Git Class1

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

SCM(Source code management tool)

Downlod the git according to your OS and install it

git status

git init

git config --global user.name "Srinath Kaithoju"

git config --gobal user.email "srinath.kaithoju@gmail.com"

to check whether the configured it properly or not

git config --list

we can edit the existing user in Git config

\*we cannot create multiple users in git because it is a local machne

how to remove the configured user name in config

git config --global --unset user.name

git config --global --unset user.email

Class2

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

In Git we have three phases

Workspace

Staging/Index

Local repo

Git init

touch file1 this is in workspace

ls file1

git status

to move file to staging

git add file1

ls

now check git status

now commit it

git status

git config --global user.name "srinath"

git config --global suer.email "srinath.kaithoju@gmail.com"

git commit -m "committed" file1

git status

now we dont have files in staging or index area

git log

this will show commit id author name email

date

git show commit\_id

it will show the file name

to add multiple files to gihub

touch file3 file4 file5

ls

git status

git add file3 file4 file5

or

git add .

or

git add \*

or

git add -A

git commit -m " files added"

git log

git show commit\_id

directly we can give git commit -m "file added"

rm file\_name

git status

git log

q to quit

after adding files from working directory to staging

git add .

to revert to working director from staging(to unstage it)

git restore --staged <file>

git rm --cached <file> to unstage it

git config --global merge.tool tortoisemerge

git mergetool

after resolving the issues save it

git status

git commit -m "merge tool"

git class3

uploading a file to a central repository

When we clone the git repository to a location

the total repo will be copied

automatically .git will come

in this we will be having .config file

ls

cd reponame

touch filename1

git add .

git commit -m "file added"

git status

git log

git push

it will ask the credentials

here we will be having token and using sign with browser

ls

touch file1 file2 file3

git add file 2

git status

\*\*to unstage a file from staging( after using git add command) , we can use

git restore --staged filename

git commit -m "added"

git push

only committed files will be pushed to central repo from local repo

\*\*we cannot push only some files from local repo to central repo,all files will be pushed

when we are pushing data after cloning from other repo, we will get the error if you don't have colloboration access

though it is public repository.

to give permission

go to settings

colloborators

add people with organisation

when we are pushing data to central repo

we have to use

\*\*git pull --rebase this will pull changes from central repo to local repo

then use git push

when we push on

Git class4

===========

how to move file from workspace to staging and to local repo

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

to move staging/index to workspace

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

git reset head file\_name

git reset head file\_name1,file\_name2

or

git reset head \*

or

git reset head .

and git restore --staged file\_name

to move files local repo to staging /index

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

git reset --soft cid

after commiting

git log

git status

git reset --soft cid-1

git status

to move files from local repo to workspace

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

git reset --mixed cid-1

\*\*when you are using command , this will bring the files from staging/index to workspace in addition to brining files from local repo

to workspace

to delete commit and commit related files we have to use

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

git status

touch file1,file2

git add .

git commit -m "two files added"

git status

git reset --hard commit\_id-1

git status

Branching

========

ls

git branch

git branch -r

git branch branch\_name

git branch

current branch will be having colour indication

git checkout branch\_name

git branch

if we adding any files in a specific branch we have add and commit there only

git chekcout main

touch file1

git add .

git commit -m "file added"

git checkout release1

ls

we don't have files , which we created in main branch

to merge the file created in main branch to release1

git checkout release1

git merge main

it will show files | number of lines

to delete branch in local

git checkout release1

git branch -d release1

we cannot delete because we are in release1

we have to move to other branch and delete it

\*\* If we have data in any branch and added and committed and we didnot merged

then we will get a message so we have to use D instead of d

git branch -D release1

how to move upload branched to central repositories

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

git branch

git branch release1

to upload release1 branch to central repo

git push origin release1

or git push path-of-central-repo release1

when we are editing or modifying the data will go to corresponding brach when we push

if we are main branch we can push data to central repository

and when we need data to release1 also

git checkour release1 (from main)

git merge main

ls

now we will get data from main to release1

now when we do

git push

we will get a fatal error

it shows a suggestions message as

git push --set-upstream origin release1

now we have to three things

git push origin branch\_name

git push pathof\_the\_central\_repository branch\_name(like we pushed branch to central repo)

or

in the .git /config we have modify the details of the branch

touch f3

git add .

git commit -m "added a file"

git push error will come

git push https://github/polarapu/batch.git release1( path of the central repository)

touch f4

git add .

git commit -m "added a file"

if we wish that git push has to work

.git/config open with notepad

[branch="release"]

remote=origin

merge=refs/heads/release1

now git push will work directly

git push

\*\* In above we merged locally and pushed to central repository

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

we can merge in central repository also

click on pull request

select base to compare

create a pull request

to let them know we will add managers in the reviewers list

we can add messages also

and click on merge pull request

and confirm merge

now it branch data will be merged to main

\*\*After adding some files in a branch and committing we can merge to main and push to central repository

we can merge from central repository only using GUI method

we have to add files and commit and push to same repo and then we have to do the pull request and upload.

Aliases

======

git status

git config --global alias.s "status"

git s

git config --list

git config --global alias.cl "config --list"

git cl

git log

git config --global alias.l "log"

git l

to know which aliases we configured

git config --list

to unset aliases

git config --global --unset alias.s

git config --global --unset alias.cl

git class 5

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

logs

========

git log

git log -n

git log --oneline

git log --oneline -n

git class 5

logs

========

git log

git log -n

git log --oneline

git log --oneline -n

git log --author=authorname

git log --author=authorname -n

git log --author=authorname --oneline

git log --author=authorname --oneline -n

git log --since =yy-mm-dd

or

git log --after=yy-mm-dd

git log --until=yy-mm-dd

or

git log --before yy-mm-dd

git log --since =yy-mm-dd --until=yy-mm-dd

or

git log --after=yy-mm-dd --before=yy-mm-dd

git log --after="yy-mm-dd 24:00" --before="yy-mm-dd 20:00"

git log --filename

git log --grep "label"

Conflicts

========

if we have two branches

like master

In the master create a file srinath.txt

git add .

git commit -m "added a file"

git checkout -b testbranch

in the srinath.txt file

add something data to srinath.txt

git add .

git commit -m "added a file"

git checkout main

git merge main

to comeout from the situation

git merge --abort

we will get the merge conflict with main|MERGING

open the file and keep the chanes and save it

git status

git add .

git commit -m "added and made some changes"

now we can merge the data

git merge testbranch

\*\* add git merge tool from Ravi Sir

Git class 6th

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

TAG:if we have many commits, to find out upto which commit we have the release point

or to mark the release point we will use this.

git clone gitrepo

git checkout main

ls

git pull --rebase

git rebase --continue

(if we have merge conflict in file1 then open it add do add and commit it.

vi file1

git add .

git commit -m "test commit"

git rebase --continue

git push

ls

git log

we dont have any tags

to create the new tag

git tag

>>git tag T-1.0 it will create tag to the latest commit

>>git tag

>>git show T-1.0

>>git log

above will disaply the tag

>>git log --oneline

got the latest we have create the tag

to delete the tag

>>git tag -d T-1.0

git tag

>>git log --oneline

to create a tag upto a specific commit

git log --oneline to know the specific commit ids

>>git tag T1-0 commitid

>>git log --oneline

>>git tag T-2.0 commitid

>>git tag T-3.0 commitid

git tag T-4.0 if don't give the commit\_id it will take the latest commit\_id

>>git tag

>>git log --online

now we can see all the tags

we can push the tags to the central repos also

to get into the specific tag

git checkout T-1.0

to upload the tag into the central repository

git push origin T-1.0

in the central repo we will find in the tags section

we can see the tags in the branch section

git push --tags

tag is for mark the release the point

we cannot merge the data from one tag to another tag

git tag

git tag T-1.0

git show T-1.0

git checkout T-1.0

git tag -d T-1.0(delete on local)

git tag t2.0 cid ( to create tag for special commmit)

git push origin T2.0( to upload into cntral repo)

git push --tags (all tags)

how to delete the branch in the central repository

git push origin -d release

to delete the tag in the central repository

git push origin -d T2.0( to delete on central repo)

STASH

=======

we cannot store the stashed files in the github

we can stash staging files and modified files, not the local repository files and not the files created in workspace

we can stash and we can back the files from stash

to list the stashed files

git stash list

git status to check files in staging any modified area

git clone a repo and add a file or modify a file then if we check git status we can see in it

git status

we can see the modified files

git stash save "save1"

git status

we cannot see the modified files

git stash list

git stash show stash@{0}

this stash@{0} will come with stash list

to know the contents of the stashc

git stash show -p stash@{0}

stash@{0} is having label as stash1

now when we create the new stash , the new stash will be having

stash@{0}: on main :stash2

stash@{1}: on main :stash1

as we are increaing the stash the stash will zero "0" will be the index

\*\*

All are stored in . git/refs/stash . git stash saves stashes indefinitely, and all of them are listed by git stash list . Please note that dropping or clearing the stash will remove it from the stash list, but you might still have unpruned nodes with the right data lying around.

how to retrive the data after stashing

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

POP,APPLY,DROP

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

POP=cut( we will having be having only one copy)

==========

to bring the data back, it will bring the files to the corresponding staging or modified files.

git stash pop >> default is 0 stash

git stash pop stash@{X} >> this is for a particular stash

APPLY=Copy (we will be having two copied at source and destination(it will bring the data back and keeps in stash )

===========

git stash apply

git stash apply stash@{X}

DROP=delete

===========

git stash drop

git stash drop stash@{X}

without applying stash apply it will not be saved

to delete we will use the drop

how to use the token in github

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

in the settings option click on the developer settings in the personal access tokens

give a name and give the expiration settings and select the all permissions then click on the generate the token

to delete the existing token in windows control manager use account credentials manager delete the existing tokens

and now add the token

git merge and git rebase

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

when we are merging one branch data to another a commit id will be created

whenever we are merging the same file which is in different branches we will get the merge conflict

when we are merging we will get the extra commit

but when we use rebase instead of the merge, it will copy all data from one branch to another . no commit ids

git rebase branchname

git rebase --continue

one more command from the git rabase

git rebase --abort

when we use git pull --rebase

all changes in the central repo will be updated to local and on top of that our commits will be noted

Git class 7

Git cherry-pick

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

>> When we merge data from one branch to another all the data will be copied

with all commit from a particular branch

>> If we wish to copy data to another

branch of a particular commit we ill use the cherrypick.

git status

if we have data in main/master

create a branch

git branch

git checkout -b srinathbranch

git branch

git chekcout main

create some files in this branch

git log

create some commits

git checkout srinathbranch

git log

git merge main

we will get all the commits from thr main

git log

here we are getting the all the commits

git checkout main

create two more commits

git log

git chekcout srinathbranch

ls

git cherry-pick commit\_id

ls

we will get the data of that commit only

git log -3

we will see that commit only

Git ammend

==========

git commit -m "lable mesg"

git commit -m " sammmmmmmpllleeee commit"

we can change the commit by bringing back the data to workspace or staging we can change

instead of doing that we can change the lable

by using

git commit --amend -m "sample commit"

other purpose:

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

Suppose we have created a file in workspace and added to staging ang commited to local

now if we want to add some data that commit we can do that using ammend, this will work only latest commit.

git status

touch s4,s5

git add .

git status

after this if use the git commit(git commit -m "added a file") it will add the data to the latest commit,

instead of this we can use

git commit --amend -m "s1 to s3 files"

git status

Git Revert

=========

\*\*To show the changes in a commit

git show commit\_id

- means removed,means changed

once if we add any files and push to github, then if that is breaking something and we want to revert the chanegs changes then we will use the revert command

after pushing the data to github

do this with the latest commit

git revert commit\_id

vi filename

git push

if suppose everything is fine with our commit, the breakdown is something with other then we have re revert te commit

git log -3

we will see the mesg as this " this revert commit....."

git revert commit\_id

git push

again can we do this revert

we can check the rever revert revert...

Git diff

======

To check the difference between two files or between two branches we can use git diff

git diff file1 file2

git diff branch1 branch2

git diff file1

vi f1

vi f2

git diff f1 f2

git diff main srinathtestbranch

vi .gitignore

touch f1 f2 f3

when we do git status it will not display the added files

GIT Blame

=========

git blame filename

git blame -L 1,2 filename

to cinfine the results to some lines

git blame tagname filename

git blame --since=1.weel filename

details of timestamp of the file

Remove dead branches

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

git remote update --prune

Git Bisect

========

it is a binary search

git log

git bisect start

git bisect bad

git bisect good commit\_id

git bisect bad

git bisect good

git show

\*\*Ammend command will applicable for only latest commit