

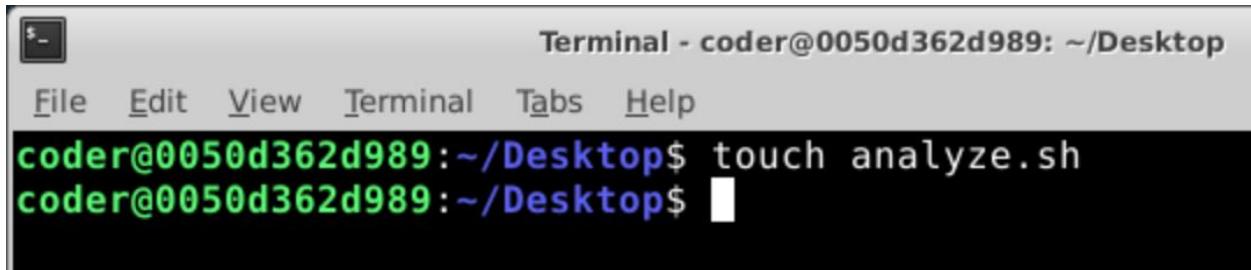
Question 1 [6 Points]

Create a shell script named analyze.sh that accepts exactly ONE command-line argument.

- If the argument is a file: – Display the number of lines, words, and characters in the file.
- If the argument is a directory: – Display the total number of files present. – Display the number of .txt files in the directory.
- If the argument count is invalid or the path does not exist: – Display an appropriate error message.

Answer:

- 1) Creating a shell script file named " analyze.sh " using the touch cmd.



The screenshot shows a terminal window titled "Terminal - coder@0050d362d989: ~/Desktop". The window has a menu bar with "File", "Edit", "View", "Terminal", "Tabs", and "Help". The main area of the terminal shows the command "touch analyze.sh" being typed by the user "coder@0050d362d989". The command is highlighted in green, indicating it is being processed. The prompt "coder@0050d362d989:~/Desktop\$" is visible at the bottom of the terminal window.

2) I used "nano analyze.sh" to open the file in editor.

3)

```
Question1 > $ analyze.sh

1  #!/bin/bash
2
3  # Check if exactly one argument is provided
4  if [ $# -ne 1 ]; then
5      echo "Error: Please provide exactly one argument."
6      exit 1
7  fi
8
9  # Store the argument
10 path="$1"
11
12 # Check if the path exists
13 if [ ! -e "$path" ]; then
14     echo "Error: The specified path does not exist."
15     exit 1
16 fi
17
18 # If the argument is a file
19 if [ -f "$path" ]; then
20     echo "File analysis:"
21     wc "$path"
22
23 # If the argument is a directory
24 elif [ -d "$path" ]; then
25     echo "Directory analysis:"
26
27     # Total number of files
28     total_files=$(find "$path" -type f | wc -l)
29     echo "Total number of files: $total_files"
30
31     # Number of .txt files
32     txt_files=$(find "$path" -type f -name "*.txt" | wc -l)
33     echo "Number of .txt files: $txt_files"
34
35 # If it's neither file nor directory
36 else
37     echo "Error: The path is neither a file nor a directory."
38     exit 1
39 fi
40
```

```
File Edit View Terminal Tabs Help
GNU nano 2.9.3                                         analyze.

lab@...:~/bin/bash

#Check if exactly one argument is provided
if [ $# -ne 1]; then
    echo "Error: Please provide exactly one argument."
    exit 1
fi

#Store the argument
path="$1"

#check if the path exists
if [ ! -e "$path" ]; then
    echo "Error: The specified path does not exist "
    exit 1
fi

#If the argument is a file
if [ -f "$path" ]; then
    echo "File analysis:"
    wc "$path"

#If the argument is a directory
elif [ -d "$path" ]; then
    echo "Directory analysis:"
```

#Total number of files

```
    total_files=$(find "$path" -type f | wc -l)
    echo "Total number of files: $total_files"
```

#Number of .txt files

```
    txt_files=$(find "$path" -type f -name "*.txt" | wc -l)
    echo "Number of .txt files: $txt_files"
```

#If it's neither file nor directory

```
else
    echo "Error: The path is neither a file nor a directory."
    exit 1
fi
```

Terminal Output:

```
sudhanshu@Sudhanshus-MacVati Question1 % ls
analyze.sh      Question1.docx
sudhanshu@Sudhanshus-MacVati Question1 % ./analyze.sh analyze.sh
File analysis:
      39      149     869 analyze.sh
sudhanshu@Sudhanshus-MacVati Question1 % ./analyze.sh .
Directory analysis:
Total number of files:      2
Number of .txt files:      0
sudhanshu@Sudhanshus-MacVati Question1 % ./analyze.sh does_not_exist
Error: The specified path does not exist.
sudhanshu@Sudhanshus-MacVati Question1 % ./analyze.sh a b
Error: Please provide exactly one argument.
sudhanshu@Sudhanshus-MacVati Question1 %
```

Explanation:

`#!/bin/bash` - This is telling linux, when someone runs this file, use bash to execute it. Without this line Linux may not know how to interpret the script.

`#` - Used to put out any comments in the script for explanation.

`if [$# -ne 1]; then` : This starts the conditional statement

`-ne` : "not equal"

`-lt`: "less than"

`-gt` : "greater than"

`exit 1`: Exit the script immediately

`fi` : means end the if block

`wc "$path"` : It prints the following things

1. Number of lines
2. Number of words
3. Number of characters
4. File name

`total_files=$(find "$path" -type f | wc -l)`

-type f : Excludes directories, Only regular files

| : Pipe used to send output of left cmd to right cmd

wc -l : Counts the number of lines