Git Hands-on lab

Install Git

Determine if Git is already installed on your computer by opening a terminal and running this command:

```
git --version
```

If Git is installed, the output is:

```
git version X.Y.Z
```

If your computer doesn't recognize git as a command, you must install Git. After you install Git, run git --version to confirm that it installed correctly.

Configure Git

To start using Git from your computer, you must enter your credentials to identify yourself as the author of your work.

In your command terminal, add your user name:

```
git config --global user.name "your_username"
```

Add your email address:

```
git config --global user.email "your_email_address@example.com"
```

To check the configuration, run:

```
git config --global --list
```

Create a local repository

1. Create a local folder (work directory) in C or D drive.
2. From the command terminal, go to the above created directory.
3. You can initialize a local folder so Git tracks it as a repository.

```
git init
```

A .git folder is created in your directory. This folder contains Git records and configuration files. You should not edit these files directly.

To check the status of your git local repository

```
git status
```

Add and commit local changes

1. Create a file named README.txt in the work directory.
2. When you type git status, locally changed files are shown in red. These changes may be new, modified, or deleted files or folders.
3. To stage a file for commit:

   ```
   git add <file-name OR folder-name>
   ```

   To all files - git add .
   To add only updated files - git add -u

4. Confirm that the files have been added to staging:

   ```
   git status
   ```

   The files should be displayed in green text.

5. To commit the staged files:

   ```
   git commit -m "COMMENT TO DESCRIBE THE INTENTION OF THE COMMIT"
   ```

6. Check the log

   ```
   git log
   ```

7. Get help for log

   ```
   git log --help
   ```

8. To see the log in one line

   ```
   git log --oneline
   ```

9. View the difference between two commits

   ```
   git diff <<commitid>>..<commitid>>
   ```

10. View the difference between latest commit and previous commit using HEAD

    ```
    git diff HEAD.. HEAD~1
    ```