Assignment 1:

Initialize a new Git repository in a directory of your choice. Add a simple text file to the repository and make the first commit.

Sol:

Administrator@DESKTOP-TIC5DM4 MINGW64 / (master)

$ mkdir Myproject

Administrator@DESKTOP-TIC5DM4 MINGW64 / (master)

$ pwd

/

Administrator@DESKTOP-TIC5DM4 MINGW64 / (master)

$ cd Myproject

Administrator@DESKTOP-TIC5DM4 MINGW64 /Myproject (master)

$ git init

Initialized empty Git repository in C:/Program Files/Git/Myproject/.git/

Administrator@DESKTOP-TIC5DM4 MINGW64 /Myproject (master)

$ touch index.html

Administrator@DESKTOP-TIC5DM4 MINGW64 /Myproject (master)

$ ls

index.html

Administrator@DESKTOP-TIC5DM4 MINGW64 /Myproject (master)

$ ls -l

total 0

-rw-r--r-- 1 Administrator 197121 0 May 8 22:03 index.html

Administrator@DESKTOP-TIC5DM4 MINGW64 /Myproject (master)

```
$ vim index.html

Administrator@DESKTOP-TIC5DM4 MINGW64 /Myproject (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        index.html

nothing added to commit but untracked files present (use "git add" to track)

Administrator@DESKTOP-TIC5DM4 MINGW64 /Myproject (master)
$ git add index.html
warning: in the working copy of 'index.html', LF will be replaced by CRLF the next time Git touches it

Administrator@DESKTOP-TIC5DM4 MINGW64 /Myproject (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   index.html


Administrator@DESKTOP-TIC5DM4 MINGW64 /Myproject (master)
$ touch bluestyle.css
```

```
Administrator@DESKTOP-TIC5DM4 MINGW64 /Myproject (master)
$ vim bluestyle.css

Administrator@DESKTOP-TIC5DM4 MINGW64 /Myproject (master)
$ git add --all
warning: in the working copy of 'bluestyle.css', LF will be replaced by CRLF the next time Git touches
it

Administrator@DESKTOP-TIC5DM4 MINGW64 /Myproject (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   bluestyle.css
        new file:   index.html


Administrator@DESKTOP-TIC5DM4 MINGW64 /Myproject (master)
$ git commit -m "Initial Commit"
[master (root-commit) ae9b0d4] Initial Commit
 2 files changed, 19 insertions(+)
 create mode 100644 bluestyle.css
 create mode 100644 index.html

Administrator@DESKTOP-TIC5DM4 MINGW64 /Myproject (master)
$ git log
commit ae9b0d459450607482522fee62aaff02073ccc52 (HEAD -> master)
```

Author: Suneel <vsuneel244@gmail.com>

Date: Wed May 8 23:41:48 2024 +0530

  Initial Commit

Assignment 2:

Branch Creation and Switching Create a new branch named 'feature' and switch to it. Make changes in the 'feature' branch and commit them.

Sol:

Administrator@DESKTOP-TIC5DM4 MINGW64 /Myproject (master)

$ git branch feature

Administrator@DESKTOP-TIC5DM4 MINGW64 /Myproject (master)

$ git checkout feature

Switched to branch 'feature'

Administrator@DESKTOP-TIC5DM4 MINGW64 /Myproject (feature)

$ ls

bluestyle.css index.html

Administrator@DESKTOP-TIC5DM4 MINGW64 /Myproject (feature)

$ vim index.html

Administrator@DESKTOP-TIC5DM4 MINGW64 /Myproject (feature)

$ git add --all

warning: in the working copy of 'index.html', LF will be replaced by CRLF the next time Git touches it

Administrator@DESKTOP-TIC5DM4 MINGW64 /Myproject (feature)

$ git status

On branch feature

Changes to be committed:

(use "git restore --staged <file>..." to unstage)

modified: index.html

Administrator@DESKTOP-TIC5DM4 MINGW64 /Myproject (feature)

$ git commit -m "feature branch"

[feature 59733a4] feature branch

1 file changed, 4 insertions(+), 2 deletions(-)

Administrator@DESKTOP-TIC5DM4 MINGW64 /Myproject (feature)

$ git log

commit 59733a4d5bd21418fa14438a2126a9bf83e36cc3 (HEAD -> feature)

Author: Suneel <vsuneel244@gmail.com>

Date: Thu May 9 00:00:26 2024 +0530

feature branch

commit ae9b0d459450607482522fee62aaff02073ccc52 (master)

Author: Suneel <vsuneel244@gmail.com>

Date: Wed May 8 23:41:48 2024 +0530

Initial Commit

Assignment 3.

Feature Branches and Hotfixes Create a 'hotfix' branch to fix an issue in the main code. Merge the 'hotfix' branch into 'main' ensuring that the issue is resolved

Sol:

Administrator@DESKTOP-TIC5DM4 MINGW64 /Myproject (feature)

$ git branch hotfixer

```
Administrator@DESKTOP-TIC5DM4 MINGW64 /Myproject (feature)
$ git checkout hotfixer
Switched to branch 'hotfixer'

Administrator@DESKTOP-TIC5DM4 MINGW64 /Myproject (hotfixer)
$ ls
bluestyle.css  index.html

Administrator@DESKTOP-TIC5DM4 MINGW64 /Myproject (hotfixer)
$ vim index.html

Administrator@DESKTOP-TIC5DM4 MINGW64 /Myproject (hotfixer)
$ git add --all
warning: in the working copy of 'index.html', LF will be replaced by CRLF the next time Git touches it

Administrator@DESKTOP-TIC5DM4 MINGW64 /Myproject (hotfixer)
$ git status
On branch hotfixer
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   index.html


Administrator@DESKTOP-TIC5DM4 MINGW64 /Myproject (hotfixer)
$ git commit -m "change to fix from Hotfixer Branch"
[hotfixer a4be224] change to fix from Hotfixer Branch
 1 file changed, 1 insertion(+), 1 deletion(-)

Administrator@DESKTOP-TIC5DM4 MINGW64 /Myproject (hotfixer)
$ git checkout master
```

Switched to branch 'master'

Administrator@DESKTOP-TIC5DM4 MINGW64 /Myproject (master)

$ git merge hotfixer

Updating ae9b0d4..a4be224

Fast-forward

index.html | 6 ++++--

1 file changed, 4 insertions(+), 2 deletions(-)

.....Shell Scripting Assignments.....

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".

Sol:

Administrator@DESKTOP-TIC5DM4 MINGW64 /Assignment (master)

$ ls

myfile.txt

Administrator@DESKTOP-TIC5DM4 MINGW64 /Assignment (master)

$ touch checkfile.sh

Administrator@DESKTOP-TIC5DM4 MINGW64 /Assignment (master)

$ ls

checkfile.sh myfile.txt

```
Administrator@DESKTOP-TIC5DM4 MINGW64 /Assignment (master)
$ vim checkfile.sh

Administrator@DESKTOP-TIC5DM4 MINGW64 /Assignment (master)
$ ./checkfile.sh
File is exists.

Administrator@DESKTOP-TIC5DM4 MINGW64 /Assignment (master)
$ ls
checkfile.sh*  myfile.txt

Administrator@DESKTOP-TIC5DM4 MINGW64 /Assignment (master)
$ cat checkfile.sh
#!/bin/bash
if [ -f "myfile.txt" ];
then
    echo "File is exists."
else
    echo "File is not exists."
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.

Sol:

```
Administrator@DESKTOP-TIC5DM4 MINGW64 /Assignment (master)
$ vim assignment2.sh
```

```
Administrator@DESKTOP-TIC5DM4 MINGW64 /Assignment (master)
$ ./assignment2.sh
Enter a number
4
4 is even number
Enter a number
5
 is a odd number
Enter a number
9
 is a odd number
Enter a number
0
```

```
Administrator@DESKTOP-TIC5DM4 MINGW64 /Assignment (master)
$ cat assignment2.sh
#!/bin/bash
a=true
while $a
do
        echo "Enter a number"
        read num
        if(($num != 0))
        then
                if(($num%2==0))
                then
                        echo "$num is even number"
                else
                        echo "$number is a odd number"
                fi
```

```
    else
        a=false
    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.

Sol:

```
Administrator@DESKTOP-TIC5DM4 MINGW64 /Assignment (master)
$ vim assignment3.sh

Administrator@DESKTOP-TIC5DM4 MINGW64 /Assignment (master)
$ ./assignment3.sh
number of lines are :
3 myfile.txt

Administrator@DESKTOP-TIC5DM4 MINGW64 /Assignment (master)
$ cat assignment3.sh
file=myfile.txt
function1() {
    echo "number of lines are : "
    wc -l $1
}
function1 $file
```

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").

Sol:

```
Administrator@DESKTOP-TIC5DM4 MINGW64 /Assignment (master)
$ vim Assignment4.sh

Administrator@DESKTOP-TIC5DM4 MINGW64 /Assignment (master)
$ ./Assignment4.sh

Administrator@DESKTOP-TIC5DM4 MINGW64 /Assignment (master)
$ ls
Assignment4.sh*  TestDir/  assignment2.sh*  assignment3.sh  checkfile.sh*  myfile.txt

Administrator@DESKTOP-TIC5DM4 MINGW64 /Assignment (master)
$ cat Assignment4.sh
#!/bin/bash

#Create the TestDir directory if it does not exist
mkdir -p TestDir

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

Administrator@DESKTOP-TIC5DM4 MINGW64 /Assignment (master)
$ cd TestDir
```

Administrator@DESKTOP-TIC5DM4 MINGW64 /Assignment/TestDir (master)

$ ls

file1.txt file10.txt file2.txt file3.txt file4.txt file5.txt file6.txt file7.txt file8.txt file9.txt

---

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.

Sol:

Administrator@DESKTOP-TIC5DM4 MINGW64 /Assignment (master)

$ ./Assignment5.sh

Administrator@DESKTOP-TIC5DM4 MINGW64 /Assignment (master)

$ cat Assignment5.sh

```
#!/bin/bash

if [ "$DEBUG" = "true" ]; then
    set -x
fi
handleErrors() {
    echo "Error: $1"
    exit 1
}

if [ -d "TestDir" ]; then
    handleErrors "Directory 'TestDir' Alreay exist"
fi

mkdir -p TestDir || handleErrors "Failed to create directory"
```

```
cd TestDir || handleErrors :"Failed to change directory"


for (( i=1; i<=10;i++)); do
    echo "File$i.txt" > "File$i.txt" || handleErrors "Failed to create files 'File$i.txt'."
done
if [ "$DEBUG" = "true" ]; then
    set +x
    fi
```

```
Administrator@DESKTOP-TIC5DM4 MINGW64 /Assignment (master)
$ ls
Assignment4.sh*  Assignment5.sh*  TestDir/  assignment2.sh*  assignment3.sh  checkfile.sh*
myfile.txt

Administrator@DESKTOP-TIC5DM4 MINGW64 /Assignment (master)
$ cd TestDir

Administrator@DESKTOP-TIC5DM4 MINGW64 /Assignment/TestDir (master)
$ ls
File1.txt  File10.txt  File2.txt  File3.txt  File4.txt  File5.txt  File6.txt  File7.txt  File8.txt  File9.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

Sol:

```
Administrator@DESKTOP-TIC5DM4 MINGW64 /Assignment (master)
$ cat sample.log
```

```
2024-05-13T08:30:30 ERROR: Disl full
2024-05-12T04:33:32;Successfull

Administrator@DESKTOP-TIC5DM4 MINGW64 /Assignment (master)
$ ./Assignment6.sh
ERROR: 2024-05-13T08:30:30 ERROR: Disl full

Administrator@DESKTOP-TIC5DM4 MINGW64 /Assignment (master)
$ cat Assignment6.sh
grep "ERROR" sample.log | awk '{print $1, $2, $0}' |
      sed 's/^[^ ]* //;s/ [^ ]* $//'

Administrator@DESKTOP-TIC5DM4 MINGW64 /Assignment (master)
$
```

.....

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.

Sol:

```
Administrator@DESKTOP-TIC5DM4 MINGW64 /Assignment (master)
$ cat Assignment7.sh
 if  [ $# -ne 3 ]; then
        echo "Usase : $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 '$input_file' not found."
    exit 1
fi
sed "s/$old_text/$new_text/g" "$input_file" > "${input_file}_modified"
echo "Replacement done.Modified content saved to'${input_file}_modified'."
```