```
MINGW64:/c/Users/shrey/desktop/Git

shrey@MSI MINGW64 ~
$ git --version
git version 2.47.0.windows.1

shrey@MSI MINGW64 ~
$ git config --global user.name "Shreyaa25"

shrey@MSI MINGW64 ~
$ git config --global user.email "shreya.17349@sakec.ac.in"

shrey@MSI MINGW64 ~
$ git config --list
diff.astextplain.textconv=astextplain
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
http.sslbackend=openssl
http.sslcainfo=C:/Program Files/Git/mingw64/etc/ssl/certs/ca-bundle.crt
core.autocrlf=true
core.fscache=true
core.symlinks=false
pull.rebase=false
credential.helper=manager
credential.https://dev.azure.com.usehttppath=true
init.defaultbranch=master
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
user.name=Shreyaa25
user.email=shreya.17349@sakec.ac.in

shrey@MSI MINGW64 ~
$ C:\Users\shrey\Desktop\Git
bash: C:UsersshreyDesktopGit: command not found

shrey@MSI MINGW64 ~
$ cd C:\Users\shrey\Desktop\Git
bash: cd: C:UsersshreyDesktopGit: No such file or directory

shrey@MSI MINGW64 ~
$ cd desktop

shrey@MSI MINGW64 ~/desktop
$ cd Git

shrey@MSI MINGW64 ~/desktop/Git
$ git clone https://github.com/Shreyaa25/practical.git
Cloning into 'practical'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (3/3), done.
```

```
shrey@MSI MINGW64 ~/desktop/Git
$ cd practical

shrey@MSI MINGW64 ~/desktop/Git/practical (main)
$ ls
README.md

shrey@MSI MINGW64 ~/desktop/Git/practical (main)
$ ls -a
./   ../   .git/   README.md

shrey@MSI MINGW64 ~/desktop/Git/practical (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   README.md

no changes added to commit (use "git add" and/or "git commit -a")

shrey@MSI MINGW64 ~/desktop/Git/practical (main)
$ git add .

shrey@MSI MINGW64 ~/desktop/Git/practical (main)
$ git commit -m "Changed"
[main 55a546a] Changed
 1 file changed, 2 insertions(+), 1 deletion(-)

shrey@MSI MINGW64 ~/desktop/Git/practical (main)
$ git push origin main
info: please complete authentication in your browser...
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Writing objects: 100% (3/3), 276 bytes | 276.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/Shreyaa25/practical.git
   ba003af..55a546a  main -> main

shrey@MSI MINGW64 ~/desktop/Git/practical (main)
$ git push origin main
Everything up-to-date

shrey@MSI MINGW64 ~/desktop/Git/practical (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   README.md

no changes added to commit (use "git add" and/or "git commit -a")
```

```
no changes added to commit (use "git add" and/or "git commit -a")

shrey@MSI MINGW64 ~/desktop/Git/practical (main)
$ git add .

shrey@MSI MINGW64 ~/desktop/Git/practical (main)
$ git commit -m "Changed"
[main 9af951b] Changed
 1 file changed, 2 insertions(+), 1 deletion(-)

shrey@MSI MINGW64 ~/desktop/Git/practical (main)
$ git push origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Writing objects: 100% (3/3), 283 bytes | 141.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/Shreyaa25/practical.git
   55a546a..9af951b  main -> main

shrey@MSI MINGW64 ~/desktop/Git/practical (main)
$ touch shreya.txt

shrey@MSI MINGW64 ~/desktop/Git/practical (main)
$ cat >>Shreya.txt
Hello Iam shreya


shrey@MSI MINGW64 ~/desktop/Git/practical (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

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

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

shrey@MSI MINGW64 ~/desktop/Git/practical (main)
$ git add .
warning: in the working copy of 'shreya.txt', LF will be replaced by CRLF the next time Git touc
hes it

shrey@MSI MINGW64 ~/desktop/Git/practical (main)
$ git commit -m "New file"
[main f909577] New file
 1 file changed, 3 insertions(+)
 create mode 100644 shreya.txt

shrey@MSI MINGW64 ~/desktop/Git/practical (main)
$ git push origin main
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 12 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 321 bytes | 321.00 KiB/s, done.
```

```
$ ls -a
./  ../  shreya.txt

shrey@MSI MINGW64 ~/desktop/Git/Repo
$ git init
Initialized empty Git repository in C:/Users/shrey/Desktop/Git/Repo/.git/

shrey@MSI MINGW64 ~/desktop/Git/Repo (master)
$ ls -a
./  ../  .git/  shreya.txt

shrey@MSI MINGW64 ~/desktop/Git/Repo (master)
$ git status
On branch master

No commits yet

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

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

shrey@MSI MINGW64 ~/desktop/Git/Repo (master)
$ git add .
warning: in the working copy of 'shreya.txt', LF will be replaced by CRLF the next time Git touches it

shrey@MSI MINGW64 ~/desktop/Git/Repo (master)
$ git commit -m "Added"
[master (root-commit) 5d718ae] Added
 1 file changed, 1 insertion(+)
 create mode 100644 shreya.txt

shrey@MSI MINGW64 ~/desktop/Git/Repo (master)
$ git remote add origin https://github.com/Shreyaa25/pract.git

shrey@MSI MINGW64 ~/desktop/Git/Repo (master)
$ git push -u origin main
error: src refspec main does not match any
error: failed to push some refs to 'https://github.com/Shreyaa25/pract.git'

shrey@MSI MINGW64 ~/desktop/Git/Repo (master)
$ git push origin main
error: src refspec main does not match any
error: failed to push some refs to 'https://github.com/Shreyaa25/pract.git'

shrey@MSI MINGW64 ~/desktop/Git/Repo (master)
$ git remote add origin https://github.com/Shreyaa25/pract.git
error: remote origin already exists.

shrey@MSI MINGW64 ~/desktop/Git/Repo (master)
$ git push origin main
error: src refspec main does not match any
error: failed to push some refs to 'https://github.com/Shreyaa25/pract.git'

shrey@MSI MINGW64 ~/desktop/Git/Repo (master)
$
```

```
MINGW64:/c/Users/shrey/desktop/Git/rep                                    —    □    ✕

shrey@MSI MINGW64 ~/desktop/Git/Repo (master)
$ ls -a
./  ../  .git/  shreya.txt

shrey@MSI MINGW64 ~/desktop/Git/Repo (master)
$ git status
On branch master

No commits yet

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

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

shrey@MSI MINGW64 ~/desktop/Git/Repo (master)
$ git add .
warning: in the working copy of 'shreya.txt', LF will be replaced by CRLF the next time Git touches it

shrey@MSI MINGW64 ~/desktop/Git/Repo (master)
$ git commit -m "Added"
[master (root-commit) 5d718ae] Added
 1 file changed, 1 insertion(+)
 create mode 100644 shreya.txt

shrey@MSI MINGW64 ~/desktop/Git/Repo (master)
$ git remote add origin https://github.com/Shreyaa25/pract.git

shrey@MSI MINGW64 ~/desktop/Git/Repo (master)
$ git push -u origin main
error: src refspec main does not match any
error: failed to push some refs to 'https://github.com/Shreyaa25/pract.git'

shrey@MSI MINGW64 ~/desktop/Git/Repo (master)
$ git push origin main
error: src refspec main does not match any
error: failed to push some refs to 'https://github.com/Shreyaa25/pract.git'

shrey@MSI MINGW64 ~/desktop/Git/Repo (master)
$ git remote add origin https://github.com/Shreyaa25/pract.git
error: remote origin already exists.

shrey@MSI MINGW64 ~/desktop/Git/Repo (master)
$ git push origin main
error: src refspec main does not match any
error: failed to push some refs to 'https://github.com/Shreyaa25/pract.git'

shrey@MSI MINGW64 ~/desktop/Git/Repo (master)
$ git branch
* master

shrey@MSI MINGW64 ~/desktop/Git/Repo (master)
$ git branch -M main

shrey@MSI MINGW64 ~/desktop/Git/Repo (main)
```

```
$ git branch -M main

shrey@MSI MINGW64 ~/desktop/Git/Repo (main)
$ git branch
* main

shrey@MSI MINGW64 ~/desktop/Git/Repo (main)
$ git checkout -b newbranch
Switched to a new branch 'newbranch'

shrey@MSI MINGW64 ~/desktop/Git/Repo (newbranch)
$ git branch
  main
* newbranch

shrey@MSI MINGW64 ~/desktop/Git/Repo (newbranch)
$ git checkout main
Switched to branch 'main'

shrey@MSI MINGW64 ~/desktop/Git/Repo (main)
$ git init
Reinitialized existing Git repository in C:/Users/shrey/Desktop/Git/Repo/.git/

shrey@MSI MINGW64 ~/desktop/Git/Repo (main)
$ git status
On branch main
nothing to commit, working tree clean

shrey@MSI MINGW64 ~/desktop/Git/Repo (main)
$ git checkout newbranch
Switched to branch 'newbranch'

shrey@MSI MINGW64 ~/desktop/Git/Repo (newbranch)
$ cat >> shreya.txt
I hate me

shrey@MSI MINGW64 ~/desktop/Git/Repo (newbranch)
$ cd ..

shrey@MSI MINGW64 ~/desktop/Git
$ mkdir rep

shrey@MSI MINGW64 ~/desktop/Git
$ touch ss.txt

shrey@MSI MINGW64 ~/desktop/Git
$ cat >> ss.txt
I am shreya

shrey@MSI MINGW64 ~/desktop/Git
$ ls
Repo/  practical/  rep/  ss.txt

shrey@MSI MINGW64 ~/desktop/Git
$ ls -a
./  ../  Repo/  practical/  rep/  ss.txt
```

```
MINGW64:/c/Users/shrey/desktop/Git/rep                                    ─    □

$ ls
ss.txt

shrey@MSI MINGW64 ~/desktop/Git/rep
$ ls -a
./  ../  ss.txt

shrey@MSI MINGW64 ~/desktop/Git/rep
$ git init
Initialized empty Git repository in C:/Users/shrey/Desktop/Git/rep/.git/

shrey@MSI MINGW64 ~/desktop/Git/rep (master)
$ git status
On branch master

No commits yet

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

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

shrey@MSI MINGW64 ~/desktop/Git/rep (master)
$ git branch -M main

shrey@MSI MINGW64 ~/desktop/Git/rep (main)
$ git add .
warning: in the working copy of 'ss.txt', LF will be replaced by CRLF the next time Git touches it

shrey@MSI MINGW64 ~/desktop/Git/rep (main)
$  git commit -m "Added"
[main (root-commit) 66d50a2] Added
 1 file changed, 1 insertion(+)
 create mode 100644 ss.txt

shrey@MSI MINGW64 ~/desktop/Git/rep (main)
$ git remote add origin https://github.com/Shreyaa25/pract.git

shrey@MSI MINGW64 ~/desktop/Git/rep (main)
$ git push origin main
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 221 bytes | 110.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/Shreyaa25/pract.git
 * [new branch]      main -> main

shrey@MSI MINGW64 ~/desktop/Git/rep (main)
$ git branch
* main

shrey@MSI MINGW64 ~/desktop/Git/rep (main)
$ git checkout -b newbranch
Switched to a new branch 'newbranch'

shrey@MSI MINGW64 ~/desktop/Git/rep (newbranch)
```

```
shrey@MSI MINGW64 ~/desktop/Git/rep (main)
$ git checkout -b newbranch
Switched to a new branch 'newbranch'

shrey@MSI MINGW64 ~/desktop/Git/rep (newbranch)
$ cat >> ss.txt
I hate myself


shrey@MSI MINGW64 ~/desktop/Git/rep (newbranch)
$ git checkout main
M       ss.txt
Switched to branch 'main'

shrey@MSI MINGW64 ~/desktop/Git/rep (main)
$ git status
On branch main
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   ss.txt

no changes added to commit (use "git add" and/or "git commit -a")

shrey@MSI MINGW64 ~/desktop/Git/rep (main)
$ git add .
warning: in the working copy of 'ss.txt', LF will be replaced by CRLF the next time Git touches it

shrey@MSI MINGW64 ~/desktop/Git/rep (main)
$ git status
On branch main
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   ss.txt


shrey@MSI MINGW64 ~/desktop/Git/rep (main)
$ git commit -m "New line"
[main 1005fdf] New line
 1 file changed, 1 insertion(+)

shrey@MSI MINGW64 ~/desktop/Git/rep (main)
$ git checkout newbranch
Switched to branch 'newbranch'

shrey@MSI MINGW64 ~/desktop/Git/rep (newbranch)
$ git add .

shrey@MSI MINGW64 ~/desktop/Git/rep (newbranch)
$ git status
On branch newbranch
nothing to commit, working tree clean

shrey@MSI MINGW64 ~/desktop/Git/rep (newbranch)
$ git push origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
```

```
MINGW64:/c/Users/shrey/desktop/Git/rep                                    —    □

$ git push origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Writing objects: 100% (3/3), 267 bytes | 133.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/Shreyaa25/pract.git
   66d50a2..1005fdf  main -> main

shrey@MSI MINGW64 ~/desktop/Git/rep (newbranch)
$ git checkout main
Switched to branch 'main'

shrey@MSI MINGW64 ~/desktop/Git/rep (main)
$ git checkout newbranch
Switched to branch 'newbranch'

shrey@MSI MINGW64 ~/desktop/Git/rep (newbranch)
$ cat >> ss.txt
Pranit is a good boy


shrey@MSI MINGW64 ~/desktop/Git/rep (newbranch)
$ git add .
warning: in the working copy of 'ss.txt', LF will be replaced by CRLF the next time Git touches it

shrey@MSI MINGW64 ~/desktop/Git/rep (newbranch)
$ git status
On branch newbranch
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   ss.txt


shrey@MSI MINGW64 ~/desktop/Git/rep (newbranch)
$ git commit -m "New lines"
[newbranch 2ae5e5a] New lines
 1 file changed, 1 insertion(+)

shrey@MSI MINGW64 ~/desktop/Git/rep (newbranch)
$ git push origin newbranch
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Writing objects: 100% (3/3), 276 bytes | 138.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'newbranch' on GitHub by visiting:
remote:      https://github.com/Shreyaa25/pract/pull/new/newbranch
remote:
To https://github.com/Shreyaa25/pract.git
 * [new branch]      newbranch -> newbranch

shrey@MSI MINGW64 ~/desktop/Git/rep (newbranch)
$ git diff main
diff --git a/ss.txt b/ss.txt
index 42f325d..edce3c3 100644
--- a/ss.txt
+++ b/ss.txt
```

```
shrey@MSI MINGW64 ~/desktop/Git/rep (newbranch)
$ git diff main
diff --git a/ss.txt b/ss.txt
index 42f325d..edce3c3 100644
--- a/ss.txt
+++ b/ss.txt
@@ -1,2 +1,2 @@
 I am Shreya
-I hate myself
+Pranit is a good boy

shrey@MSI MINGW64 ~/desktop/Git/rep (newbranch)
$ git checkout main
Switched to branch 'main'

shrey@MSI MINGW64 ~/desktop/Git/rep (main)
$ git pull origin main
From https://github.com/Shreyaa25/pract
 * branch            main          -> FETCH_HEAD
Already up to date.

shrey@MSI MINGW64 ~/desktop/Git/rep (main)
$
```

1. Git Basics

- **Initialize a Git repository**

git init

2. Git Add

- **Stage files for commit**

git add <file_name>   # Adds a specific file

git add .   # Stages all changed files in the current directory

3. Git Commit

- **Commit staged changes**

git commit -m "Commit message"   # Commits with a message

git commit -am "Commit message"   # Adds and commits in one step for tracked files

4. Git Push

- **Push changes to a remote repository**

git push origin <branch_name>   # Pushes your changes to the remote branch

git push -u origin <branch_name> # Sets the default remote and branch for future pushes

5. Git Pull

- **Fetch and integrate changes from a remote repository**

git pull origin <branch_name>   # Pulls changes from the remote branch

6. Creating User Profile

- **Set global Git user name and email**

git config --global user.name "Your Name"

git config --global user.email "youremail@example.com"

- **Unset global Git user and email**

git config --global --unset user.name

git config --global --unset user.email

7. Pull Remote Repo to Local

- **Clone a repository**

git clone <repository_url>   # Clones the remote repository to your local machine

- **Update your local repository with changes from remote**

git pull   # Fetch and merge changes from the remote repository

8. Version Control System in Git and GitHub

Git provides version control through commits and branches:

- **Check the current status of the repository**

git status

- **View commit history**

git log

- **Switch between branches**

git checkout <branch_name>

- **Create a new branch**

git branch <branch_name>

9. Git Merge

- **Merge changes from one branch into another**

git merge <branch_name>   # Merges the specified branch into your current branch

10. Git Fork

- **Forking a repository is done on GitHub**. Once you've forked a repository, you can clone it:

git clone <forked_repo_url>   # Clones your forked repo from GitHub to your local machine

11. Git Delete Commands

- **Delete a branch (local)**

git branch -d <branch_name>   # Deletes the branch locally (only if merged)

git branch -D <branch_name>   # Force deletes the branch (even if not merged)

- **Delete a branch (remote)**

git push origin --delete <branch_name>   # Deletes the branch on the remote repository

- **Delete a file**

git rm <file_name>   # Removes the file from the working directory and stages the change

- **Remove untracked files**

git clean -f   # Deletes untracked files from the working directory

git clean -fd   # Deletes untracked files and directories

12. Git Reset & Revert

- **Undo the last commit (keep changes)**

git reset --soft HEAD^

- **Undo the last commit (remove changes)**

git reset --hard HEAD^

- **Revert a specific commit**

git revert <commit_hash>   # Creates a new commit that undoes the changes from the specified commit

13. Unset Commands and Clean-Up

- **Unset global configuration**

git config --global --unset <config_name>

- **Delete local git repository (remove .git directory)**

rm -rf .git   # Deletes the Git repository from the current directory

- **Clear Git log files**

git reflog expire --expire=now --all

git gc --prune=now

This list covers most of the commonly used Git and GitHub commands for adding, committing, pushing, pulling, merging, and deleting. Let me know if you need more details on any of these topics!

How to merge

Neev@DESKTOP-FA48JBH MINGW64 ~/testGit (master)

$ git status

On branch master

Changes to be committed:

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

    new file: index.txt

Neev@DESKTOP-FA48JBH MINGW64 ~/testGit (master)

$ git commit -m "New branch and index.txt file addition."

[master bdd9cab] New branch and index.txt file addition.

 1 file changed, 1 insertion(+)

 create mode 100644 index.txt

Neev@DESKTOP-FA48JBH MINGW64 ~/testGit (master)

$ git push origin master

Enumerating objects: 4, done.

Counting objects: 100% (4/4), done.

Delta compression using up to 20 threads

Compressing objects: 100% (2/2), done.

Writing objects: 100% (3/3), 379 bytes | 379.00 KiB/s, done.

Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)

remote:

remote: Create a pull request for 'master' on GitHub by visiting:

remote: https://github.com/Alpha0705/testGit/pull/new/master

remote:

To https://github.com/Alpha0705/testGit.git

 * [new branch]   master -> master

Neev@DESKTOP-FA48JBH MINGW64 ~/testGit (master)

$ git checkout main

Switched to branch 'main'

Neev@DESKTOP-FA48JBH MINGW64 ~/testGit (main)

$ git merge master

Updating 3b5460e..bdd9cab

Fast-forward

 index.txt | 1 +

 1 file changed, 1 insertion(+)

 create mode 100644 index.txt

Neev@DESKTOP-FA48JBH MINGW64 ~/testGit (main)

$ git status

On branch main

nothing to commit, working tree clean

Neev@DESKTOP-FA48JBH MINGW64 ~/testGit (main)

$ git add .

Neev@DESKTOP-FA48JBH MINGW64 ~/testGit (main)

$ git push origin main

Total 0 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)

To https://github.com/Alpha0705/testGit.git

   3b5460e..bdd9cab main -> main

```
MINGW64:/c/Users/Neev/test          +  ⌄                                              —  □  ✕

Neev@DESKTOP-FA48JBH MINGW64 ~/testGit (master)
$ git push origin master
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 20 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 379 bytes | 379.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'master' on GitHub by visiting:
remote:      https://github.com/Alpha0705/testGit/pull/new/master
remote:
To https://github.com/Alpha0705/testGit.git
 * [new branch]      master -> master

Neev@DESKTOP-FA48JBH MINGW64 ~/testGit (master)
$ git merge main
Already up to date.

Neev@DESKTOP-FA48JBH MINGW64 ~/testGit (master)
$ git merge master main
Already up to date.

Neev@DESKTOP-FA48JBH MINGW64 ~/testGit (master)
$ git checkout main
Switched to branch 'main'

Neev@DESKTOP-FA48JBH MINGW64 ~/testGit (main)
$ git merge master
Updating 3b5460e..bdd9cab
Fast-forward
 index.txt | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 index.txt

Neev@DESKTOP-FA48JBH MINGW64 ~/testGit (main)
$ git status
On branch main
nothing to commit, working tree clean

Neev@DESKTOP-FA48JBH MINGW64 ~/testGit (main)
$ git add .

Neev@DESKTOP-FA48JBH MINGW64 ~/testGit (main)
```

```
MINGW64:/c/Users/Neev/test          +  ⌄                                              —  □  ✕

Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'master' on GitHub by visiting:
remote:      https://github.com/Alpha0705/testGit/pull/new/master
remote:
To https://github.com/Alpha0705/testGit.git
 * [new branch]      master -> master

Neev@DESKTOP-FA48JBH MINGW64 ~/testGit (master)
$ git merge main
Already up to date.

Neev@DESKTOP-FA48JBH MINGW64 ~/testGit (master)
$ git merge master main
Already up to date.

Neev@DESKTOP-FA48JBH MINGW64 ~/testGit (master)
$ git checkout main
Switched to branch 'main'

Neev@DESKTOP-FA48JBH MINGW64 ~/testGit (main)
$ git merge master
Updating 3b5460e..bdd9cab
Fast-forward
 index.txt | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 index.txt

Neev@DESKTOP-FA48JBH MINGW64 ~/testGit (main)
$ git status
On branch main
nothing to commit, working tree clean

Neev@DESKTOP-FA48JBH MINGW64 ~/testGit (main)
$ git add .

Neev@DESKTOP-FA48JBH MINGW64 ~/testGit (main)
$ git push origin main
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/Alpha0705/testGit.git
   3b5460e..bdd9cab  main -> main

Neev@DESKTOP-FA48JBH MINGW64 ~/testGit (main)
$ |
```

Theory

Git and GitHub Commands Overview

1. Version Control System (VCS)

A **Version Control System (VCS)** is a system that tracks changes in source code or other collections of files, allowing multiple users to collaborate, view history, and revert changes. Git is a **Distributed Version Control System (DVCS)**, meaning every user has a full copy of the repository history, making collaboration flexible and efficient.

2. Git Initialization

- **Command:** git init

- **Purpose:** Initializes a new Git repository in the current directory, creating a .git directory to store repository history and metadata.

- **Usage:** Run this command inside any folder to begin version control.

 git init

3. Staging and Adding Files

- **Command:** git add <file_name> or git add .

- **Purpose:** Moves files from the working directory to the staging area (also called the index), preparing them for the next commit. You can stage specific files or all changes in the directory.

- **Usage:**

 git add myfile.txt   # Add specific file

 git add .   # Add all changes in the directory

4. Committing Changes

- **Command:** git commit -m "Commit message"

- **Purpose:** Records the changes staged in the repository's history by creating a new commit. Each commit represents a snapshot of the project at a given time.

- **Usage:**

 git commit -m "Initial commit"

- **Flags:**

- -m: Allows you to add a commit message directly.

5. Pushing to a Remote Repository

- **Command:** git push origin <branch_name>

- **Purpose:** Sends committed changes from the local repository to a remote repository on platforms like GitHub. You need to specify the remote (origin) and the branch to push to (e.g., main, master).

- **Usage:**

git push origin main

6. Pulling Changes from a Remote Repository

- **Command:** git pull origin <branch_name>

- **Purpose:** Fetches changes from the remote repository and integrates them into the current branch. This command combines git fetch (downloading changes) and git merge (applying changes).

- **Usage:**

git pull origin main

7. Branching

- **Command:** git branch <branch_name>

- **Purpose:** Creates a new branch. Branches allow developers to work on different features or fixes simultaneously without affecting the main or master branch.

- **Usage:**

git branch feature-branch

8. Switching Branches

- **Command:** git checkout <branch_name>

- **Purpose:** Switches to the specified branch in the repository. It updates the working directory to reflect the state of the selected branch.

- **Usage:**

git checkout main

9. Merging Branches

- **Command:** git merge <branch_name>

- **Purpose:** Combines changes from the specified branch into the current branch. For example, merging master into main incorporates the changes from master into main.

- **Usage:**

git merge master

10. Cloning a Repository

- **Command:** git clone <repository_url>

- **Purpose:** Creates a local copy of a remote Git repository, allowing you to contribute to the project. Cloning brings down the entire project history.

- **Usage:**

git clone https://github.com/user/repository.git

11. Forking a Repository (on GitHub)

- **Forking** is not a command you run locally but an action you perform on GitHub. When you fork a repository, it creates a copy of the repository under your GitHub account, allowing you to contribute to it independently.

- **Purpose:** Forking allows you to make changes and later request them to be merged back via a **pull request**.

12. Git Status

- **Command:** git status

- **Purpose:** Shows the status of the working directory and staging area. It lists the files that are staged for the next commit, files with changes not staged, and untracked files.

- **Usage:**

git status

13. Git Log

- **Command:** git log

- **Purpose:** Displays the commit history for the current branch, showing commit messages, authors, and timestamps. Useful for understanding the history of changes.

- **Usage:**

git log

14. Undoing Changes

- **Command:** git reset --soft, git reset --hard, git revert

- **Purpose:** These commands allow you to undo changes:

- git reset --soft: Reverts the last commit, but keeps the changes in the staging area.

- git reset --hard: Reverts the last commit and discards all changes.

- git revert <commit_hash>: Creates a new commit that undoes the changes of the specified commit.

- **Usage:**

git reset --soft HEAD^

git reset --hard HEAD^

git revert <commit_hash>

15. Removing Files

- **Command:** git rm <file_name>

- **Purpose:** Removes a file from both the working directory and the staging area. When committed, the file is deleted from the repository.

- **Usage:**

git rm myfile.txt

16. Unstaging Files

- **Command:** git reset <file_name>

- **Purpose:** Removes a file from the staging area but leaves the file in the working directory.

- **Usage:**

git reset myfile.txt

17. Deleting Branches

- **Command:** git branch -d <branch_name> or git branch -D <branch_name>

- **Purpose:** Deletes a branch from the local repository.

- -d: Deletes the branch only if it has been fully merged.

- -D: Force deletes the branch, even if it hasn't been merged.

- **Usage:**

git branch -d feature-branch

git branch -D feature-branch

18. Setting Global Username and Email

- **Command:** git config --global user.name and git config --global user.email

- **Purpose:** Sets your name and email globally in Git, which is attached to every commit you make.

- **Usage:**

git config --global user.name "Your Name"

git config --global user.email "your.email@example.com"

- **Unset global username and email**

git config --global --unset user.name

git config --global --unset user.email

19. Git Clean

- **Command:** git clean -f or git clean -fd

- **Purpose:** Removes untracked files and directories from the working directory.

- -f: Force clean.

- -d: Also clean untracked directories.

- **Usage:**

git clean -f

git clean -fd

20. Removing Global Configurations

- **Command:** git config --global --unset

- **Purpose:** Removes global configurations like username or email.

- **Usage:**

git config --global --unset user.name

git config --global --unset user.email

Version Control System in Git and GitHub

Git uses a **Distributed Version Control System (DVCS)**. Each contributor has a full copy of the project's history and can commit changes offline. It enables version tracking, collaboration, and multiple users working on the same project. GitHub is a cloud-based Git repository hosting service, adding features like pull requests, issue tracking, and project management.

Main Features of Version Control with Git and GitHub:

1. **Branching and Merging**: Multiple users can work on the same project simultaneously by creating branches. Once the work is done, changes can be merged back into the main branch.

2. **Commits**: Each snapshot of the project history is a **commit** with a unique ID, allowing you to revert to any point in the project's history.

3. **Collaboration**: GitHub enables collaboration by allowing multiple users to contribute, fork, and open pull requests to suggest changes.

4. **History and Rollback**: Git tracks every change, allowing you to see who made which changes and revert any mistakes.

Git and GitHub provide powerful tools for managing codebases, tracking changes, and collaborating effectively across teams.