Placement Empowerment Program

Cloud Computing and DevOps Centre

Day 14 – File Word/Line/Character Counter Script

Count the number of lines, words, and characters in all .txt files within a directory and generate a summary report.

Name: Hema S

Department: ECE

St. JOSEPH'S COLLEGE OF ENGINEERING | St. JOSEPH'S INSTITUTE OF TECHNOLOGY

AUTONOMOUS INSTITUTIONS, AFFILIATED TO ANNA UNIVERSITY

Introduction

In Linux systems, working with text files is a daily task for developers, system administrators, and data analysts. Whether you're auditing logs, analyzing content, or generating reports, it's often useful to know how many lines, words, and characters a file contains. Doing this manually for multiple files can be time-consuming.

This Proof of Concept (PoC) demonstrates how to automate the process using a **shell script** that scans all .txt files in a directory and generates a **summary report** containing:

Line count
Word count
Character count

for each file.
Using Linux commands like **wc, awk, and loops**, this script is a simple but powerful example of how shell scripting can be used for batch file analysis and reporting.

This PoC enhances your understanding of **file handling, text processing, and shell scripting automation.**

Overview

This PoC focuses on creating a shell script that automates the process of counting the number of lines, words, and characters in all .txt files within a specified directory.
The script uses the Linux wc (word count) command and loops through each .txt file. It collects file-wise statistics and generates a structured summary report in a log file named file_summary_report.log.

Tools and Commands Used

wc – for word, line, and character count
for loop – to iterate over files
awk – to extract specific count values
File redirection (>, >>) – to write to a report file

This script is useful for text processing tasks such as:

- Codebase analysis
- Log file audits
- Document size monitoring

Objectives :

✅ Automate File Analysis
Automatically count the number of lines, words, and characters in all .txt files within a directory.

✅ Generate a Structured Report
Collect file statistics and write them to a summary log file (file_summary_report.log) in a clear and readable format.

✅ Practice Core Linux Commands
Strengthen knowledge of important shell commands:

- wc for word counting
- awk for field extraction
- Shell loops, conditionals, and file redirection

✅ Improve Text Processing Skills
Learn how to:

- Traverse files
- Extract specific information
- Handle batch text operations in Linux

Importance :

✅ Automates Repetitive File Analysis
Manually counting lines, words, or characters in multiple files is slow and error-prone. This script automates the process in seconds.

✅ Builds Real-World Shell Scripting Skills
This task improves your understanding of:

1.wc, awk, for, if, redirection operators
2.How to write scripts that **scan**, **process**, and **report**
These are critical for system automation, scripting interviews, and DevOps workflows.

✅ **Prepares You for Log and Code Auditing**
Counting lines and words is useful in:

1.Log analysis (e.g., check size of logs)
2.Codebase audits (e.g., measure file complexity)
3.Report generation from raw text data

✅ **Enhances Text Processing Confidence**
Working with files in batch teaches you how to:

1.Loop through patterns (*.txt)
2.Filter and transform text using powerful tools
3.Summarize useful insights from data

✅ **Introduces Basic Scripting Best Practices**

1.Creating logs
2.Formatting reports
3.Writing reusable shell utilities

Step-by-Step Overview

Step 1:Open Terminal

Launch a terminal window on your Linux system.

Step 2: Create Sample .txt Files

```
hemas@Hema:~$ echo "hello world" > sample.txt
hemas@Hema:~$ echo "linux scripting is powerful" > notes.txt
```

Check that they exist:

```
hemas@Hema:~$ ls *.txt
notes.txt  sample.txt
```

Step 3: Create a New Shell Script

```
hemas@Hema:~$ nano file_counter.sh
```

Step 4: Paste the Script into nano

Copy and paste the following:

```
hemas@Hema: ~
  GNU nano 7.2                                  file_counter.sh
# Output log file
REPORT="file_summary_report.log"

# Create or clear the report
echo "Summary Report - $(date)" > "$REPORT"
echo "Filename | Lines | Words | Characters" >> "$REPORT"
echo "-----------------------------------" >> "$REPORT"

# Loop through all .txt files
for file in *.txt; do
    echo "Checking file: $file"  # <-- add this line
    if [[ -f "$file" ]]; then
        stats=$(wc "$file")
        lines=$(echo $stats | awk '{print $1}')
        words=$(echo $stats | awk '{print $2}')
        chars=$(echo $stats | awk '{print $3}')
        echo "$file | $lines | $words | $chars" >> "$REPORT"
    fi
done

echo "Report generated in $REPORT"
```

Step 5: Save and Exit

Press Ctrl + O → Enter to save
Press Ctrl + X to exit

Step 6: Make the Script Executable

Back in the terminal:

```
hemas@Hema:~$ chmod +x file_counter.sh
```

This gives the script permission to run as a program.

Step 7: Run the Script

```
hemas@Hema:~$ ./file_counter.sh
Checking file: notes.txt
Checking file: sample.txt
Report generated in file_summary_report.log
```

Step 8: View the Report

```
hemas@Hema:~$ cat file_summary_report.log
Summary Report - Fri Jun 27 07:05:42 UTC 2025
Filename | Lines | Words | Characters
-----------------------------------
notes.txt | 1 | 4 | 28
sample.txt | 1 | 2 | 12
```

Outcomes:

✅Created a shell script to count **lines, words, and characters in .txt files.**

✅Learned to use essential Linux commands: **wc, awk**, **for**, and **if.**

✅Generated a structured **summary report** in a .log file.

✅Practiced **text replacement** using nano with **Ctrl + \.**

✅Strengthened file processing and **automation skills** in Linux scripting.