Ex. No. 6	IMPLEMENTATION OF CORE REVIEW IN GIT
Date of Exercise	11/03/2024

AIM

To implement CORE review in GIT .

Use a shared repository

Using a shared repository is the foundation of effective collaboration withGit. By hosting your repository on a remote server, you can give your team members access to the latest version of your code, regardless of their location.

- Create a new repository: git init
- Clone a repository: git clone [respository URL]
- Add a remote repository: git remote add [name] [repository URL]
- Fetch changes from a remote repository: git fetch
- Pull changes from a remote repository: git pull
- Push changes to a remote repository: git push

Use branching and merging

Branching and merging are powerful features of Git that allow you to work on multiple versions of your code simultaneously, without interfering with each other's work. By using branching and merging effectively, your team can work on different features or bug fixes in parallel, without causing conflicts.

- Create a new branch: git branch [name]
- Switch to a branch: git checkout [name]
- Merge a branch: git merge [name]

Use Pull requests

Pull requests are a great way to review and merge changes from different team members before they are merged into the main codebase. By using pull requests, you can ensure that your code is reviewed and tested before it is merged into the main branch, which can help prevent bugs and other issues.

- Create a new pull request: git request-pull [branch] [repository URL]
- Review and merge a pull request: git pull-request
- Close a pull request: git request-pull -C [branch] [repository URL]

Use Issue Tracking

Issue tracking is a great way to keep track of bugs, feature requests, and other issues that need to be addressed in your code. By using an issue tracking system like GitHub Issues, you can assign tasks to team members, track progress, and ensure that all issues are addressed in a timely manner.

- Create a new issue
- Assign an issue to a team member
- Clone an issue
- Reopen an issue

Effective communication is key to collaboration, especially in a remote team setting. Use tools like Slack or Microsoft Teams to stay in touch with yourteam members, and use video calls or screen sharing to discuss code changes or work on problems together.

- Start a video call: git video-call [username]
- Share your screen: git screen-share
- Send a message: git message [username] [message]

Use Git hooks

Git hooks are scripts that Git runs automatically at certain points in the Git workflow. You can use Git hooks to automate repetitive tasks, enforce coding standards, or perform other tasks that are important to your team's workflow.

- Install a git hook: git init [hook name]
- Write a git hook script: nano .git/hooks/[hook name]
- Make the Git hook script executable: chmod +x .git/hooks/[hook name]

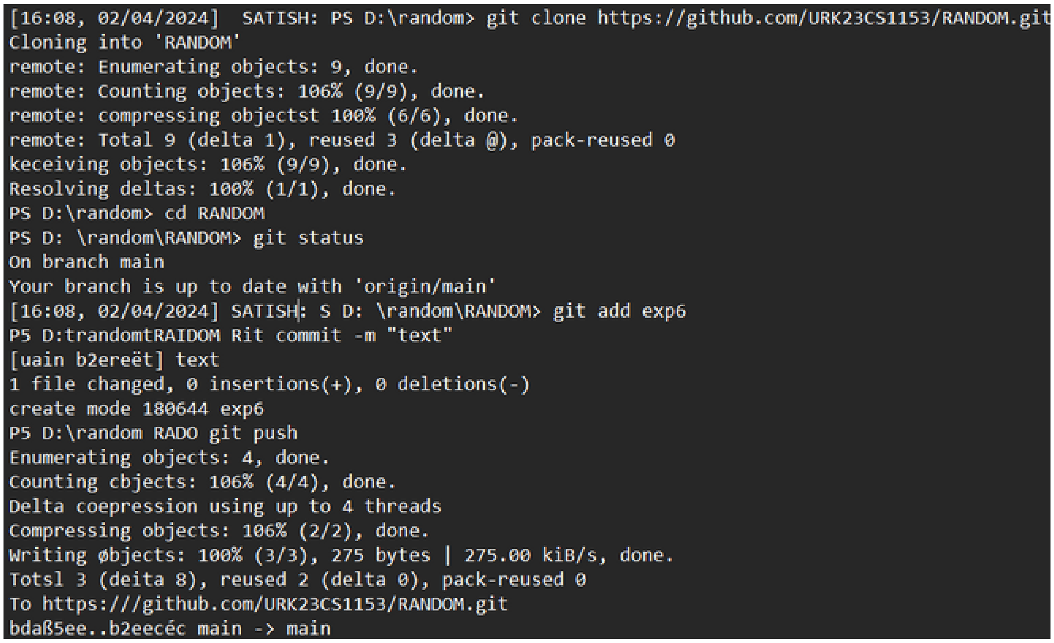
Use Git submodules

Git submodules are repositories that are embedded inside other repositories. You can use Git submodules to manage dependencies, or to include shared code in multiple projects.

Add a Git submodule: git submodule add [repository URL]

Update a Git submodule: gitsubmodule update

Remove a Git submodule: git submodule deinit [submodule path]



```
[16:08, 02/04/2024]  SATISH: PS D:\random> git clone https://github.com/URK23CS1153/RANDOM.git
Cloning into 'RANDOM'
remote: Enumerating objects: 9, done.
remote: Counting objects: 106% (9/9), done.
remote: compressing objectst 100% (6/6), done.
remote: Total 9 (delta 1), reused 3 (delta @), pack-reused 0
keceiving objects: 106% (9/9), done.
Resolving deltas: 100% (1/1), done.
PS D:\random> cd RANDOM
PS D: \random\RANDOM> git status
On branch main
Your branch is up to date with 'origin/main'
[16:08, 02/04/2024] SATISH: S D: \random\RANDOM> git add exp6
P5 D:trandomtRAIDOM Rit commit -m "text"
[uain b2ereët] text
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 180644 exp6
P5 D:\random RADO git push
Enumerating objects: 4, done.
Counting cbjects: 106% (4/4), done.
Delta coepression using up to 4 threads
Compressing objects: 106% (2/2), done.
Writing øbjects: 100% (3/3), 275 bytes | 275.00 kiB/s, done.
Totsl 3 (deita 8), reused 2 (delta 0), pack-reused 0
To https:///github.com/URK23CS1153/RANDOM.git
bdaß5ee..b2eecéc main -> main
```

Use Git aliases

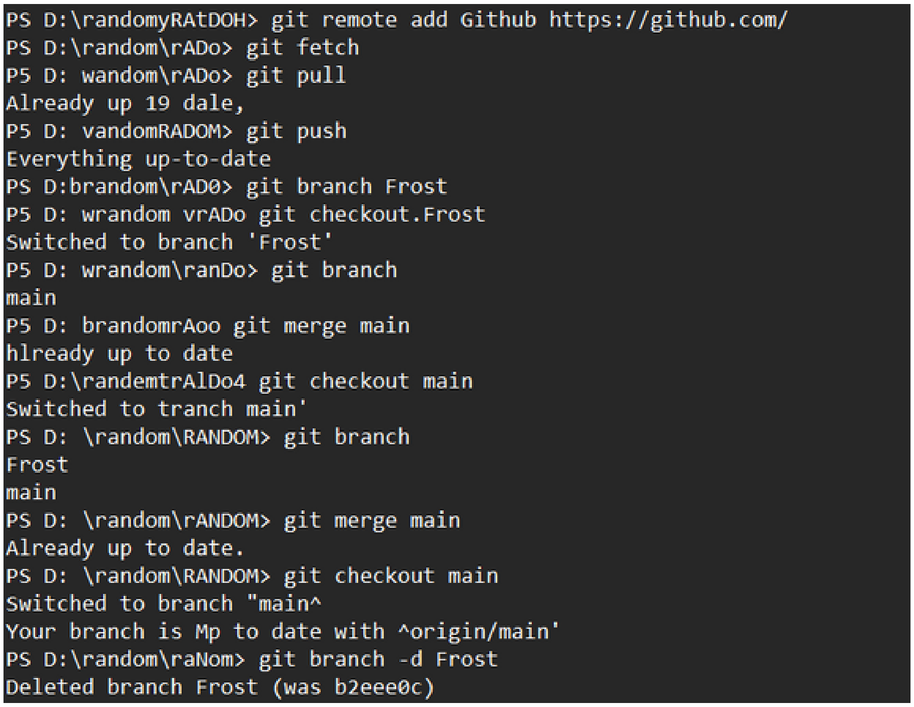
Git aliases are shortcuts for commonly used Git commands. You can use Git aliases to save time and improve your productivity when working with Git.

Set up a Git alias: git config –global alias.[alias name] '[Git command]'

Use a Git alias: git alias [alias name]

conflicts: git merge tool

Delete a branch: git branch -d [name]



```
PS D:\randomyRAtDOH> git remote add Github https://github.com/
PS D:\random\rADo> git fetch
P5 D: wandom\rADo> git pull
Already up 19 dale,
P5 D: vandomRADOM> git push
Everything up-to-date
PS D:brandom\rAD0> git branch Frost
P5 D: wrandom vrADo git checkout.Frost
Switched to branch 'Frost'
P5 D: wrandom\ranDo> git branch
main
P5 D: brandomrAoo git merge main
hlready up to date
P5 D:\randemtrAlDo4 git checkout main
Switched to tranch main'
PS D: \random\RANDOM> git branch
Frost
main
PS D: \random\rANDOM> git merge main
Already up to date.
PS D: \random\RANDOM> git checkout main
Switched to branch "main^
Your branch is Mp to date with ^origin/main'
PS D:\random\raNom> git branch -d Frost
Deleted branch Frost (was b2eee0c)
```

Result:

The program to implement core review in GIT was completed successfully .