Quoting Phrases

The metacharacters that have special meaning to the Bash shell can be used literally, without applying their special meaning, by enclosing them within a pair of ‘’ single-quote characters to form a quoted phrase. For example, to include the name of a shell variable in a phrase without interpreting its value:

1. At a prompt, type `echo Processed By: $SHELL` then hit Return to see the shell variable get interpreted in output.

2. Now, enter `echo 'Processed By: $SHELL'` to see the shell variable printed literally in output.

```
$ echo Processed By: $SHELL
Processed By: /bin/bash
$ echo 'Processed By: $SHELL'
Processed By: $SHELL
```

Alternatively, the significance of the leading $ metacharacter of a shell variable can be ignored if preceded by a \ backslash character to “escape” it from recognition as having special meaning:

3. At a prompt, type `echo Processed By: $SHELL` then hit Return to see the shell variable get interpreted in output.

4. Now, enter `echo Processed By: \$SHELL` to see the shell variable printed literally in output.

```
$ echo Processed By: $SHELL
Processed By: /bin/bash
$ echo Processed By: \$SHELL
Processed By: $SHELL
```

The newline \n and tab \t sequences can be included in phrases if preceded by a backslash – for example, `echo \nNEWLINE \tTAB`.

Always enclose phrases you want to use literally within single quotes to avoid misinterpretation.
...cont’d

It is necessary to precede a single-quote character with a \ backslash when it is used as an apostrophe, so it is not interpreted as an incomplete quoted phrase. An incomplete quoted phrase or a \ backslash at the end of a line allows a command to continue on the next line as they escape the newline when you hit Return:

5. At a prompt, enter `echo It’s escaped` to see the apostrophe appear in output

6. Next, type `echo Continued \` then hit Return, type `text written along ` then hit Return, and type `several lines` then hit Return to see the continued phrase in output

Double-quote marks “ ” are regarded as weak by the Bash shell as they do allow the interpretation of shell variables they enclose. They can, however, be useful to print out a quoted string if the entire string (and its double quotes) are enclosed in single quotes:

7. Type `echo “Interpreted With $SHELL”` then hit Return to see the shell variable get interpreted in unquoted output

8. Now, enter `echo ‘“Interpreted With $SHELL”’` to see the shell variable printed literally in quoted output

You could alternatively escape double-quote characters with a backslash to print them in output – for example, `echo \“With \$SHELL\“`. Notice that the shell prompt string changes to a > to indicate it is awaiting further input.