Special Characters

coursera.org/learn/linux-for-developers/supplement/oHIuj/special-characters

A number of characters have a special meaning and cause certain actions to take place. If you want to print them directly, you usually have to prefix them with a backslash (\) or enclose them in single quotes.

Redirection Special Characters

Character	Usage
\#>	Redirect output descriptor (Default # = **1, stdout**)
<	Redirect input descriptor
>>	Append output
>&	Redirect **stdout** and **stderr** (equivalent to **.. > .. 2>&1**)

Compound Commands Special Characters

Character	Usage
|	Piping
()	Execute in a separate shell
&&	AND list
||	OR list
;	Separate commands

Expansion Special Characters

Character	Usage
{}	Lists
~	Usually means **$HOME**
$	Parameter substitution
‘	Back tick; used in expression evaluation (also **$()** syntax)

Character	Usage
$(( ))	Arithmetic substitution
[]	Wildcard expressions, and conditionals

Escapes Special Characters

Character	Usage
\	End of line, escape sequence
’ ’	Take exactly as is
" "	Take as is, but do parameter expansion

Other Special Characters

Character	Usage
&	Redirection and putting task in background
#	Used for comments
***?**	Used in wildcard expansion
!	Used in history expansion

Note there are three different quoting mechanisms listed above:

- \ (as in \|; try **echo** | vs **echo** \|)
- single quotes: preserves literal value
- double quotes: same except for **$**, ‘, and \ .

Note you can get a literal quote character by using \’ or \".

Try:

```
$ echo $HOME

$ echo \$HOME

$ echo ’$HOME’

$ echo "$HOME"
```