How Bash Processes Command Lines

Worked Example 3

In this Video…

- We are going to be throwing in some globbing and command substitution

Let’s Go!

Initial Command Line

```
echo “$(ls *.txt)”
```

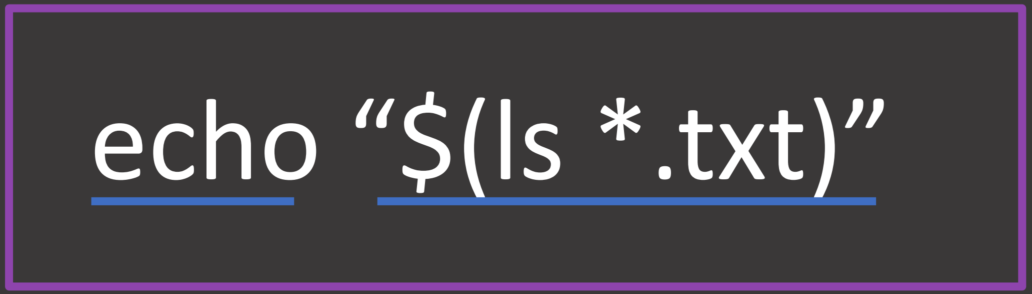
Step 1: Tokenisation – Identify Unquoted Metacharacters

```
echo “$(ls *.txt)”
```

Step 1: Tokenisation – Find Words and Operators

```
echo “$(ls *.txt)”
```

Step 2: Command Identification



```
echo “$(ls *.txt)”
```

Step 3: Expansions

```
echo “$(ls *.txt)”
```

Sub Shell
Initial Command Line

```
ls *.txt
```

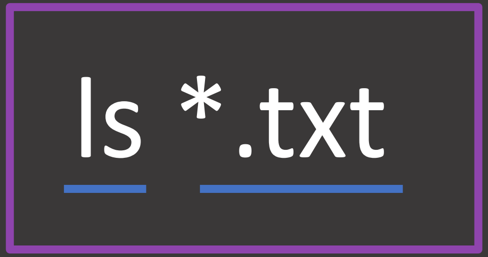
Sub Shell
Step 1: Tokenisation – Identify Metacharacters

```
ls *.txt
```

Sub Shell
Step 1: Tokenisation – Find Words and Operators

```
ls *.txt
```

Sub Shell
Step 2: Command Identification



```
ls *.txt
```

Sub Shell
Step 3: Expansions (Globbing)

```
ls a.txt b.txt
```

Sub Shell
Step 4: Quote Removal

```
ls a.txt b.txt
```

There is no Quote Removal

Sub Shell
Step 5: Redirection

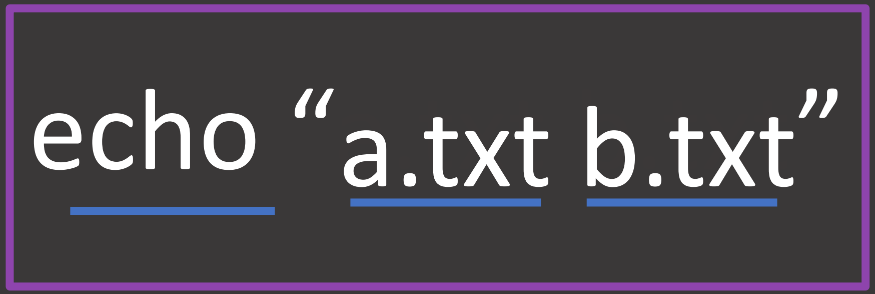
```
ls a.txt b.txt
```

There is no Redirection

Sub Shell Result

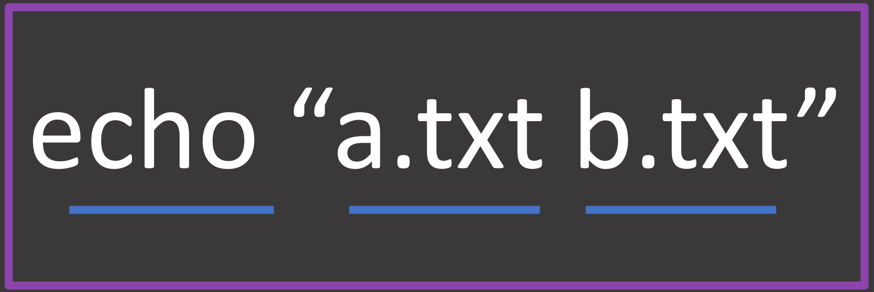
```
a.txt b.txt
```

Main Shell Step 3: Expansions (Stage 2 - After)



```
echo “a.txt b.txt”
```

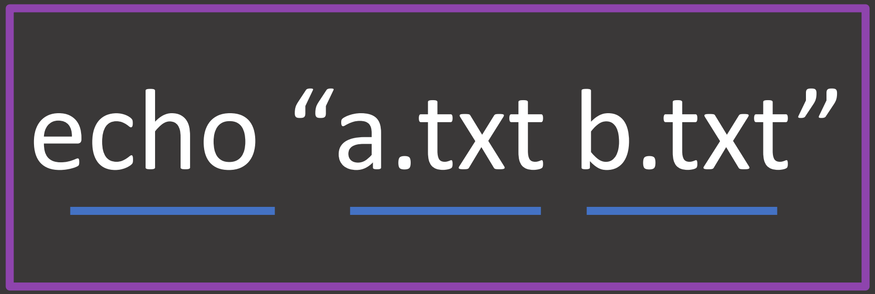
Main Shell Step 3: Word Splitting (Stage 3)



```
echo “a.txt b.txt”
```

There is no word splitting,
because the command substitution was performed
within double quotes

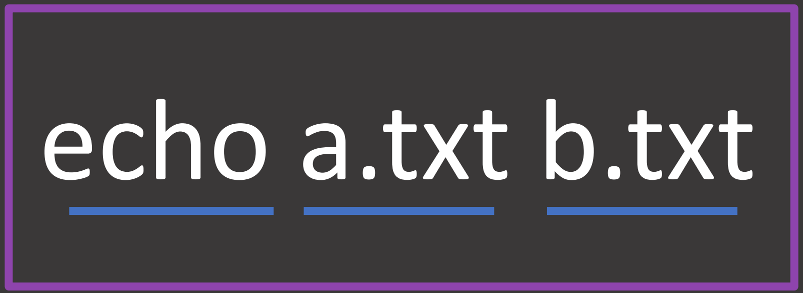
Main Shell Step 3: Globbing (Stage 4)



```
echo “a.txt b.txt”
```

There is no globbing

Main Shell Step 4: Quote Removal



```
echo a.txt b.txt
```

Main Shell Step 5: Redirections

```
echo a.txt b.txt
```

There are no Redirections

Up Next:

Problem Set
+ Section Summary