21CS8056
Sayan Paul

Compiler Design Lab-3

1. write a script to read a system log file (/var/log/syslog), extract lines containing "error" (case-insensitive), and use regular expressions to extract timestamps and error messages.

Code:

```
#!/bin/bash

log_file="/var/log/syslog"

grep -i "error" "$log_file" | while read -r line; do

   timestamp=$(echo "$line" | grep -oP "^\w{3} \d{1,2} \d{2}:\d{2}:\d{2}")

   error_message=$(echo "$line" | sed -n -e 's/^.*error: \(.*\)$/\1/p')

   echo "Timestamp: $timestamp, Error: $error_message"

done
```

Executing command & Output:

```
student@nitcse-OptiPlex-7000:~/Desktop/sayan56/cdlab/bash$ ll
total 12
drwxrwxr-x 2 student student 4096 Jan 24 14:33 ./
drwxrwxr-x 3 student student 4096 Jan 24 14:32 ../
-rw-rw-r-- 1 student student  310 Jan 24 14:33 slog.sh
student@nitcse-OptiPlex-7000:~/Desktop/sayan56/cdlab/bash$ chmod +x slog.sh
nitcse@nitcse-OptiPlex-7000:/home/student/Desktop/sayan56/cdlab/bash$ sudo ./slog.sh
[sudo] password for nitcse:
Timestamp: Jan 19 09:03:18, Error:
Timestamp: Jan 19 09:03:18, Error:
Timestamp: Jan 19 09:03:18, Error:
Timestamp: Jan 19 09:03:18, Error:
Timestamp: Jan 19 09:03:34, Error:
Timestamp: Jan 19 09:03:34, Error:
Timestamp: Jan 19 09:03:34, Error:
```

```
Timestamp: Jan 19 09:03:36, Error:
Timestamp: Jan 19 09:03:37, Error:
Timestamp: Jan 19 09:03:55, Error:
Timestamp: Jan 19 09:03:55, Error:
Timestamp: Jan 19 09:04:01, Error: database disk image is malformed (errno: Success)
Timestamp: Jan 19 09:04:02, Error: database disk image is malformed (errno: Success)]
Timestamp: Jan 23 14:05:42, Error:
Timestamp: Jan 23 14:05:42, Error:
Timestamp: Jan 23 14:05:42, Error:
Timestamp: Jan 23 14:05:45, Error:
Timestamp: Jan 23 14:05:45, Error:
Timestamp: Jan 23 14:05:49, Error:
Timestamp: Jan 23 14:05:49, Error:
Timestamp: Jan 23 14:05:49, Error:
Timestamp: Jan 23 14:05:58, Error:
Timestamp: Jan 23 14:05:58, Error:
Timestamp: Jan 23 14:05:58, Error:
Timestamp: Jan 23 14:05:58, Error:
Timestamp: Jan 23 14:06:14, Error:
Timestamp: Jan 23 14:06:14, Error:
Timestamp: Jan 23 14:06:14, Error:
Timestamp: Jan 23 14:06:14, Error:
Timestamp: Jan 23 14:06:15, Error:
Timestamp: Jan 23 14:06:16, Error:
Timestamp: Jan 23 14:06:50, Error:
Timestamp: Jan 23 14:06:50, Error:
Timestamp: Jan 23 14:06:52, Error:
Timestamp: Jan 23 14:06:52, Error:
Timestamp: Jan 23 14:06:54, Error:
Timestamp: Jan 23 14:06:54, Error:
Timestamp: Jan 23 14:06:54, Error:
Timestamp: Jan 23 14:06:54, Error:
Timestamp: Jan 23 14:06:54, Error:
Timestamp: Jan 23 14:11:02, Error: too many requests
Timestamp: Jan 23 15:19:18, Error:
Timestamp: Jan 23 15:19:18, Error:
Timestamp: Jan 23 15:19:18, Error:
Timestamp: Jan 23 15:19:21, Error:
Timestamp: Jan 23 15:19:21, Error:
Timestamp: Jan 23 15:19:26, Error:
Timestamp: Jan 23 15:19:26, Error:
Timestamp: Jan 23 15:19:26, Error:
```

```
Timestamp: Jan 23 15:19:35, Error:
Timestamp: Jan 23 15:19:35, Error:
Timestamp: Jan 23 15:19:35, Error:
Timestamp: Jan 23 15:19:35, Error:
Timestamp: Jan 23 15:19:54, Error:
Timestamp: Jan 23 15:19:54, Error:
Timestamp: Jan 23 15:19:54, Error:
Timestamp: Jan 23 15:19:56, Error:
Timestamp: Jan 23 15:19:57, Error:
Timestamp: Jan 23 15:20:08, Error:
Timestamp: Jan 23 15:20:08, Error:
Timestamp: Jan 23 15:20:11, Error:
Timestamp: Jan 23 15:20:11, Error:
Timestamp: Jan 23 15:20:11, Error:
Timestamp: Jan 23 15:20:11, Error:
Timestamp: Jan 23 15:29:58, Error:
Timestamp: Jan 23 16:16:27, Error:
Timestamp: Jan 23 17:02:02, Error:
Timestamp: Jan 23 17:02:04, Error:
Timestamp: Jan 24 14:00:19, Error:
Timestamp: Jan 24 14:00:19, Error:
Timestamp: Jan 24 14:00:19, Error:
Timestamp: Jan 24 14:00:24, Error:
Timestamp: Jan 24 14:00:24, Error:
Timestamp: Jan 24 14:00:28, Error:
Timestamp: Jan 24 14:00:28, Error:
Timestamp: Jan 24 14:00:28, Error:
Timestamp: Jan 24 14:00:36, Error:
Timestamp: Jan 24 14:00:36, Error:
Timestamp: Jan 24 14:00:36, Error:
Timestamp: Jan 24 14:00:36, Error:
Timestamp: Jan 24 14:00:54, Error:
Timestamp: Jan 24 14:00:54, Error:
Timestamp: Jan 24 14:00:54, Error:
Timestamp: Jan 24 14:00:56, Error:
Timestamp: Jan 24 14:00:57, Error:
Timestamp: Jan 24 14:01:04, Error:
Timestamp: Jan 24 14:01:04, Error:
Timestamp: Jan 24 14:01:07, Error:
Timestamp: Jan 24 14:01:07, Error:
Timestamp: Jan 24 14:03:37, Error: warning: queue 7f913ba6bdf0 destroyed while proxies
still attached:
```

```
Timestamp: Jan 24 14:03:37, Error:   wl_display@1 still attached
Timestamp: Jan 24 14:30:44, Error: decode new commands catalog: net/http: request
canceled (Client.Timeout or context cancellation while reading body)
```

2. Write a script using ps aux and grep to obtain information about the specified process and use regular expressions to extract relevant details.

Code:

```
#!/bin/bash



process_name="your_process_name"



ps aux | grep "$process_name" | while read -r line; do

   # Extract relevant details using regular expressions

   pid=$(echo "$line" | awk '{print $2}')

   user=$(echo "$line" | awk '{print $1}')

   cpu=$(echo "$line" | awk '{print $3}')

   mem=$(echo "$line" | awk '{print $4}')

   command=$(echo "$line" | awk '{$1=$2=$3=$4=""; print $0}')



   echo "PID: $pid, User: $user, CPU: $cpu, Memory: $mem, Command: $command"

done
```

Executing command & Output:

```
nitcse@nitcse-OptiPlex-7000:/home/student/Desktop/sayan56/cdlab/bash$ nano problem2.sh
nitcse@nitcse-OptiPlex-7000:/home/student/Desktop/sayan56/cdlab/bash$ ll
```

```
total 12
drwxrwxr-x 2 student student 4096 Jan 24 14:42 ./
drwxrwxr-x 3 student student 4096 Jan 24 14:32 ../
-rw-r--r-- 1 root    root       0 Jan 24 14:42 problem2.sh
-rwxrwxr-x 1 student student  310 Jan 24 14:33 slog.sh*
nitcse@nitcse-OptiPlex-7000:/home/student/Desktop/sayan56/cdlab/bash$ chmod 777
problem2.sh
chmod: changing permissions of 'problem2.sh': Operation not permitted
nitcse@nitcse-OptiPlex-7000:/home/student/Desktop/sayan56/cdlab/bash$ sudo chmod 777
problem2.sh
nitcse@nitcse-OptiPlex-7000:/home/student/Desktop/sayan56/cdlab/bash$ ll
total 12
drwxrwxr-x 2 student student 4096 Jan 24 14:42 ./
drwxrwxr-x 3 student student 4096 Jan 24 14:32 ../
-rwxrwxrwx 1 root    root       0 Jan 24 14:42 problem2.sh*
-rwxrwxr-x 1 student student  310 Jan 24 14:33 slog.sh*
nitcse@nitcse-OptiPlex-7000:/home/student/Desktop/sayan56/cdlab/bash$ nano problem2.sh
PID: 5671, User: nitcse, CPU: 0.0, Memory: 0.0, Command:     9212 2432 pts/0 S+ 14:45
0:00 grep mysql
```

3. Write a script that uses 'ifconfig' to get network interface information, 'grep' to filter relevant lines, 'sed' to extract interface names, and 'awk' to extract and print IP addresses for each interface.

Code:

```
#!/bin/bash

# Use ip command to get network interface information
ip_output=$(ip address show)

# Use grep to filter relevant lines
relevant_lines=$(echo "$ip_output" | grep -E "^[0-9]+: [a-zA-Z0-9]+:")

# Use awk to extract and print interface names and IP addresses
echo "$relevant_lines" | awk '{print "Interface:", $2, "IP Address:", $2}' FS=': '
```

Executing command & Output:

```
nitcse@nitcse-OptiPlex-7000:/home/student/Desktop/sayan56/cdlab/bash$ ll
```

```
total 16
drwxrwxr-x 2 student student 4096 Jan 24 14:42 ./
drwxrwxr-x 3 student student 4096 Jan 24 14:32 ../
-rwxrwxrwx 1 root    root     482 Jan 24 14:45 problem2.sh*
-rwxrwxr-x 1 student student  310 Jan 24 14:33 slog.sh*
nitcse@nitcse-OptiPlex-7000:/home/student/Desktop/sayan56/cdlab/bash$ touch netinfo.sh
touch: cannot touch 'netinfo.sh': Permission denied
nitcse@nitcse-OptiPlex-7000:/home/student/Desktop/sayan56/cdlab/bash$ sudo touch
netinfo.sh
nitcse@nitcse-OptiPlex-7000:/home/student/Desktop/sayan56/cdlab/bash$ sudo chmod 777
netinfo.sh
```

4. Write a script that uses 'du' to get disk usage information, 'sort' to sort the output by size in reverse order, and 'awk' to print the top five largest directories.

Code:

```
#!/bin/bash
du -h --max-depth=1 | sort -rh | head -n
```

Executing command & Output:

```
nitcse@nitcse-OptiPlex-7000:/home/student/Desktop/sayan56/cdlab/bash$ ll
total 24
drwxrwxr-x 2 student student 4096 Jan 24 14:57 ./
drwxrwxr-x 3 student student 4096 Jan 24 14:32 ../
-rwxrwxrwx 1 root    root      57 Jan 24 14:55 du.sh*
-rwxrwxrwx 1 root    root     482 Jan 24 14:45 problem2.sh*
-rwxrwxr-x 1 student student  310 Jan 24 14:33 slog.sh*
-rwxrwxrwx 1 root    root      40 Jan 24 14:53 try.sh*
nitcse@nitcse-OptiPlex-7000:/home/student/Desktop/sayan56/cdlab/bash$ ./du.sh
20K .
```

5. Write a script that would extract URLs from an HTML file using advanced regular expressions with grep -oP and sed.

Code:

```
#!/bin/bash
html_file="sayan.html"
```

```
grep -oP '(?<=href=")[^"]*' "$html_file" | sed 's/^/URL: /'
```

Executing command & Output:

```
nitcse@nitcse-OptiPlex-7000:/home/student/Desktop/sayan56/cdlab/bash$ sudo chmod 777
sayan.html
nitcse@nitcse-OptiPlex-7000:/home/student/Desktop/sayan56/cdlab/bash$ nano sayan.html
nitcse@nitcse-OptiPlex-7000:/home/student/Desktop/sayan56/cdlab/bash$ ./url.sh
URL: https://dpenandpaper.blogspot.com/feeds/posts/default
URL: https://dpenandpaper.blogspot.com/feeds/posts/default?alt=rss
URL: https://www.blogger.com/feeds/3443255668241014471/posts/default
URL: https://dpenandpaper.blogspot.com/feeds/5718535556180333564/comments/default
URL: )[^
```

6. Write a script to define a function 'validate_ipv4' to validate IPv4 addresses using a regular expression. It should check whether the input string matches the specified IPv4 regex pattern.

Code:

```
#!/bin/bash

validate_ipv4() {
   ipv4_regex="^([0-9]{1,3}\.){3}[0-9]{1,3}$"
   if [[ $1 =~ $ipv4_regex ]]; then
       echo "Valid IPv4 address: $1"
   else
       echo "Invalid IPv4 address: $1"
   fi
}
validate_ipv4 "192.168.1.1"
validate_ipv4 "256.0.0.1"
```

Executing command & Output:

```
nitcse@nitcse-OptiPlex-7000:/home/student/Desktop/sayan56/cdlab/bash$ nano ip.sh
nitcse@nitcse-OptiPlex-7000:/home/student/Desktop/sayan56/cdlab/bash$ ./ip.sh
Valid IPv4 address: 192.168.1.1
nitcse@nitcse-OptiPlex-7000:/home/student/Desktop/sayan56/cdlab/bash$ nano ip.sh
nitcse@nitcse-OptiPlex-7000:/home/student/Desktop/sayan56/cdlab/bash$ ./ip.sh
Valid IPv4 address: 192.168.1.1
Valid IPv4 address: 256.0.0.1
```

7. Write a script that uses awk to capture fields in a CSV file, even if they contain commas and are enclosed in double quotes. You may create a sample CSV file for the purpose.

Code:

```
#!/bin/bash
csv_file="sample.csv"
awk -F'","' '{gsub(/^"|"$/,"",$1); gsub(/^"|"$/,"",$2); gsub(/^"|"$/,"",$3); print
"Name: "$1", Age: "$2", Location: "$3}' "$csv_file"
```

Executing command & Output:

```
nitcse@nitcse-OptiPlex-7000:/home/student/Desktop/sayan56/cdlab/bash$ sudo touch
csvfile.sh
nitcse@nitcse-OptiPlex-7000:/home/student/Desktop/sayan56/cdlab/bash$ sudo chmod 777
csvfile.sh
nitcse@nitcse-OptiPlex-7000:/home/student/Desktop/sayan56/cdlab/bash$ nano csvfile.sh
nitcse@nitcse-OptiPlex-7000:/home/student/Desktop/sayan56/cdlab/bash$ sudo touch
sample.csv
nitcse@nitcse-OptiPlex-7000:/home/student/Desktop/sayan56/cdlab/bash$ sudo chmod 777
sample.csv
nitcse@nitcse-OptiPlex-7000:/home/student/Desktop/sayan56/cdlab/bash$ nano sa
sample.csv  sayan.html
nitcse@nitcse-OptiPlex-7000:/home/student/Desktop/sayan56/cdlab/bash$ nano sample.csv
nitcse@nitcse-OptiPlex-7000:/home/student/Desktop/sayan56/cdlab/bash$ ./csvfile.sh
```

8.Create a script that extracts unique email domains from a text file. You may build a text file with many different email addresses with different domains for the purpose.

Code:

```
#!/bin/bash
email_file="emails.txt"
grep -oP '@\K\S+' "$email_file" | sort -u
```

Executing command & Output:

```
nitcse@nitcse-OptiPlex-7000:/home/student/Desktop/sayan56/cdlab/bash$ nano email.sh
```

```
nitcse@nitcse-OptiPlex-7000:/home/student/Desktop/sayan56/cdlab/bash$ sudo touch
emails.txt
nitcse@nitcse-OptiPlex-7000:/home/student/Desktop/sayan56/cdlab/bash$ sudo chmod 777
emails.txt
nitcse@nitcse-OptiPlex-7000:/home/student/Desktop/sayan56/cdlab/bash$ nano emails.txt
nitcse@nitcse-OptiPlex-7000:/home/student/Desktop/sayan56/cdlab/bash$ ./email.sh
example.com
gmail.com
hotmail.co.in
linux.io
nitcse@nitcse-OptiPlex-7000:/home/student/Desktop/sayan56/cdlab/bash$
```