1.  user@bash: ls
2.  barry.txt bob example.png firstfile foo1 video.mpeg
3.  user@bash: ls > myoutput
4.  user@bash: ls
5.  barry.txt bob example.png firstfile foo1 myoutput video.mpeg
6.  user@bash: cat myoutput
7.  barry.txt
8.  bob
9.  example.png
10. firstfile
11. foo1
12. myoutput
13. video.mpeg
14. user@bash:
```

Redirecting from a file

If we use the less than operator (<) then we can send data the other way. We will read data from the file and feed it into the program via its STDIN stream.

Terminal

```
1. user@bash: wc -l myoutput
2. 8 myoutput
3. user@bash: wc -l < myoutput
4. 8
5. user@bash:
```

wc: words count in file(s)
-l: line count

Piping

- Instead of sending the program output to a file, you can send it as the input of another one
- The operator we use is (|)

```
Terminal
1. user@bash: ls
2. barry.txt bob example.png firstfile foo1 myoutput video.mpeg
3. user@bash: ls | head -3
4. barry.txt
5. bob
6. example.png
7. user@bash:
```

head -n: displays top n rows of the file

```
Terminal
1. user@bash: ls | head -3 | tail -1
2. example.png
3. user@bash:
```

tail -n: displays bottom n rows of the file

Let's start with real examples in Shell

Write a shell script to solve this problem

In a company, one of the programmer designed an application to work on a database of files. The only mistake the programmer did is that he sat a condition that all file names must be in small letters. Therefore, file names including capital letters are not accepted.

Task: Check files and rename file names where needed to make it all in small letters.

Basename: Get the file name without the path.

tr: translate, change, squeeze characters

echo |: instead of direct output to terminal, send the output to some statement or function to output it from there

mv: Rename SOURCE to DEST, or move SOURCE(s) to DIRECTORY

Example: Changes all filenames to lowercase

```
#!/bin/bash
for filename in *
                        # Traverse all files in directory.
do
                        # Get the file name without the path.
 fname=$(basename $filename)
                # Change name to lowercase.
 n=$(echo $fname | tr A-Z a-z)
 if [ $fname != $n ]
                # Rename only files not already lowercase.
 then
   mv $fname $n
 fi
done
exit 0
```

How to compare text files

Your task in a company is to compare text files created by a tool. Identical text files must be reported.

Use the “cmp” command

Returns 0 if equal, 1 if not, and 2 for trouble

Redirect the result message to /dev/null

Check your output by $?

Note: no commands allowed between cmd and output checking

Example: Compare two files with a script

```
#!/bin/bash
ARGS=2                                  # Two args to script expected.
if [ $# -ne “$ARGS” ]; then
  echo “Usage: $(basename $0) file1 file2” ; exit 1
fi
if [[ ! -r $1 || ! -r $2 ]] ;  then
  echo “Both files must exist and be readable.” ; exit 2
fi
                                        # /dev/null buries the output of the “cmp” command.
cmp $1 $2 > /dev/null
                                        # Also works with 'diff', i.e., diff $1 $2 > /dev/null
if [ $? -eq 0 ]                         # Test exit status of “cmp” command.
then
  echo “File $1 is identical to file $2.”
else
  echo “File $1 differs from file $2.”
fi
exit 0
```

Bash scripting Cheat sheet

https://devhints.io/bash

It is a useful link