Shell - Assignment 2

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".

Script :

```
#!/bin/bash
filename="myfile.txt"
if [ -f "$filename" ]; then
        echo "File exists"
else
        echo "File not found"
fi
~
```

Commands to run the script written above:

```
kingsnehith@hp:/mnt/c/WINDOWS/system32/myfolder$ vi file_search.sh
kingsnehith@hp:/mnt/c/WINDOWS/system32/myfolder$ chmod +x file_search.sh
kingsnehith@hp:/mnt/c/WINDOWS/system32/myfolder$ bash file_search.sh
File exists
```

Process :

- Open a text editor using **'vim'**.
- Write and save the script in **'file_search.sh'**.
- Make the script executable using **'chmod +x file_search.sh'**.
- Run the script using **'bash file_search.sh'**.

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.

Script :

```
#!/bin/bash
while true; do
        read -p "Enter a number: " number
        if [ "$number" -eq 0 ]; then
                break
        fi
        if [ $((number % 2)) -eq 0 ]; then
                echo "$number is even"
        else
                echo "$numbmer is odd"
        fi
done
```

Commands :

```
kingsnehith@hp:/mnt/c/WINDOWS/system32/myfolder$ vi odd_or_even.sh
kingsnehith@hp:/mnt/c/WINDOWS/system32/myfolder$ chmod +x odd_or_even.sh
kingsnehith@hp:/mnt/c/WINDOWS/system32/myfolder$ bash odd_or_even.sh
Enter a number: 4
4 is even
Enter a number: 5
 is odd
Enter a number: 6
6 is even
Enter a number: 17
 is odd
Enter a number: 20
20 is even
Enter a number: 0
kingsnehith@hp:/mnt/c/WINDOWS/system32/myfolder$
```

Process :

- Create the script and save it as '**odd_or_even.sh**'.
- Make it executable using '**chmod +x odd_or_even.sh**'.
- Run the script with **'bash odd_or_even.sh'**.
- Enter numbers to check if they are odd or even, and enter '0' to stop.

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.

Script :

```
#!/bin/bash
count_lines() {
        local filename=$1
        if [ -f "$filename" ]; then
                local line_count=$(wc -l < "$filename")
                echo "File "$filename" has $line_count lines."
        else
                echo "File "$filename" does not exist."
        fi
}

count_lines "odd_or_even.sh"
count_lines "file_search.sh"
count_lines "file.txt"
~
```

Commands :

```
kingsnehith@hp:/mnt/c/WINDOWS/system32/myfolder$ vi count_lines.sh
kingsnehith@hp:/mnt/c/WINDOWS/system32/myfolder$ chmod +x count_lines.sh
kingsnehith@hp:/mnt/c/WINDOWS/system32/myfolder$ bash count_lines.sh
File odd_or_even.sh has 12 lines.
File file_search.sh has 7 lines.
File file.txt has 2 lines.
kingsnehith@hp:/mnt/c/WINDOWS/system32/myfolder$
```

Process :

- Create the script and save it as ;**count_lines.sh'**.
- Make it executable with '**chmod +x count_lines.sh'**.
- Run the script using '**bash count_lines.sh'**.

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").

Script :

```
#!/bin/bash
mkdir -p TestDir
cd TestDir
for i in {1..10}; do
        filename="File$i.txt"
        echo "$filename" > "$filename"
done
echo "10 files created in TestDir."
~
```

Commands :

```
kingsnehith@hp:/mnt/c/WINDOWS/system32/myfolder$ vi count_lines.sh
kingsnehith@hp:/mnt/c/WINDOWS/system32/myfolder$ vi create_files.sh
kingsnehith@hp:/mnt/c/WINDOWS/system32/myfolder$ chmod +x create_files.sh
kingsnehith@hp:/mnt/c/WINDOWS/system32/myfolder$ bash create_files.sh
10 files created in TestDir.
kingsnehith@hp:/mnt/c/WINDOWS/system32/myfolder$ ls
TestDir         create_files.sh  file_search.sh  odd_or_even.sh
count_lines.sh  file.txt         myfile.txt
kingsnehith@hp:/mnt/c/WINDOWS/system32/myfolder$ cd TestDir/
kingsnehith@hp:/mnt/c/WINDOWS/system32/myfolder/TestDir$ ls
File1.txt   File2.txt  File4.txt  File6.txt  File8.txt
File10.txt  File3.txt  File5.txt  File7.txt  File9.txt
```

Process :

- Create the script and save it as '**create_files.sh**'.
- Make it executable with '**chmod +x create_files.sh**'.
- Run the script using '**bash create_files.sh**'.

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.

Script :

```
#!/bin/bash
DEBUG=false
if [ "$1" == "debug" ]; then
        DEBUG=true
fi

debug() {
        if [ "$DEBUG" == "true" ]; then
                echo "DEBUG: $1"
        fi
}

if mkdir MyDir 2>/dev/null; then
        debug "Directory MyDir created."
else
        if [ -d "MyDir" ]; then
                echo "Error: Directory MyDir already exists."
        else
                echo "Error: Failed to create directory MyDir. Check your permissions."
                exit 1
        fi
fi

if cd MyDir; then
        debug "Changed to MyDir directory."
else
        echo "Error: Failed to change to MyDir directory."
        exit 1
fi

for i in {1..10}; do
        filename="File$i.txt"
        if echo "$filename" > "$filename"; then
                debug "File $filename created with content."
        else
                echo "Error: Failed to create file $filename. Check your permissions."
                exit 1
        fi
done

echo "10 files created successfully in MyDir."
```

Commands :

```
kingsnehith@hp:/mnt/c/WINDOWS/system32/myfolder$ vi handling_errors.sh
kingsnehith@hp:/mnt/c/WINDOWS/system32/myfolder$ chmod +x handling_errors.sh
kingsnehith@hp:/mnt/c/WINDOWS/system32/myfolder$ bash handling_errors.sh
Error: Directory MyDir already exists.
10 files created successfully in MyDir.
kingsnehith@hp:/mnt/c/WINDOWS/system32/myfolder$ bash handling_errors.sh debug
Error: Directory MyDir already exists.
DEBUG: Changed to MyDir directory.
DEBUG: File File1.txt created with content.
DEBUG: File File2.txt created with content.
DEBUG: File File3.txt created with content.
DEBUG: File File4.txt created with content.
DEBUG: File File5.txt created with content.
DEBUG: File File6.txt created with content.
DEBUG: File File7.txt created with content.
DEBUG: File File8.txt created with content.
DEBUG: File File9.txt created with content.
DEBUG: File File10.txt created with content.
10 files created successfully in MyDir.
```

Process :

1. Debug Mode : Enable by passing "debug" as the first argument.
2. Debug Function : Prints debug messages if debugging is enabled.
3. Create Directory :
    - Attempts to create ‘TestDir’.
    - If ‘TestDir’ already exists, it prints an error message.
    - If another error occurs (like lack of permissions), it prints an error and exits.
4. Change Directory :
    - Changes to ‘TestDir’.
    - If it fails, it prints an error and exits.
5. Create Files :
    - Loops to create ‘File1.txt’ to ‘File10.txt’.
    - Writes the filename as the content.
    - If it fails (like lack of permissions), it prints an error and exits.

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

Content : sample.log file

```
2024-05-17 10:30:15 ERROR: File not found
2024-05-17 10:31:20 WARNING: Network connection lost
2024-05-17 10:32:05 ERROR: Database connection failed
2024-05-17 10:33:10 ERROR: Out of memory
2024-05-17 10:34:25 ERROR: Configuration file corrupted
2024-05-17 10:35:30 INFO: Application started
2024-05-17 10:36:45 ERROR: Server timeout
2024-05-17 10:37:50 ERROR: Invalid input received
2024-05-17 10:38:55 DEBUG: Debugging information
2024-05-17 10:40:00 ERROR: Disk space full
```

Script using awk :

```
#!/bin/bash
logfile="sample.log"
grep "ERROR" "$logfile" | awk '{print $1, $2, substr($0, index($0,$3))}'
~
```

Commands when awk is used :

```
kingsnehith@hp:/mnt/c/WINDOWS/system32/myfolder$ vi error.sh
kingsnehith@hp:/mnt/c/WINDOWS/system32/myfolder$ bash error.sh
2024-05-17 10:30:15 ERROR: File not found
2024-05-17 10:32:05 ERROR: Database connection failed
2024-05-17 10:33:10 ERROR: Out of memory
2024-05-17 10:34:25 ERROR: Configuration file corrupted
2024-05-17 10:36:45 ERROR: Server timeout
2024-05-17 10:37:50 ERROR: Invalid input received
2024-05-17 10:40:00 ERROR: Disk space full
```

Script using SED :

```
#!/bin/bash
logfile="sample.log"
grep "ERROR" "$logfile" | sed -E 's/^([0-9]{4}-[0-9]{2}-[0-9]{2} [0-9]{2}:[0-9]{2}:[0-9]{2}) (.*)$/\1 \2/'
```

Commands when SED is used :

```
kingsnehith@hp:/mnt/c/WINDOWS/system32/myfolder$ vi error.sh
kingsnehith@hp:/mnt/c/WINDOWS/system32/myfolder$ bash error.sh
2024-05-17 10:30:15 ERROR: File not found
2024-05-17 10:32:05 ERROR: Database connection failed
2024-05-17 10:33:10 ERROR: Out of memory
2024-05-17 10:34:25 ERROR: Configuration file corrupted
2024-05-17 10:36:45 ERROR: Server timeout
2024-05-17 10:37:50 ERROR: Invalid input received
2024-05-17 10:40:00 ERROR: Disk space full
```

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.

Script :

```
#!/bin/bash
if [ "$#" -ne 3 ]; then
        echo "Usage: $0 input_file old_text new_text"
        exit 1
fi
input_file="$1"
old_text="$2"
new_text="$3"

if [ ! -f "$input_file" ]; then
        echo "Error: Input file '$input_file' not found."
        exit 1
fi
output_file="${input_file%.txt}_updated.txt"
sed "s/$old_text/$new_text/g" "$input_file" > "$output_file"
echo "updated and result written to '$output_file'."
~
```

Commands :

```
kingsnehith@hp:/mnt/c/WINDOWS/system32/myfolder$ cat sample.log
2024-05-17 10:30:15 ERROR: File not found
2024-05-17 10:31:20 WARNING: Network connection lost
2024-05-17 10:32:05 ERROR: Database connection failed
2024-05-17 10:33:10 ERROR: Out of memory
2024-05-17 10:34:25 ERROR: Configuration file corrupted
2024-05-17 10:35:30 INFO: Application started
2024-05-17 10:36:45 ERROR: Server timeout
2024-05-17 10:37:50 ERROR: Invalid input received
2024-05-17 10:38:55 DEBUG: Debugging information
2024-05-17 10:40:00 ERROR: Disk space full

kingsnehith@hp:/mnt/c/WINDOWS/system32/myfolder$ vi new_test.sh
kingsnehith@hp:/mnt/c/WINDOWS/system32/myfolder$ chmod +x new_test.sh
kingsnehith@hp:/mnt/c/WINDOWS/system32/myfolder$ bash new_test.sh sample.log ERROR Warning
updated and result written to 'sample.log_updated.txt'.
kingsnehith@hp:/mnt/c/WINDOWS/system32/myfolder$ cat sample.log_updated.txt
2024-05-17 10:30:15 Warning: File not found
2024-05-17 10:31:20 WARNING: Network connection lost
2024-05-17 10:32:05 Warning: Database connection failed
2024-05-17 10:33:10 Warning: Out of memory
2024-05-17 10:34:25 Warning: Configuration file corrupted
2024-05-17 10:35:30 INFO: Application started
2024-05-17 10:36:45 Warning: Server timeout
2024-05-17 10:37:50 Warning: Invalid input received
2024-05-17 10:38:55 DEBUG: Debugging information
2024-05-17 10:40:00 Warning: Disk space full
```

Process :

1. Input Validation:
    - The script checks if the correct number of arguments (input file, old text, new text) are provided. If not, it prints usage instructions and exits.
    - It checks if the input file exists. If not, it prints an error message and exits.

2. File Processing:

- The script defines the output file name based on the input file name with "_updated" appended.
- It uses sed to perform the text replacement, substituting all occurrences of "old_text" with "new_text".
- The output is redirected to the output file.