Shell Scripting with Bash

Name: Vaishali Ramesh Kale

Email ID: kalevaishalir16@gmail.com

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".

main.bash

```bash
#                                Online Bash Shell.
#                   Code, Compile, Run and Debug Bash script online.
# Write your code in this editor and press "Run" button to execute it.

#!/bin/bash

file_name="myfile.txt"

if [ -e "$file_name" ]; then
    echo "File exists"
else
    echo "File not found"
fi

```

input

```
File not found


...Program finished with exit code 0
Press ENTER to exit console.
```

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.

main.bash | file1.txt | file2.txt | file3.txt

```bash
#!/bin/bash

while true; do
    echo -n "Enter a number (0 to exit): "
    read num

    if [ "$num" -eq 0 ]; then
        echo "Exiting..."
        break
    elif [ "$((num % 2))" -eq 0 ]; then
        echo "$num is even"
    else
        echo "$num is odd"
    fi
done
```

```
Enter a number (0 to exit): 16
16 is even
Enter a number (0 to exit): 13
13 is odd
Enter a number (0 to exit): 0
Exiting...
```

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.

main.bash | file1.txt | file2.txt | file3.txt

```bash


#!/bin/bash

# Function to count the number of lines in a file
count_lines() {
    file="$1"
    lines=$(wc -l < "$file")
    echo "Number of lines in $file: $lines"
}

# Call the function with different filenames
count_lines "file1.txt"
count_lines "file2.txt"
count_lines "file3.txt"
```

```
Number of lines in file1.txt: 2
Number of lines in file2.txt: 3
Number of lines in file3.txt: 4
```

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")

main.bash | file1.txt | file2.txt | file3.txt | Tes

```bash

#!/bin/bash
# Create a directory named TestDir
mkdir TestDir

# Change into the TestDir directory
cd TestDir || exit

# Loop to create ten files
for ((i = 1; i <= 10; i++)); do
    file_name="File${i}.txt"
    echo "$file_name" > "$file_name"
done

echo "Files created successfully."
```

```
Files created successfully.
```

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.

Code:

MINGW64:/d

```
$ cat errorhand.sh
#!/bin/bash
# Function to create a directory and handle errors
create_directory() {
    local path="$1"
    local debug="$2"
    if [ "$debug" == "true" ]; then
        echo "Attempting to create directory: $path"
    fi
    if mkdir -p "$path" 2>/dev/null; then
        if [ "$debug" == "true" ]; then
            echo "Directory created successfully: $path"
        fi
    else
        if [ -d "$path" ]; then
            if [ "$debug" == "true" ]; then
                echo "Directory already exists: $path"
            fi
        else
            echo "Permission denied or other error: Unable to create directory $path"
        fi
    fi
}
# Function to create a file and handle errors
create_file() {
    local path="$1"
    local content="$2"
    local debug="$3"
    if [ "$debug" == "true" ]; then
        echo "Attempting to create file: $path"
    fi

    if echo "$content" > "$path" 2>/dev/null; then
        if [ "$debug" == "true" ]; then
            echo "File created successfully: $path"
        fi
    else
        if [ -f "$path" ]; then
            echo "File already exists and cannot be overwritten: $path"
        else
            echo "Permission denied or other error: Unable to create file $path"
        fi
    fi
}
# Main function to demonstrate the creation of directory and file
main() {
    local directory_path="example_dir"
    local file_path="$directory_path/example_file.txt"
    local file_content="This is a sample content."

    # Enable or disable debugging mode
    local debug_mode="true"

    create_directory "$directory_path" "$debug_mode"
    create_file "$file_path" "$file_content" "$debug_mode"
}
main
```

Output:

```
MINGW64:/d

Bileni@DESKTOP-SDDJ0CG MINGW64 /d
$ chmod +x errorhand.sh

Bileni@DESKTOP-SDDJ0CG MINGW64 /d
$ ./errorhand.sh
Attempting to create directory: example_dir
Directory created successfully: example_dir
Attempting to create file: example_dir/example_file.txt
File created successfully: example_dir/example_file.txt
```

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

1). Create Sample log file:

```
Bileni@DESKTOP-SDDJ0CG MINGW64 /d
$ cat sample.log
2024-05-17 10:15:32 INFO Starting the application
2024-05-17 10:16:33 ERROR Failed to connect to the database
2024-05-17 10:17:34 WARN Low disk space
2024-05-17 10:18:35 ERROR Failed to load configuration file
2024-05-17 10:19:36 INFO Application shutdown

Bileni@DESKTOP-SDDJ0CG MINGW64 /d
$ |
```

2). Bash Script:

```
MINGW64:/d

Bileni@DESKTOP-SDDJOCG MINGW64 /d
$ log_file="sample.log"

Bileni@DESKTOP-SDDJOCG MINGW64 /d
$ grep "ERROR" "$log_file" | awk '{print $1,$2 substr($0, index($0,$4))}'
2024-05-17 10:16:33Failed to connect to the database
2024-05-17 10:18:35Failed to load configuration file

Bileni@DESKTOP-SDDJOCG MINGW64 /d
$ grep "ERROR" "$log_file" | sed 's/ERROR/ERR/' | awk '{print $1,$2 substr($0, i
ndex($0,$4))}'
2024-05-17 10:16:33Failed to connect to the database
2024-05-17 10:18:35Failed to load configuration file

Bileni@DESKTOP-SDDJOCG MINGW64 /d
$ |
```

Explanation:

- grep "ERROR" "$LOG_FILE": Extracts lines containing the word "ERROR" from the log file.
- awk '{print $1, $2, substr($0, index($0,$4))}':
- $1 and $2 represent the date and time fields, respectively.
- substr($0, index($0,$4)) extracts the error message starting from the fourth field (which is the error message in this case).
- Data Processing with sed:
- sed 's/ERROR/ERR/' replaces the word "ERROR" with "ERR" in the extracted lines before processing with awk.

3). Running the Script

Make the script executable and run it:

```
Bileni@DESKTOP-SDDJOCG MINGW64 /d
$ cat sample.log
2024-05-17 10:15:32 INFO Starting the application
2024-05-17 10:16:33 ERROR Failed to connect to the database
2024-05-17 10:17:34 WARN Low disk space
2024-05-17 10:18:35 ERROR Failed to load configuration file
2024-05-17 10:19:36 INFO Application shutdown

Bileni@DESKTOP-SDDJOCG MINGW64 /d
$ chmod +x extract_errors.sh

Bileni@DESKTOP-SDDJOCG MINGW64 /d
$ ./extract_errors.sh
2024-05-17 10:16:33 Failed to connect to the database
2024-05-17 10:18:35 Failed to load configuration file
2024-05-17 10:16:33 Failed to connect to the database
2024-05-17 10:18:35 Failed to load configuration file

Bileni@DESKTOP-SDDJOCG MINGW64 /d
$
```

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.

1). Create a script named replace.sh:

Vi replace.sh

MINGW64:/d

```
#!/bin/bash

# Check if the correct number of arguments are provided
if [ "$#" -ne 3 ]; then
    echo "Usage: $0 input_file old_text new_text"
    exit 1
fi

# Assign arguments to variables
input_file="$1"
old_text="$2"
new_text="$3"
output_file="output_$(basename "$input_file")"

# Perform the text replacement using sed and output to a new file
sed "s/$old_text/$new_text/g" "$input_file" > "$output_file"

# Check if the sed command was successful
if [ $? -eq 0 ]; then
    echo "Text replacement successful. Output written to $output_file."
    echo ""
    echo "Contents of $output_file:"
    echo "----------------------------------------"
    cat "$output_file"
    echo "----------------------------------------"
else
    echo "Error occurred during text replacement."
    exit 1
fi
```

2). Create a input.txt file:

```
$ cat input.txt
This is the old_text that needs to be replaced.
Here is another old_text occurrence.

Bileni@DESKTOP-SDDJ0CG MINGW64 /d
$
```

3). Make the script executable and run it:

Set file permission:

See output:

```
Bileni@DESKTOP-SDDJOCG MINGW64 /d
$ chmod +x replace.sh

Bileni@DESKTOP-SDDJOCG MINGW64 /d
$ ./replace.sh input.txt old_text new_text
Text replacement successful. Output written to output_input.txt.

Contents of output_input.txt:
----------------------------------------
This is the new_text that needs to be replaced.
Here is another new_text occurrence.
----------------------------------------

Bileni@DESKTOP-SDDJOCG MINGW64 /d
$
```