Assignment 6
Given a sample log file, write a script using grep to extract all lines containing "ERROR". Use awk to print the date, time, and error message of each extracted line.
Data Processing with sed
ANS:

sample log file
2024-05-24 14:32:10 INFO Starting the service
2024-05-24 14:33:45 ERROR An error occurred in the service
2024-05-24 14:35:12 INFO Service running smoothly
2024-05-24 14:36:05 ERROR Another error occurred

Use 'grep' to extract lines containing "ERROR".
Use 'awk' to print the date, time, and error message.
Optionally, use 'sed' for further processing if needed.

```
# Check if the log file is provided as an argument
if [ -z "$1" ]; then
  echo "Usage: $0 <logfile>"
  exit 1
fi

LOGFILE="$1"

# Extract lines containing "ERROR" and process with awk
grep "ERROR" "$LOGFILE" | awk '{print $1, $2, $3, $4, $5}'

#Run the script with your log file
./extract_errors.sh sample.log

#Given the example log file provided, the output would be
2024-05-24 14:33:45 ERROR An error occurred in the service
2024-05-24 14:36:05 ERROR Another error occurred

#This output shows the date, time, and error message for each line containing "ERROR".
#If additional text processing is needed, sed can be incorporated. For example, to remove extra spaces in the error message:

#!/bin/bash

# Check if the log file is provided as an argument
if [ -z "$1" ]; then
  echo "Usage: $0 <logfile>"
  exit 1
fi
```

LOGFILE="$1"

 # Extract lines containing "ERROR", process with awk, and further refine with sed
grep "ERROR" "$LOGFILE" | awk '{print $1, $2, $3, $4, substr($0, index($0,$5))}' | sed 's/ \+/ /g'S

```
~$ cd New
~/New$ # Check if the correct number of arguments is provided
~/New$ if [ "$#" -ne 4 ]; then
>     echo "Usage: $0 input_file old_text new_text output_file"
>     exit 1
> fi
Usage: /bin/bash input_file old_text new_text output_file
exit


[Process completed - press any key]

~$
~$ input_file=$1
~$ old_text=$2
~$ new_text=$3
~$ output_file="output_${input_file}"
~$
~$ # Use sed to replace all occurrences of old_text with new_text and save to a new file
~$ sed "s/${old_text}/${new_text}/g" "$input_file" > "$output_file"
sed: can't read : No such file or directory
~$
~$ # Notify the user of the operation completion
~$ echo "All occurrences of '${old_text}' have been replaced with '${new_text}' in the file '${input_file}'."
All occurrences of '' have been replaced with '' in the file ''.
~$ echo "The result has been saved to '${output_file}'."
The result has been saved to 'output_'.
~$
```