Bash Practice Questions for Chapter 3: Passing Arguments to Bash Scripts

The third chapter of the Bash Beginner Series on Linux Handbook focuses on passing arguments to a shell script. These practice questions will help you practice what you learned in this chapter.

Exercise 1: What is this script?

Write a shell script in that prints the name of the script.

Difficulty level: Easy

Hint: Use $0 to get script name.

Exercise 2: Let me reverse that

Write a shell script that takes 3 arguments and prints them in the reverse order.

Difficulty level: Easy

Hint: Since if-else is not covered at this stage, no need to make a check on number of arguments. Only make sure to provide 3 arguments to the script.

Sample output:

```
./reverse.sh Ritchie Stallman Torvalds

Output:

Original Input - Ritchie Stallman Torvald

Reversed Output - Torvalds Stallman Ritchie
```

Exercise 3: Creating a directory and a file

Write a shell script that accepts a directory name called hello and creates it if it doesn't exist already. Switch to this directory and create a new file named new.txt.

Difficulty level: Easy

Hint: Use the -p option of the mkdir command

Exercise 4: Count number of lines and words

Create a text file named sample.txt and add the following text to it:

```
 This is a sample file.
A sample file contains sample text.
It can be easily counted using wc command.
```

Pass this sample.txt file as an argument to a script that counts the number of lines and words in the file.

Difficulty level: Intermediate

Hint: Use wc command to count number of lines and words. Use the cut command to remove the file name from the output of the wc command.

Exercise 5: Where is this command?

Write a shell script in that accepts a command as argument and gives the location of its binary file.

Difficulty level: Easy

Hint: You may use either whereis or which command

Solutions to the Exercises

Solution 1: What is this script?

> Write a shell script in that prints the name of the script.

```
#!/bin/bash

echo "This script is called: $0"
```

Solution 2: Let me reverse that

> Write a shell script that takes 3 arguments and prints them in the reverse order.

```
#!/bin/bash

echo "Original Input - $1 $2 $3"
echo "Reversed Output - $3 $2 $1"
```

Solution 3: Creating a directory and a file

> Write a shell script that accepts a directory name called hello and creates it if it doesn't exist already. Switch to this directory and create a new file named new.txt

```
#!/bin/bash

mkdir -p $1
cd $1
touch new.txt
```

You'll notice that even though the directory was not 'visibly' changed, the new.txt file is created inside it. How come?

If you run the script like `bash script.sh` or `./script.sh` , it won't change the directory. Why? Because by default, the scripts run in their own subshell. Even if the directory is changed (in the subshell), it won't be 'visible' to your current shell (which is the parent shell).

To run a shell script in the current shell, either use the source command like `source script.sh directory_name` or run it like this `. script.sh directory_name` .

Read more on subshell and login shell.

Solution 4: Count number of lines and words

> Pass the sample.txt file as an argument to a script that counts the number of lines and words in the file.

```
#!/bin/bash

lines=$(wc -l $1 | cut -d ' ' -f 1)
words=$(wc -w $1 | cut -d ' ' -f 1)
echo "Number of lines in $1 are $lines"
echo "Number of words in $1 are $words"
```

Solution 5: Where is this command?

> Write a shell script in that accepts a command as argument and gives the location of its binary file.

```
#!/bin/bash

location=$(which $1)
echo "Location of the binary of the command $1 is $location"
```