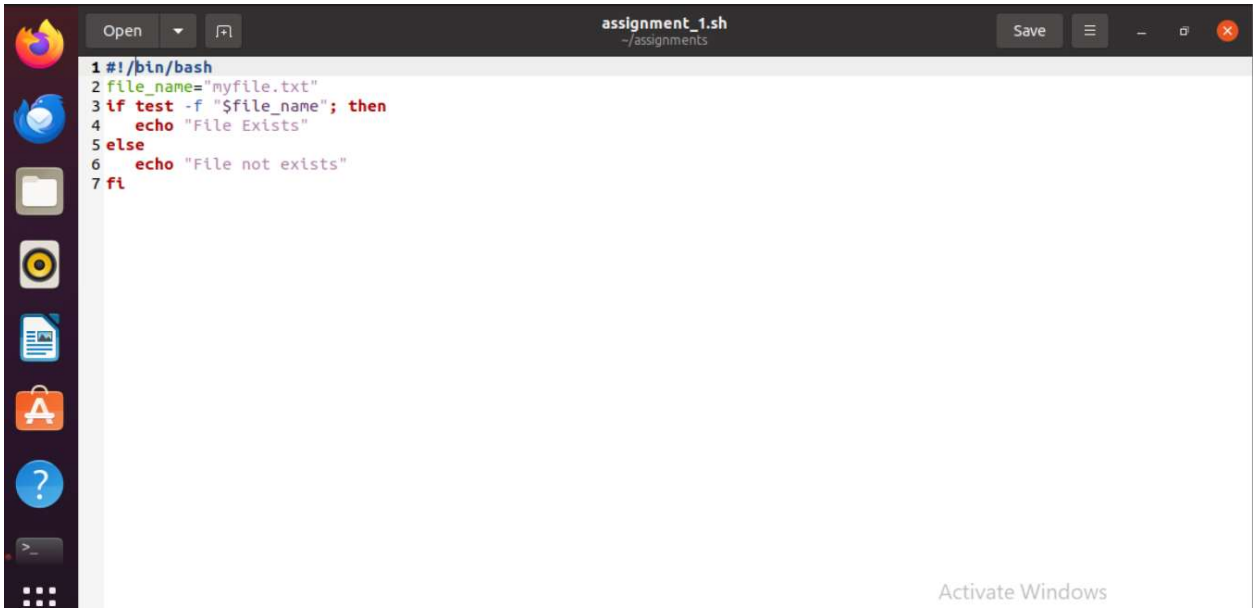
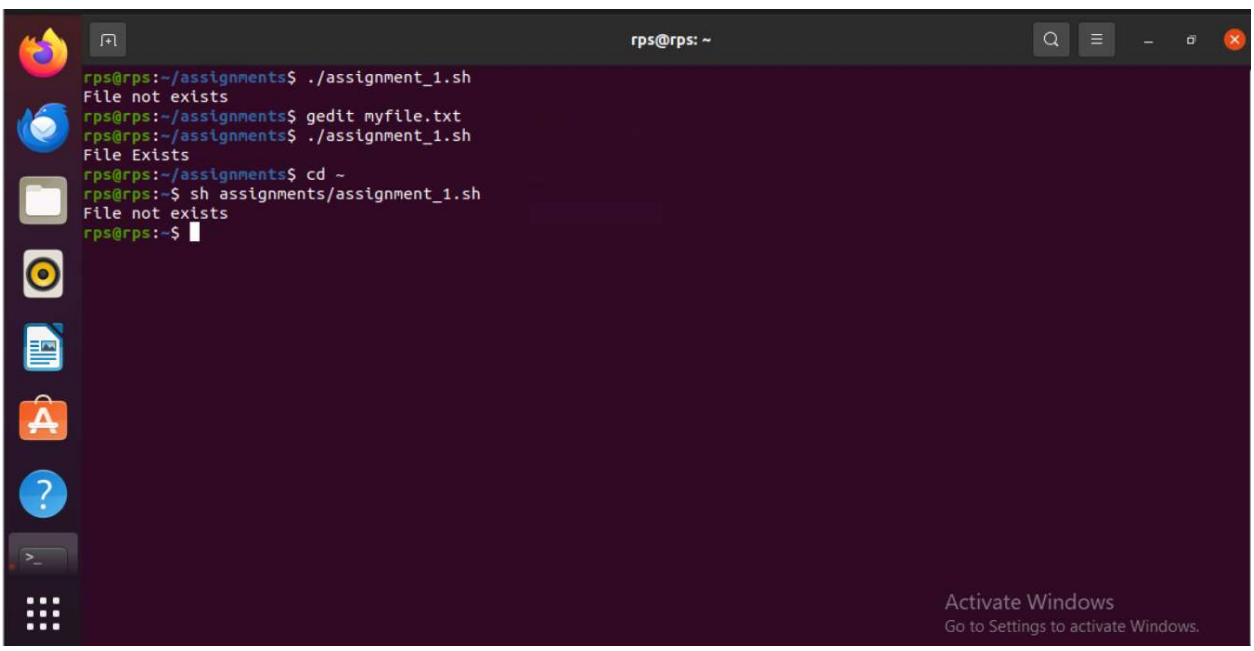
Assignment 1: Ensure the script checks if a specific file (e.g., myfile.txt) exists in the current directory. If it exists, print "File exists", otherwise print "File not found".

Shell script to check if a specific file (e.g., myfile.txt) exists in the current directory:



Terminal Output:

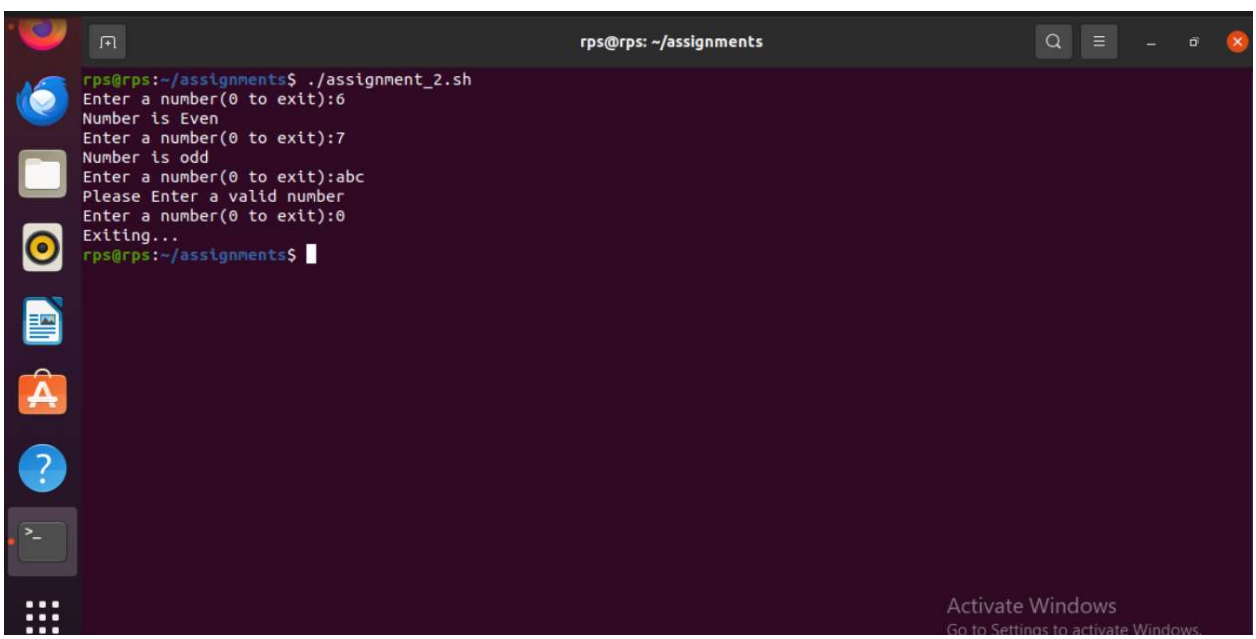


Assignment 2: Write a script that reads numbers from the user until they enter '0'. The script should also print whether each number is odd or even.

Shell script to reads number and print whether each number is odd or even:


```
#!/bin/bash
check_fn(){
    local num=$1

    if((num % 2 == 0)); then
       echo "Number is Even"
    else
       echo "Number is odd"
    fi
}

read_fn(){
     local number
     while true; do
       read -p "Enter a number(0 to exit):" number
       if [[ $number =~ ^-?[0-9]+$ ]]; then
           if test "$number" -eq 0; then
           echo "Exiting..."
           break
           elif test "$number" -gt 0; then
              check_fn "$number"
           fi
       else
         echo "Please Enter a valid number"
       fi
     done
}
read_fn
```

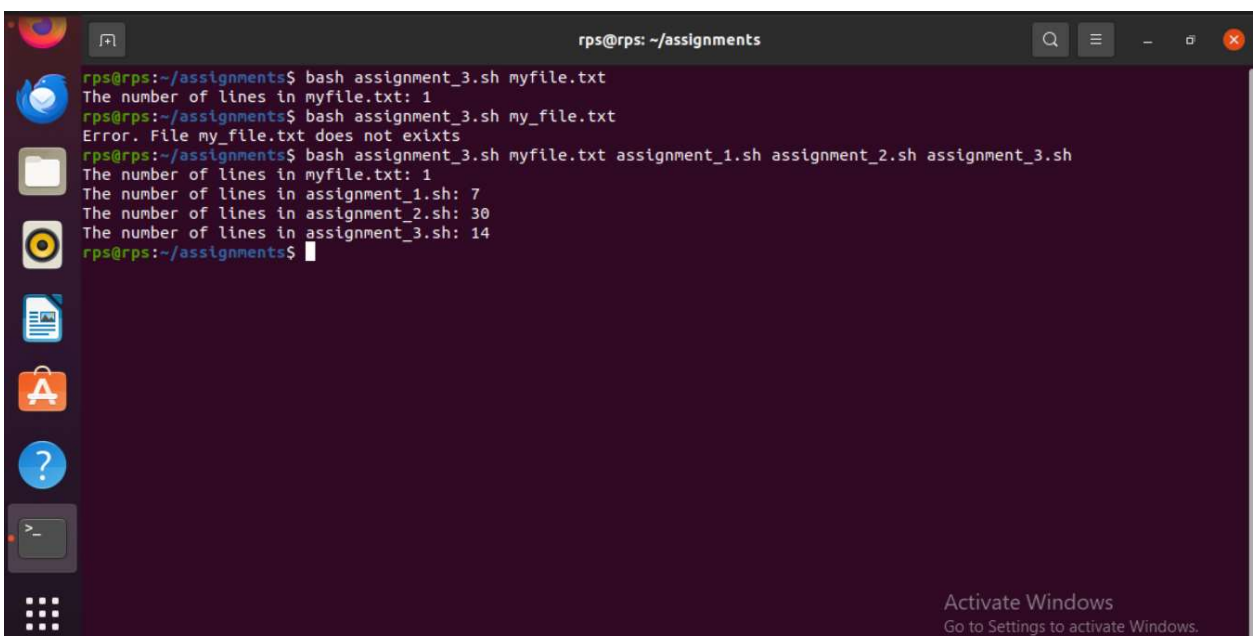
Terminal Output:



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.

Shell script to write a function that takes a filename and prints the number of lines in the file:

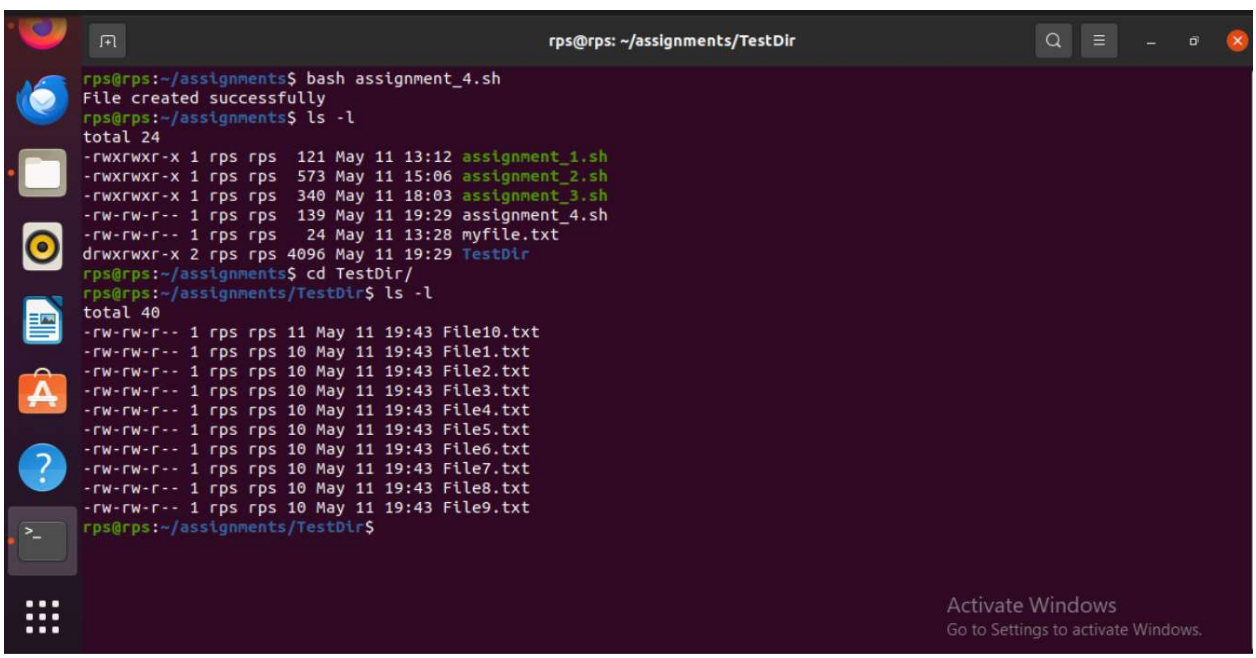
Terminal Output:



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").

Shell script to that creates a directory named TestDir and inside it creates ten files with its filename as its content:

Terminal Output:



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.

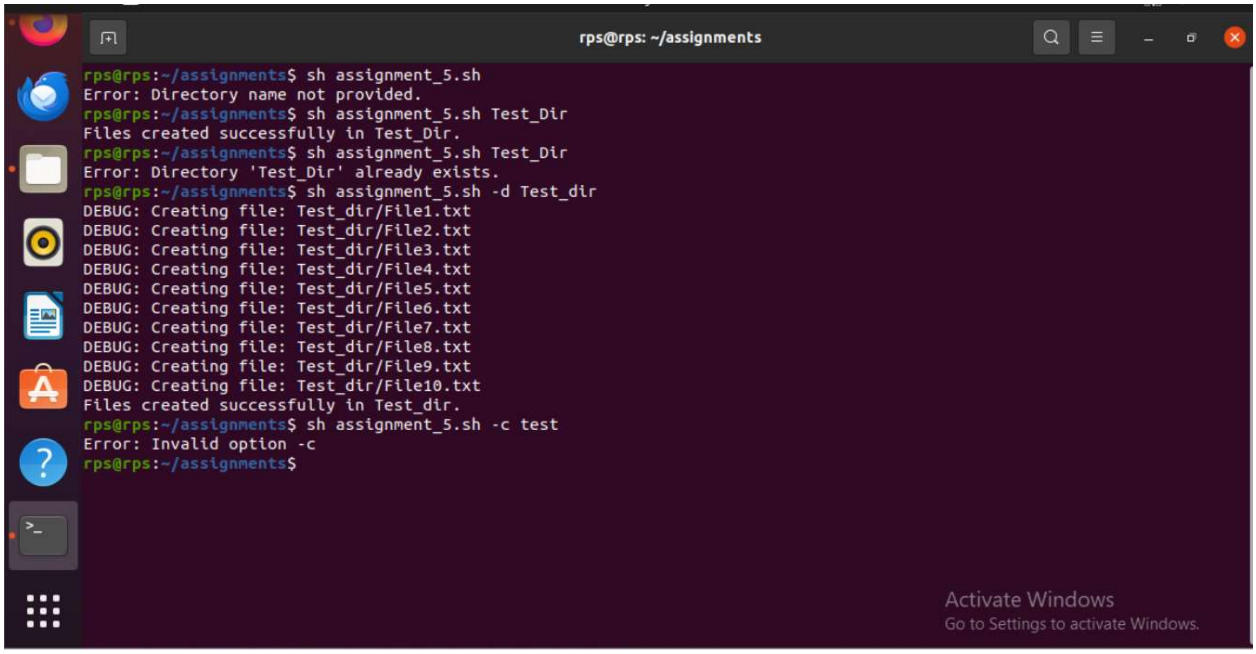
Shell script to handle errors with addition of debugging mode that prints additional information when enabled:

```
debug_msg() {
   [ "$DEBUG" = "true" ] && echo "DEBUG: $1"
}

file_create() {
   local directory="$1"
   if [ -d "$directory" ]; then
      echo "Error: Directory '$directory' already exists." >&2
      exit 1
   fi

   mkdir -p "$directory" || { echo "Error: Failed to create directory '$directory'." >&2; exit 1; }

   filename=""
   for i in $(seq 1 10); do
       filename="$directory/File${i}.txt"
       debug_msg "Creating file: $filename"
       echo "File${i}.txt" > "$filename" || { echo "Error: Failed to create file '$filename'." >&2; exit 1; }
   done

   echo "Files created successfully in $directory."
}

while getopts ":d" opt; do
   case $opt in
        d) DEBUG="true";;
        \?) echo "Error: Invalid option -$OPTARG" >&2; exit 1;;
```

sh Tab Width: 8 Ln 35, Col 10 INS

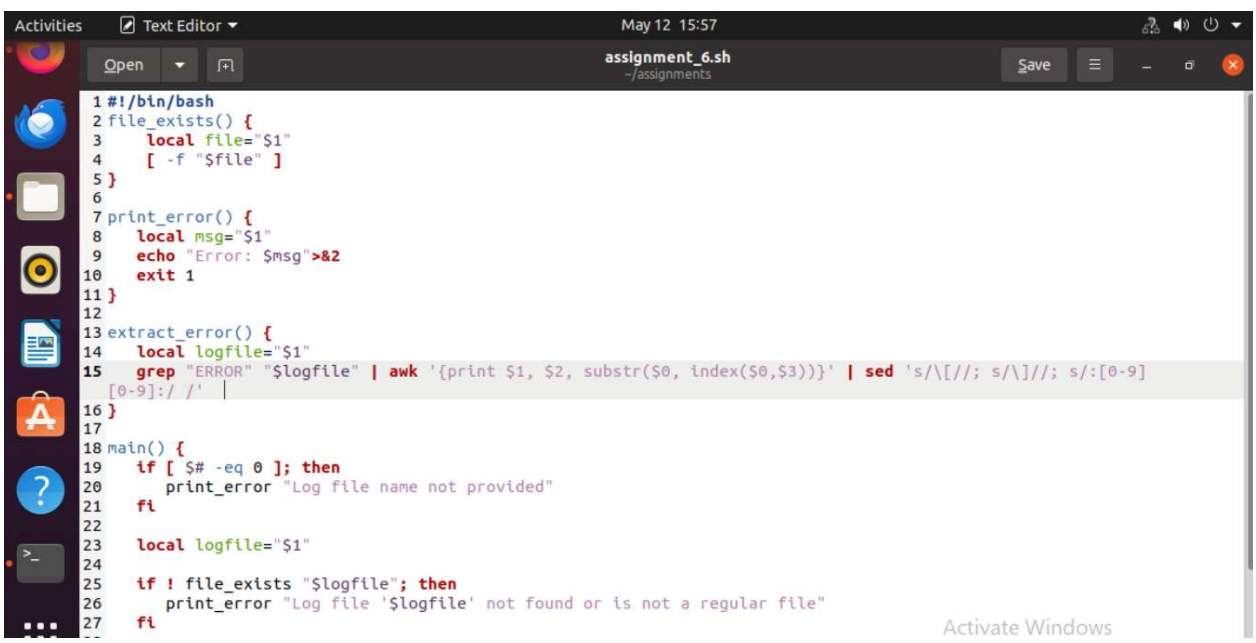
```
   esac
done

shift $((OPTIND -1))

if [ $# -eq 0 ]; then
   echo "Error: Directory name not provided." >&2
   exit 1
fi

file_create "$1"
```

Terminal Output:



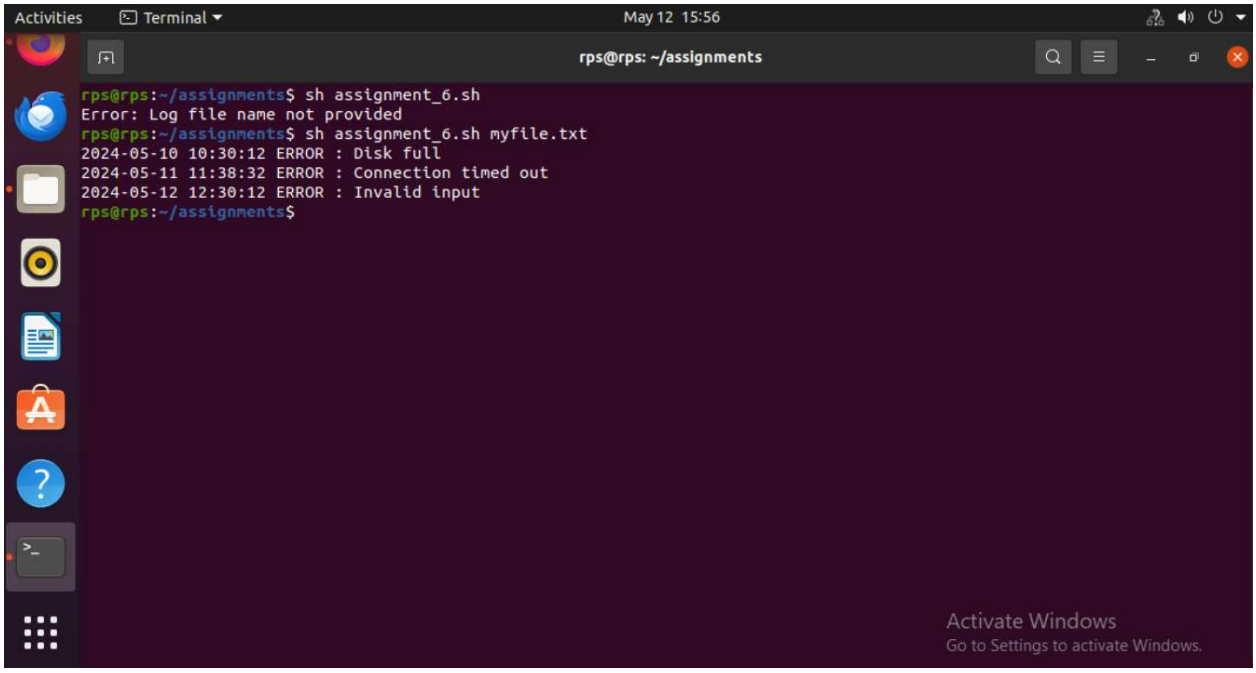
Assignment 6: Given a sample log file, write a script using grep to extract all lines containing "ERROR". Use awk to print the date, time, and error message of each extracted line.
Data Processing with sed

Shell script to extract all lines containing "ERROR" and to print the date, time, and error message:



```
  extract_error "$logfile"
}

main "$@"
```


Terminal Output:



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.

Shell script to take a text file and replaces all occurrences of "old_text" with "new_text":

assignment_7.sh
~/assignments

```
#!/bin/bash
text_replacement() {
    local input_file=$1
    local old_text=$2
    local new_text=$3

    if [ -z "$old_text" ] || [ -z "$new_text" ]; then
        echo "Error: Both old_text and new_text must be non empty."
        exit 1
    fi

    if [ ! -f "$input_file" ]; then
        echo "Error: Input file '$input_file' not found."
        exit 1
    fi

    local output_file="${input_file%.txt}_replaced.txt"
    sed "s/$old_text/$new_text/g" "$input_file" > "$output_file" 2>&1

    echo "File Replacement completed and Output is saved to $output_file"
}

check_arg() {
    if [ "$#" -ne 3 ]; then
        echo "Usage: $0 input_file old_text new_text"
        exit 1
    fi
}

main() {
    check_arg "$@"
    text_replacement "$@"
}
main "$@"
```

Terminal Output:

Activities Terminal May 12 18:02

rps@rps: ~/assignments

```
rps@rps:~/assignments$ sh assignment_7.sh
Usage: assignment_7.sh input_file old_text new_text
rps@rps:~/assignments$ sh assignment_7.sh input.txt "" "Is"
Error: Both old_text and new_text must be non empty.
rps@rps:~/assignments$ sh assignment_7.sh input.txt "This is a" "It was the"
File Replacement completed and Output is saved to input_replaced.txt
rps@rps:~/assignments$
```