Git Merging

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

Merging of a branch always happens based on the time stamp

of the commits

1 Create few commits on the master

touch file1

git add .

git commit -m “a”

touch file2

git add .

git commitm -m “b”

2 Check the commit history

git log –oneline

3 Create a new branch and create few commits

git checkout -b test

touch file3

git add .

git commit -m “c”

touch file4

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 file5

git add .

git commit -m “e”

touch file6

git add .

git commit -m “f”

6 Check the commit history

git log –oneline

7 Merge the test branch with master

git merge test

8 Delete the test branch

git branch -d test

9 Check the commit history

git log –oneline

Git Rebasing

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

This is also called fast forward merge and here the commits coming from a

branch will projected as the top most commits on the master branch

1 Repeat step 1-6 from previous scenario

2 Rebase test with master

git checkout test

git rebase master

git checkout master

git merge test

3 Check the commit history

git log –oneline

This is also known as fast forward merge

Git cherry pick

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

Both git merge and git rebase bring all the commits from a

branch into the master branch.

If we want to selectively pick up only certain commits and add them

to the master branch then we can use cherry pick

1 Create a commit on master

touch f1

git add .

git commit -m “a”

2 Check the commit history

git log –oneline

3 Create a new branch and create few commits

touch f2

git add .

git commit -m “b”

touch f3

git add .

git commit -m “c”

touch f4

git add .

git commit -m “d”

touch f5

git add .

git commit -m “e”

4 Check the commit history

git log –oneline

5 To cherry pick only c and e commits to master

git checkout master

git cherry-pick c\_commitid e\_commitid