QUIZ

Saturday, 17 April 2021 6:31 PM

Question 1 of 6

If we ran a script with the following content by typing `posargs.sh one two three` what would it print on the screen?

```
#!/bin/bash
echo $0
```

You are correct!

./posargs.sh

Feedback

The first positional argument referenced by $0 holds the script name with the path needed to execute it. In this case we typed the name of the script with no path so it's in the current directory.

Question 2 of 6

What does the following code do?

```
while IFS= read -r LINE; do
       echo "$LINE"
done < "$1"
```

You are correct!

The redirect sends the file referenced by the $1 positional variable to the read command. The while command steps through the file one line at a time and echo prints that line on the screen.

Feedback

Correct, the read command reads the file and assigns each line to the LINE variable which the while command loops through.

Question 3 of 6

What is the difference between the `readarray` and the `mapfile` commands?

You are correct!

Nothing, they are the same.

Feedback

Correct, they are the same command.

Question 5 of 6

Review the script below. When reading data that has been piped into this script, why do you need a conditional?

```
if [[ -p /dev/stdin ]]; then
        while IFS= read -r LINE; do
                echo "Line: $LINE"
        done
fi
```

You are correct!

You need to make sure `/dev/stdin` is a pipe. If it is we'll read the data using the read command. If it is not then we don't want to read from it. For instance if it were a block device.

Question 6 of 6

What is the purpose of `:a in this code snippet?`

```
while getopts ":a" opt; do
  case $opt in
    a) echo "You passed the -a option" >&2 ;;
    \?) echo "Invalid option: -$opt" >&2 ;;
  esac
done
```

You are correct!

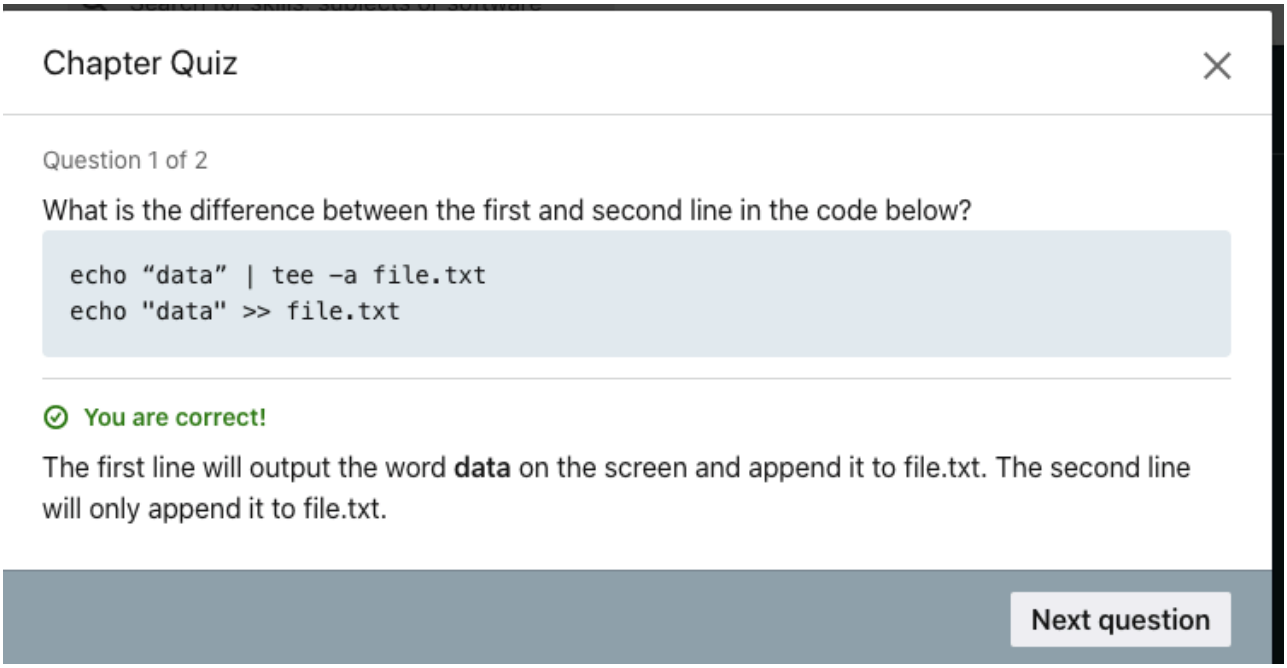
The `:a` options turn off verbose errors and "a" is the only valid option to the script.

Feedback

Correct, not providing the colon will result in displaying getopts error messages on the screen.

Next

CHAPTER 4:



CHAPTER 4 IF_ELSE:

Question 1 of 5

How does the following code snippet work?

```
if grep root /etc/passwd ;then
```

```
        echo "1"
else
        echo "0"
fi
```

You are correct!

The grep command searches the /etc/passwd file for the word "root". If it exists grep returns 0 to the if conditional and "1" will be displayed to the screen. If it is not found then grep returns a non-zero value to the if conditional and "0" is displayed

---

Question 2 of 5

What advantage does a `case` statement have over an `if` conditional?

You are correct!

If checking a condition against multiple values a case statement can be more efficient as it does the comparison one time. With if conditionals you may need an if conditional with successive elifs to accomplish the same goal.

Chapter Quiz

Question 3 of 5

Which is the preferred way of doing integer math in Bash?

You are correct!

((a=17+23))

Feedback

The double parenthesis method is the preferred way of doing integer math. This method also returns codes so can be used in an if conditional. Values can be returned to STDOUT by using the $(()) syntax such as a=$((17+23)).

Next question

What do you have to be aware of in the following conditional?

```
if [[ 4 > 4 ]]
```

You are correct!

This is a string comparison and will not evaluate 4 as an integer. The > symbol will be comparing ASCII values of the character 4, not the integer value.

Feedback

To compare integer equality in an if conditional you should use if [[4 -eq 4]]. The > operator is for string comparisons.

What is wrong with the following code snippet?

```
if ls /etc/passwd &> /dev/null
then
    echo "exists"
fi
```

`<`

This answer is incorrect.

The `ls` command will output the name of the file to the screen.

Correct answer:

The `if` conditional has test conditions for the existence of a file that are far more reliable.

Hint