UNIX

Lesson 04 : I/O redirection and Piping

Lesson Objectives

- In this lesson, you will learn:
  - Input output redirection
  - Pipies

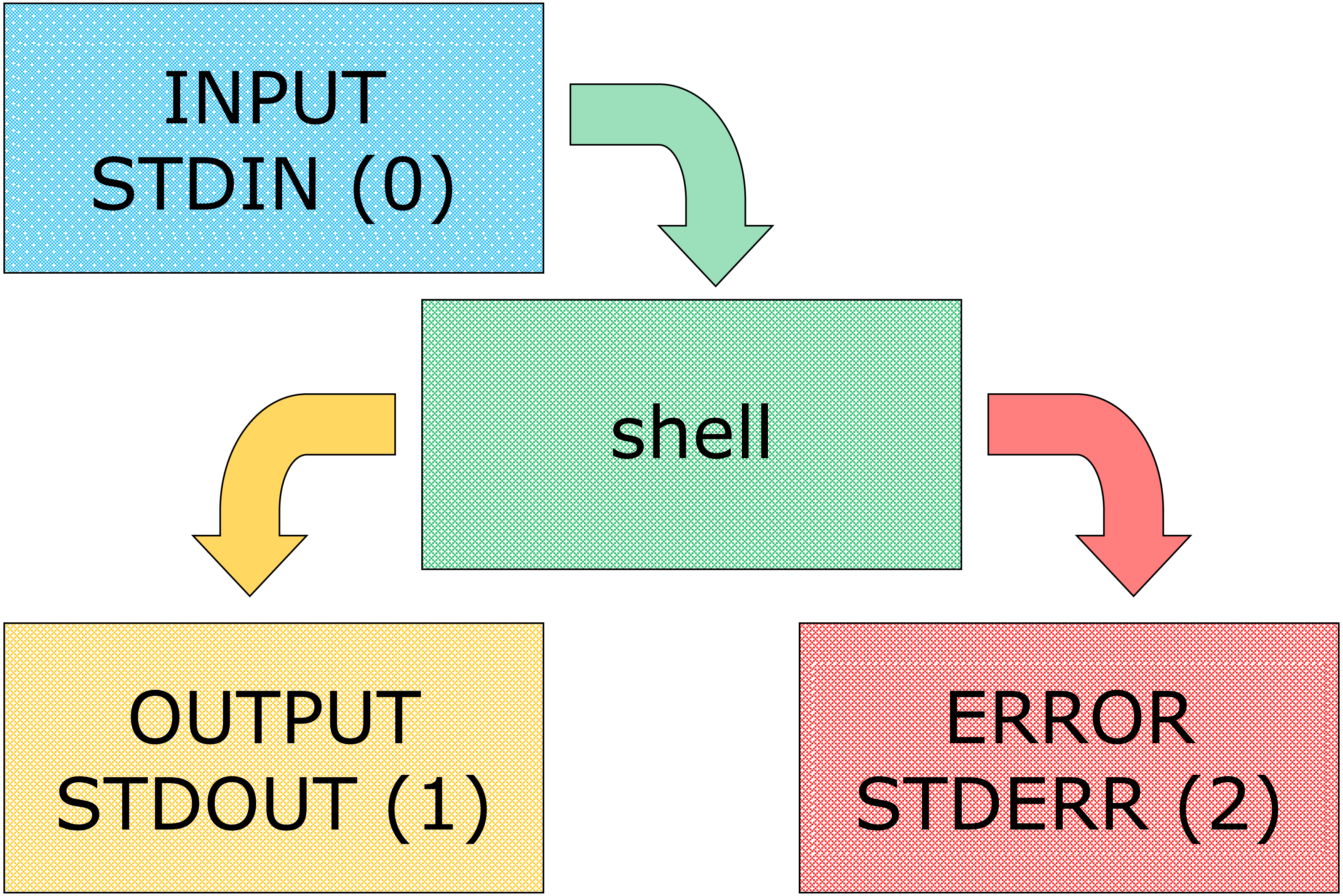
Lesson Objectives

In this lesson, you will learn:

- Input output redirection
- Pipies

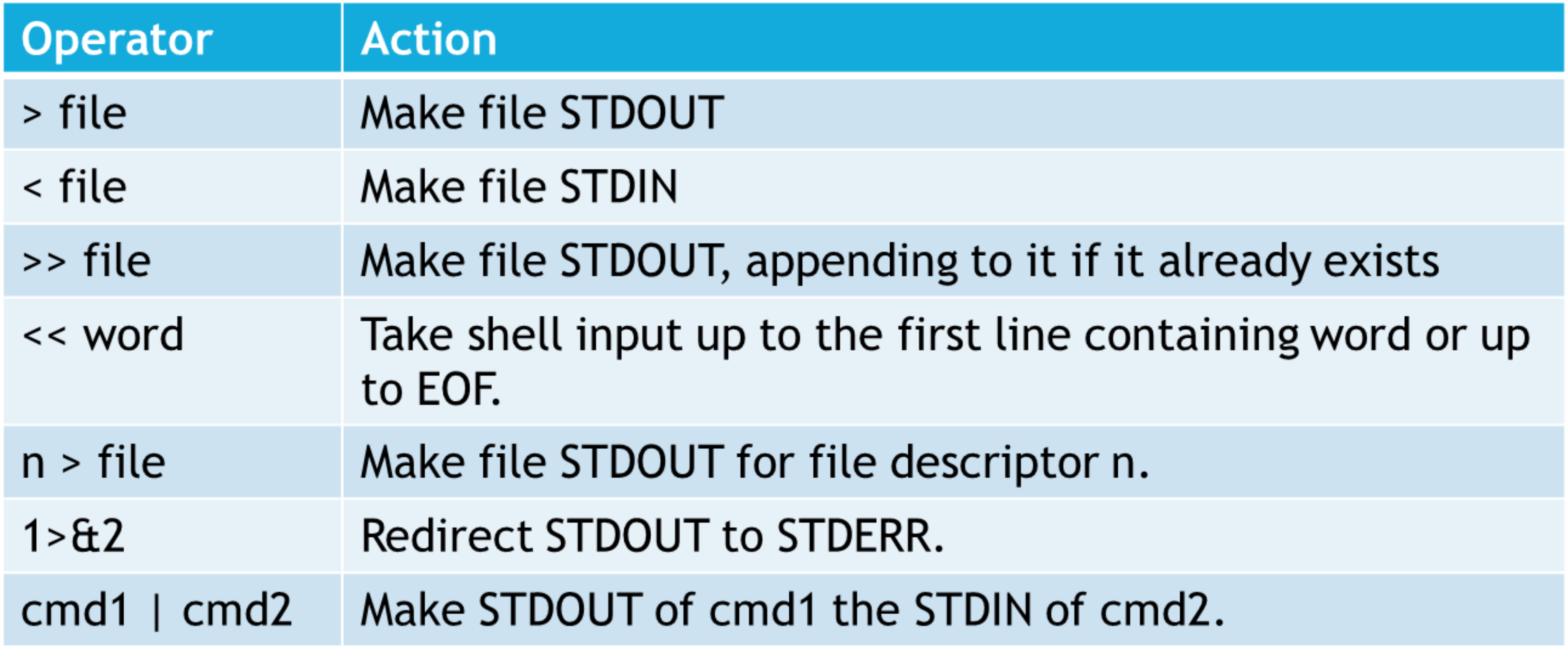
4.1: Redirection

Input Output Flow



Redirection

- Default STDIN is keyboard, for STDOUT & STDERR is screen.
- Unix allows us to change the STDIN and STDOUT by redirection.



Operator	Action
> file	Make file STDOUT
< file	Make file STDIN
>> file	Make file STDOUT, appending to it if it already exists
<< word	Take shell input up to the first line containing word or up to EOF.
n > file	Make file STDOUT for file descriptor n.
1>&2	Redirect STDOUT to STDERR.
cmd1 | cmd2	Make STDOUT of cmd1 the STDIN of cmd2.

Redirection examples

```
$ cat file1 > file2
```

file2 is STDOUT, the output of cat which normally sent to screen is now sent to file2, also it is created if not exist, if it exists then it is wiped clear and refilled with new data.

```
$ cat
Did you hear the story of optimist and the pessimist?
<CTRL D>
Did you hear the story of optimist and the pessimist?
```

```
$ cat < newfile
$ cat newfile
```

Both above command displays the same o/p, the subtle difference is in the latter newfile was not STDIN. Instead, the STDIN remained the terminal.

Redirection examples

```
$ cat newfile -
We all shall plant some trees.
<CTRL D>
Did you hear the story of optimist and the pessimist?
We all shall plant some trees.
```

cat concatenated the two inputs, prints filename first then the keyboard input, represented by the hyphen ‘-’

```
$ cat < currentfile > newfile
$ cat > newfile < currentfile
```

Input is taken from the currentfile and o/p is routed to newfile.

```
$ who >> logfile
```

Popular redirection operator called append, in the above command it appends current list of users who have logged in to the end of the file logfile.

Redirection examples

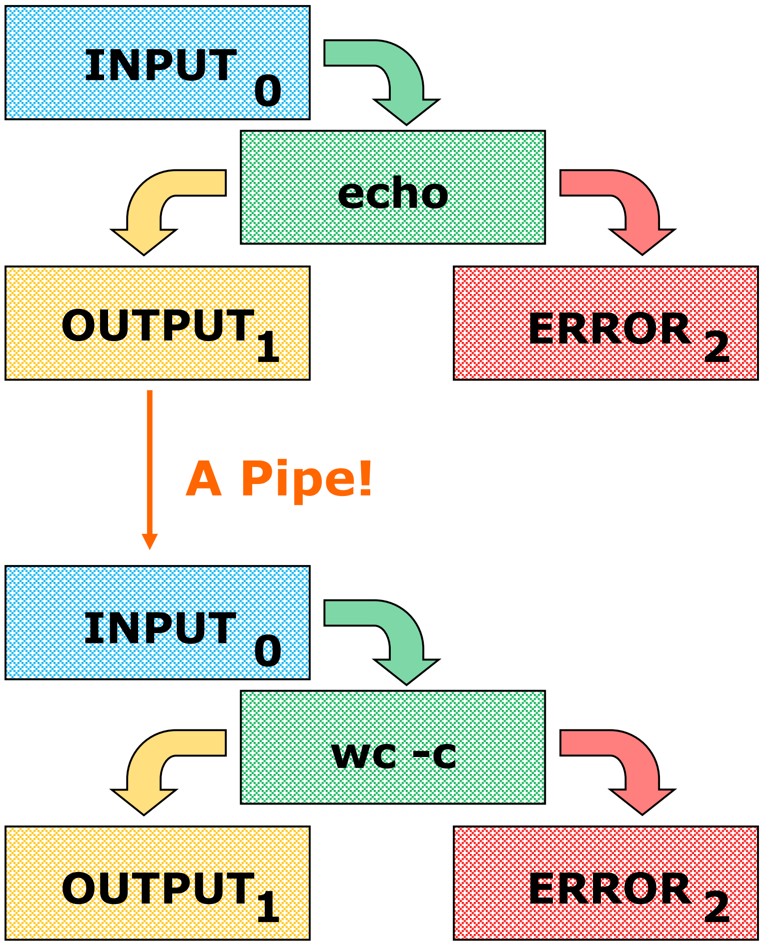
Don’t jump to conclusions. Try your hand on the below commands. You are in for a few surprises.

```
$ ls > filelist
$ Banner Hi-Fi > message
$ cat par.3 par.4 par.5 >> report
$ cat file1 > file1
$ cat < file1 > file1
$ cat > file <file1
$ date;who
$ date;who > logfile
$ (date;who) > logfile
```

4.2: Pipe
Piping

A single Unix command does not suffice to solve a problem or do a task.

```
$ echo “Hello” | wc -c
```



Piping

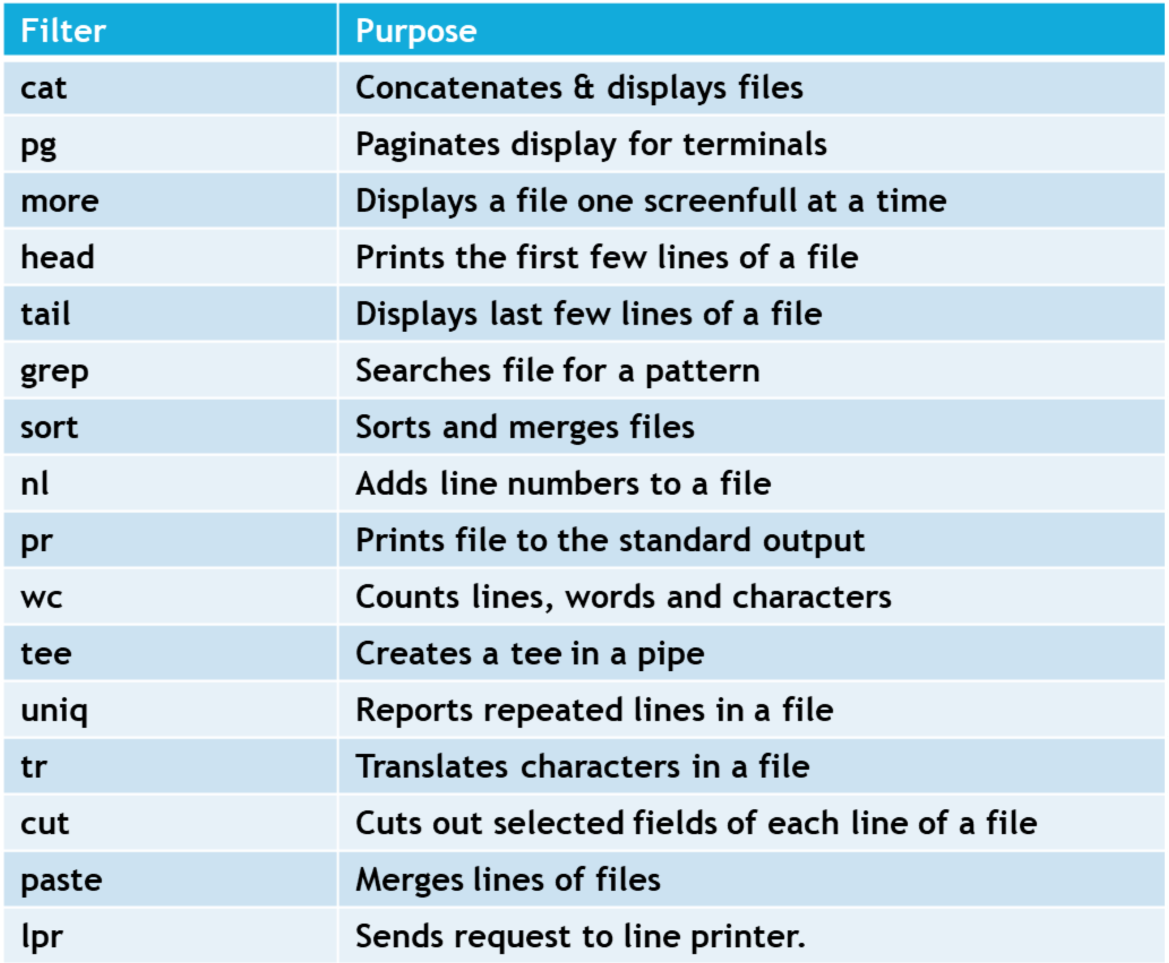
- **Redirection lets us connect commands to files.**

- **The UNIX piping facility lets us connect commands to other commands.**

- **It is possible to join commands using a pipe (|) only when command accept input from STDIN and send o/p to STDOUT. Incidentally, such commands are known as filters.**

- **man a command to know if it’s a filter.**

Filter commands



Filter	Purpose
cat	Concatenates & displays files
pg	Paginates display for terminals
more	Displays a file one screenfull at a time
head	Prints the first few lines of a file
tail	Displays last few lines of a file
grep	Searches file for a pattern
sort	Sorts and merges files
nl	Adds line numbers to a file
pr	Prints file to the standard output
wc	Counts lines, words and characters
tee	Creates a tee in a pipe
uniq	Reports repeated lines in a file
tr	Translates characters in a file
cut	Cuts out selected fields of each line of a file
paste	Merges lines of files
lpr	Sends request to line printer.

Piping examples

$ ls | wc –l => O/p of ls is input to wc which counts lines.

$ ls | wc –l > countfile => Instead of displaying, redirect to file.

$ who | sort => O/p of who is piped to sort, which sorts the data.

$ who | sort > sortedlist => Instead of displaying, redirect to file.

Piping examples

Situations where we can use pipe operator.

```
$ who | grep aravind
$ ls | grep file1
$ who | wc –l
$ ls | sort –r
$ ls –r | more
$ cat file1 file2 file3 | pg
$ cal 1996 | head -10
$ cal 1996 | tail -5
$ cat poetry | grep love
```

Piping examples

Input from a pipe can be combined with input from files.

The trick, as in combining redirection with files, is to use the special symbol – (a hyphen) for those commands that recognizes STDIN.

```
$ who | sort – logfile > newfile
```

O/p of who becomes STDIN to sort. Meanwhile sort opens the file logfile. The contents of this file are sorted together with the o/p of who, and the sorted o/p is redirected to the newfile.

Piping examples

- **Redirection routes o/p to files**
- **Piping routes o/p to programs**
- **What if you want to do both? Unix offers tee command to achieve this.**
- **tee – reads from STDIN and routes one copy to STDOUT and the other to a file.**

- **$ who | tee logfile | sort**
- **$ who | tee logfile newlogfile | sort**
- **$ who | tee file1 file2 /dev/tty3a | sort > file3**
- **$ cat file1 file2 | tee –a completefile | more**
- **$ cat myfile > newfile 2> errorfile**
- **$ cat myfile 1> newfile 2> errorfile**
- **$ cat errorfile   => If myfile did not exist**
- **cat: cannot open myfile**

SUMMARY

- In this lesson, you have learnt:
  - Redirection operators
  - How to use pipes

Review Questions

❖Question 1: 2> symbol is used as error redirection

- True / False

- **Question 2:What is the file descriptor number for > STDOUT operator?**
- **What is the use of pipe ?**