

## Assignment-Day-5

**Assignment 1: Ensure the script checks if a specific file (e.g., myfile.txt) exists in the current directory. If it exists, print "File exists", otherwise print "File not found".**

**Solution:**

```
#!/bin/bash

# Specify the file name to check
filename="myfile.txt"

# Check if the file exists in the current directory
if [ -f "$filename" ]; then
    echo "File exists"
else
    echo "File not found"
fi
```

**Assignment 2: Write a script that reads numbers from the user until they enter '0'. The script should also print whether each number is odd or even.**

**Solution:**

```
#!/bin/bash

while true; do
    #Prompt the user for input
    read -p "Enter a number(0 to exit): " num

    # Check if the input is '0' to break out of the loop
    if [ "$num" -eq 0 ]; then
        echo "Exiting the program."
        break
    fi

    # Check if the number is odd or even
    if [ $((num % 2)) -eq 0 ]; then
```

```
        echo "$num is even."
    else
        echo "$num is odd."
    fi
done
```

**Assignment 3: Create a function that takes a filename as an argument and prints the number of lines in the file. Call this function from your script with different filenames.**

```
#!/bin/bash

function count_lines {
    filename="$1"
    if [ -f"$filename" ]; then
        line_count=$(wc -l < "$filename")
        echo "$filename has $line_count lines."
    else
        echo "File '$filename' not found."
    fi
}

count_lines "myfile.txt"
count_line "change_file.txt"
```

**Assignment 4: Write a script that creates a directory named TestDir and inside it, creates ten files named File1.txt, File2.txt, ... File10.txt. Each file should contain its filename as its content (e.g., File1.txt contains "File1.txt").**

**Solution:**

```
#!/bin/bash

# Create the directory TestDir if it doesn't exist
Mkdir -p TestDir

# Loop through and create the files
```

```
for ((i=1; i<=10; i++)); do
    filename="File$i.txt"
    echo "$filename" > "TestDir/$filename"
done
```

**Assignment 5: Modify the script to handle errors, such as the directory already existing or lacking permissions to create files.**

**Add a debugging mode that prints additional information when enabled.**

**Solution:**

```
#!/bin/bash

# Directory name
dir_name="TestDir"

#Number of files
num_files=10

#Enable debugging mode (set to true for additional info)
debug_mode=false

#Function to print debug message
function debug_print {
    if [[ "$debug_mode" == true ]]; then
        echo "[DEBUG] $1"
    fi
}

#Check if directory already exist (informative message)
if [ -d "$dir_name" ]; then
    echo "Directory '$dir_name' already exists. Skipping creation."
    exit 0
fi

#Create the directory (handle errors)
debug_print "Creating directory: $dir_name"
if ! mkdir -p "$dir_name"; then
```

```

echo "Error: Insufficient permission to create directory '$dir_name'."
exit 1
fi

#Loop to create files with unique content
for I in $(seq 1 $num_files); do
filename="File$i.txt"
filepath="$dir_name/$filename"
#Create the file and write content (redirect to avoid overwriting)
debug_print "Creating file: $filepath"
if ! echo "$filename" > "$filepath"; then
echo "Error creating file '$filepath'."
exit 1
fi
done
echo "Created directory '$dir_name' with $num_files files."

```

**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**

#### **Solution:**

```

#!/bin/bash

#Sample log file path
logfile="sample.log"

#Use grep to extract lines containing "ERROR" and pipe the output to awk
grep "ERROR" "$logfile" | awk '{
    # Use regex to extract date, time, and error message
    if ($0~ /[0-9]{4}-[0-9]{2} [0-9]{2}:[0-9]{2}:[0-9]{2}/){
        match($0,/ [0-9]{4}-[0-9]{2}-[0-9]{2} [0-9]{2}:[0-9]{2}:[0-9]{2}/);
    }
}'

```

```
    date_time = substr($0, RSTART, RLENGTH);  
    message = substr($0, RSTART + RLENGTH + 1);  
    print "Date-Time: " date_time ", Error Message: " message;  
}  
}
```

**Assignment 7: Create a script that takes a text file and replaces all occurrences of "old\_text" with "new\_text". Use sed to perform this operation and output the result to a new file.**

**Solution:**

```
#!/bin/bash  
  
# Check if correct number of arguments are provided  
if [ $# -ne 3 ]; then  
    echo "Usage: $0 input_file old_text"  
    exit 1  
fi  
  
#Assign input arguments to variables  
input_file="$1"  
old_text="$2"  
new_text="$3"  
output_file="${input_file}_updated"  
  
# Replace old_text with new_text using sed and save output to a new file  
sed "s/$old_text/$new_text/g" "$input_file" > "$output_file"  
  
# Display success message  
echo "Replacement complete. Updated content saved to $output_file"
```

