Version Controlling

=======================

This is the process of maintaining multiple versions of the code All the team members upload their code(check in) into the remote version controlling system. The VCS accepts the code uploads from multiple team members and integrates it so that when the other team members download the code they will be able to see the entire work down by the team

VCS's also preserve older and later versions of the code so that at any time we can switch between which ever version we want

VCS's also keep a track of who is making what kind of changes

VCSs are categorised into 2 types

1 Centralised version controlling

2 Distributed version controlling

Centralised Version controlling

-----------------------------------

Here we have a remote server(code repository) into which all the team members check in the code and all the features of version controlling are implemented in this remote server

Distributed version controlling

-------------------------------------

Here we have a local repository installed on every team members machines where version controlling happens at the level of individual team members form where it is uploaded into a remote server where version controlling happens for the entire team

==================================================================

Setting up git on Windows

-------------------------------

1 Download git from

https://git-scm.com/downloads

2 Install it

3 Open gitbash and execute the git commands

==================================================================

Setting up git in ubuntu linux servers

--------------------------------------------

1 Update the apt repository

sudo apt-get update

2 Install git

sudo apt-get install -y git

----------------------------------------------------------------------

Configuring user and email globally for all users on a system

git config --global user.name "your name"

git config --global user. Email "your gmail@gmail.com"

1 To initialize the working dir as a git repo

git init

2 To send a file into from working dir to stagging area

git add filename

To send multiple files

git add file1 file2 file3

To send all files

git add .

. represents current working dir

3 To bring files from stagging area to working dir

git rm --cached filename (or) git reset filename

4 To send files from stagging area to local repository

git commit -m "Some msg"

5 To check the status of working dir and stagging area files

git status

6 To see the commits done on the local repository

git log

To see this output in simple one line format

git log --oneline

Branching in Git

========================

This is a feature of git using which we can create separate branches for different functionalities and later merge them with the main branch also known as the master branch. This will help in creating the code in an uncluttered way

1 To see the list of local branches

git branch

2 To see the list all branches local and remote

git branch -a

3 To create a branch

git branch branch\_name

4 To move into a branch

git checkout branch\_name

5 To create a branch and also move into it

git checkout -b branch\_name

6 To merge a branch

git merge branch\_name

7 To delete a branch that is merged

git branch -d branch\_name

This is also called as soft delete

8 To delete a branch that is not merged

git branch -D branch\_name

This is also known as hard delete

Git Merge

===============

Merging always happens bases on the time stamps of the commits

1 Create few commits on master

touch f1

git add .

git commit -m "a"

touch f2

git add .

git commit -m "b"

2 Check the git commit history

git log --oneline

3 Create a test branch and create few commits on it

git checkout -b test

touch f3

git add .

git commit -m "c"

touch f4

git add .

git commit -m "d"

4 Check the commit history

git log --oneline

5 Go back to master and create few more commits

git checkout master

touch f5

git add .

git commit -m "e"

touch f6

git add .

git commit -m "f"

6 Check the commit history

git log --oneline

9 Merge test with master

git merge test

10 Check the commit history

git log --oneline

=================================================================

Git rebase

This is called as fast forward merge where the commits coming from a

branch are projected as the top most commits on master branch

1 Implement step1-6 from above scenario

2 To rebase test with master

git checkout test

git rebase master

git checkout master

git merge test

3 Check the commit history

git log --oneline

Git Cherry picking

=========================

This is used to selectively pick up certain commits and add them to the

master branch

1 On master create few commits

a--->b

2 Create a test branch and create few commits

git checkout -b test

a--->b--->c--->d--->e--->f--->g

3 To bring only c and e commits to master

git checkout master

git cherry-pick c\_commitid e\_commitid

Working on the Github

=========================

This is the remote repository into which the code is uploaded and

this process is called as checking.

1 Signup for a github account

2 Sign in into that account

3 Click on + on top right corner

4 Click on New repository

5 Enter some repository name

6 Select Public or Private

7 Click on Create repository

8 Go to Push an existing repository from command line and copy paste

the commands

Enter username and password of github

Downloading the code from the remote github

===============================================

This can be done in three ways

git clone

git fetch

git pull

==========

git clone

This will download all the code from the remote repository into the local repository and it is generally used only once when all the team members want a copy of the same code

Syntax: git clone remote\_git\_repo\_url

git fetch

==================

This will download only the modified files but it will place them on a separate branch called as "remote branch”, we can go into this remote branch check if the modifications are acceptable and then merge it with the main branch

1 Open the github

2 Go to the repository that we uploaded

3 Select a file and edit it--->Click on commit changes

4 Open git bash

5 git fetch

6 To see the name of remote branch

git branch -a

7 To switch into this branch

git checkout branch\_name\_from\_step6

8 View the modified file

cat filename

9 If these modifications are ok then merge with main branch

git checkout main

git merge branch\_name\_from\_step6

git pull

===================

This will download only the modified files and merge them with

our local branches

1 Open the github

2 Go to the repository that we uploaded

3 Select a file and edit it--->Click on commit changes

4 Open git bash

5 git pull

We can see the modified files on the main branch