📚Linux Shell Script Interview questions and answers📚

Article by - Krishan Bhatt

Question1: What is the purpose of using shebang (#!) at the beginning of a shell script?

Answer: The shebang (#!) at the beginning of a shell script indicates the path to the shell interpreter that should be used to execute the script. For example, #!/bin/bash specifies that the Bash shell should be used. This ensures the script is executed using the correct shell environment.

Question 2: How do you define and use variables in shell scripting?

Answer: Variables in shell scripting are defined by assigning a value to a name. For example:

```
name="John"

echo "Hello, $name"
```

This would output “Hello, John” to the terminal.

Question 3: What is command substitution in shell scripting?

Answer: Command substitution allows the output of a command to replace the command itself. It is denoted by $(...) or backticks. For example:

```
current_date=$(date)
echo "Today's date is $current_date"
```

Question 4: How do you read user input in a shell script?

Answer: User input can be read using the read command. For example:

```
echo "Enter your name:"
read name
echo "Hello, $name"
```

Question 5: What are exit codes in shell scripting?

Answer: Exit codes are numeric values returned by a command to indicate its success or failure. Conventionally, 0 represents success and any non-zero value represents failure.

Question 6: How do you use conditional statements in shell scripting?

Answer: Conditional statements in shell scripting include if, elif, and else. For example:

```
if [ "$1" -eq 1 ]; then
   echo "Argument is 1"
else
   echo "Argument is not 1"
fi
```

Question 7: What is the purpose of a loop in shell scripting?

Answer: Loops in shell scripting, such as for and while loops, are used to execute a block of code repeatedly based on certain conditions or for a specified number of iterations.

Question 8: How do you comment a line in a shell script?

Answer: Lines in a shell script can be commented using the # symbol. For example:

```
# This is a comment
echo "This is not a comment"
```

Question 9: How do you pass arguments to a shell script?

Answer: Arguments are passed to a shell script as command-line arguments. They can be accessed within the script using positional parameters like $1, $2, etc.

Question 10: What is the difference between single and double quotes in shell scripting?

Answer: Single quotes preserve the literal value of each character within the quotes, while double quotes allow for variable expansion and command substitution.

Question 11: How do you check if a file exists in a shell script?

Answer: You can check if a file exists using the -e option with the test command or the [-e FILE] construct. For example:

```
if [ -e "$filename" ]; then
    echo "File exists"
else
    echo "File does not exist"
fi
```

Question 12: What is the purpose of the case statement in shell scripting?

Answer: The case statement is used for conditional branching based on the value of a variable. It allows for multiple possible conditions to be tested against.

Question 13: How do you iterate over the contents of a file in a shell script?

Answer: You can iterate over the contents of a file using a while loop combined with the read command. For example:

```
while IFS= read -r line; do
   echo "Line: $line"
done < "$filename"
```

Question 14: How do you redirect output in shell scripting?

Answer: Output redirection in shell scripting is achieved using the > symbol for standard output and the 2> symbol for standard error. For example:

```
echo "Hello, world!" > output.txt
```

Question 15: What is the purpose of the grep command in shell scripting?

Answer: The grep command is used to search for specific patterns or text within files. It is often used for text processing and filtering.

Question 16: How do you define functions in shell scripting?

Answer: Functions in shell scripting are defined using the function keyword followed by the function name and curly braces {} to enclose the function body. For example:

```
function greet {
   echo "Hello, $1"
}

greet "John"
```

Question 17: What is the purpose of the export command in shell scripting?

Answer: The export command is used to make variables available to child processes. It allows variables defined in the current shell to be accessed by subsequent commands and scripts.

Question 18: How do you check the permissions of a file in shell scripting?

Answer: You can check the permissions of a file using the ls -l command to list detailed file information, or using the stat command. Additionally, you can use conditional statements to check specific permissions.

Question 19: How do you terminate a shell script if an error occurs?

Answer: You can use the set -e option at the beginning of a shell script to terminate the script if any command returns a non-zero exit status. Additionally, you can use the trap command to catch errors and execute specific actions.

Question 20: What is the purpose of the shift command in shell scripting?

Answer: The shift command is used to shift positional parameters in a shell script. It moves each parameter one position to the left, effectively discarding the first parameter and making the next parameter become $1. This is often used in loops to iterate over variable-length argument lists.

Question 21: How do you concatenate strings in shell scripting?

Answer: Strings can be concatenated using the concatenation operator + or by simply placing them adjacent to each other. For example:

```
first_name="John"
last_name="Doe"
full_name="$first_name $last_name"
echo "Full name: $full_name"
```

Question 22: What is the purpose of the dirname command in shell scripting?

Answer: The dirname command is used to extract the directory portion of a file path. It returns the directory name without the filename itself.

Question 23: How do you perform arithmetic operations in shell scripting?

Answer: Arithmetic operations in shell scripting are performed using the expr command or using double parentheses ((...)). For example:

```
result=$((2 + 3))
```

```
echo "Result: $result"
```

Question 24: What is the purpose of the set command in shell scripting?

Answer: The set command is used to modify shell options and positional parameters. It can be used to enable or disable certain options and change the values of positional parameters.

Question 25: How do you check if a directory exists in shell scripting?

Answer: You can check if a directory exists using the -d option with the test command or the [-d DIRECTORY] construct. For example:

```
if [ -d "$directory" ]; then
    echo "Directory exists"
else
    echo "Directory does not exist"
fi
```

Question 26: What is the purpose of the cut command in shell scripting?

Answer: The cut command is used to extract sections from each line of files or standard input. It is often used to extract specific columns or fields from text files.

Question 27: How do you find and replace text in a file using shell scripting?

Answer: Text can be found and replaced using tools like sed (stream editor) or awk. For example:

```
sed -i 's/old_text/new_text/g' filename
```

Question 28: What is the purpose of the until loop in shell scripting?

Answer: The until loop is similar to the while loop, but it continues executing a block of code until a specified condition becomes true.

Question 29: How do you capture the output of a command in a variable in shell scripting?

Answer: The output of a command can be captured in a variable using command substitution with $(...) or backticks. For example:

```
current_date=$(date)
echo "Today's date is $current_date"
```

Question 30: What is the purpose of the tee command in shell scripting?

Answer: The tee command reads from standard input and writes to standard output and files simultaneously. It is often used to display and log output from commands.

Question 31: How do you check if a string contains a substring in shell scripting?

Answer: You can use the grep command with the -q option to check if a string contains a substring. For example:

```
if echo "$string" | grep -q "substring"; then
    echo "Substring found"
```

```
else
    echo "Substring not found"
fi
```

Question 32: What is the purpose of the printf command in shell scripting?

Answer: The printf command is used to format and print data. It provides more control over the output format compared to echo.

Question 33: How do you split a string into an array in shell scripting?

Answer: You can split a string into an array using the IFS (Internal Field Separator) variable and the read command. For example:

```
string="one two three"
IFS=' ' read -r -a array <<< "$string"
```

Question 34: What is the purpose of the basename command in shell scripting?

Answer: The basename command is used to extract the filename from a given path. It returns only the last component of the path.

Question 35: How do you check if a string is empty in shell scripting?

Answer: You can check if a string is empty using the -z option with the test command or the [-z STRING] construct. For example:

```
if [ -z "$string" ]; then
  echo "String is empty"
else
  echo "String is not empty"
fi
```

Question 36: What is the purpose of the pwd command in shell scripting?

Answer: The pwd command is used to print the current working directory.

Question 37: How do you remove leading and trailing whitespace from a string in shell scripting?

Answer: You can use parameter expansion with ## and %% to remove leading and trailing whitespace, respectively. For example:

```
trimmed_string="${string##*( )}"
trimmed_string="${trimmed_string%%*( )}"
```

Question 38: What is the purpose of the trap command in shell scripting?

Answer: The trap command is used to catch signals and execute specified commands when they occur. It is often used to handle cleanup tasks or error handling.

Question 39: How do you compare strings in shell scripting?

Answer: Strings can be compared using operators like =, !=, <, >, -z (empty), and -n (not empty) with the test command or the [ ] construct.

Question 40: What is the purpose of the dirname command in shell scripting?

Answer: The dirname command extracts the directory component of a file path, excluding the filename itself.

Question 41: How do you find the number of arguments passed to a shell script?

Answer: The number of arguments passed to a shell script can be obtained using the $# variable. For example:

```
echo "Number of arguments: $#"
```

Question 42: What is the purpose of the grep command in shell scripting?

Answer: The grep command is used to search for patterns or specific text within files. It can also be used in conjunction with pipes to filter output.

Question 43: How do you check if a command exists in shell scripting?

Answer: You can check if a command exists by using the command -v or type command. For example:

```
if command -v ls >/dev/null 2>&1; then
  echo "ls command exists"
else
  echo "ls command does not exist"
fi
```

Question 44: What is the purpose of the awk command in shell scripting?

Answer: The awk command is a powerful text processing tool used for pattern scanning and processing. It allows for data extraction and reporting based on specified patterns.

Question 45: How do you execute a command in the background in shell scripting?

Answer: You can execute a command in the background by appending & at the end of the command. For example:

```
sleep 10 &
```

Question 46: What is the purpose of the dirname command in shell scripting?

Answer: The dirname command extracts the directory component of a file path, excluding the filename itself.

Question 47: How do you check if a file is readable, writable, or executable in shell scripting?

Answer: You can use the -r, -w, and -x options with the test command or the [-r FILE], [-w FILE], and [-x FILE] constructs to check file permissions. For example:

```
if [ -r "$filename" ]; then
  echo "File is readable"
fi
```

Question 48: How do you find the length of a string in shell scripting?

Answer: You can find the length of a string using the ${#string} syntax. For example:

```
string="Hello, world!"
echo "Length of string: ${#string}"
```

Question 49: What is the purpose of the sort command in shell scripting?

Answer: The sort command is used to sort lines of text files alphabetically or numerically. It can also be used to merge and compare sorted files.

Question 50: How do you check if a directory is empty in shell scripting?

Answer: You can check if a directory is empty by listing its contents and checking if the output is empty. For example:

```
if [ -z "$(ls -A $directory)" ]; then
  echo "Directory is empty"
else
  echo "Directory is not empty"
```

fi