**ITU SWE 525 Version Control : Question Bank for Inclass MidTerm:**

1.Follow the instructions here (https://github.com) to create a Github account. Write your Github account name in your homework submission.

**Github account name : archiullal**

2.Clone the Angular project onto your computer. Make a screenshot of the git command you entered to do this and the result, and include it in your response. Link to Angular Project <https://github.com/angular/angular>

**mkdir midterm**

**cd midterm**

**git init**

**git clone** <https://github.com/angular/angular>

3.Use your local Angular repo to find the commit made at Mon Dec 14 20:02:40 2015 (depending on how you do it, you might find Tue Jan 12 12:10:42 2016 instead). Who is the author? What are the commit's short hash and one-line description?

**ls you will find angular**

**git log –p -1**

Answer the following questions:

1.Who is the creator of git? What license does git use?

* **Creator of git is Linus Torvalds**
* **Git is** [**free software**](https://en.wikipedia.org/wiki/Free_software) **distributed under the terms of the** [**GNU General Public License**](https://en.wikipedia.org/wiki/GNU_General_Public_License) **version 2.**

2.Are git and Github different things?

**GIT**

* Git is a version control system, a tool to manage your source code history.
* Git is a tool

**GitHub**,

* It is a web page on which we can publish our Git repositories and collaborate with other people.
* GitHub the service for projects that use Git

3.Explain what a commit in git represents.

Git commit Record changes to the repository. It saves the changes in remote repository along with log message and creates commit hash to track the changes.

4.What is the difference between a CVCS and a DVCS?

**CVCS – Centralized Version Control System**

* Centralized version control systems are based on the idea that there is a single “central” copy of your project somewhere (probably on a server), and programmers will “commit” their changes to this central copy.
* Committing” a change simply means recording the change in the central system. Other programmers can then see this change. They can also pull down the change, and the version control tool will automatically update the contents of any files that were changed.
* Most modern version control systems deal with “changesets,” which simply are a groups of changes (possibly to many files) that should be treated as a cohesive whole

**DVCS- Distributed Version Control System**

* DVCS do not necessarily rely on a central server to store all the versions of a project’s files. Instead, every developer “clones” a copy of a repository and has the full history of the project on their own hard drive. This copy (or “clone”) has all of the metadata of the original.
* Performing actions other than pushing and pulling change sets is extremely fast in DVCS, because the tool only needs to access the hard drive, not a remote server.
* Committing new changesets can be done locally without anyone else seeing them. Once you have a group of changesets ready, you can push all of them at once.
* Everything but pushing and pulling can be done without an Internet connection. So you can work on a plane, and you won’t be forced to commit several bug fixes as one big changeset.
* Since each programmer has a full copy of the project repository, they can share changes with one or two other people at a time if they want to get some feedback before showing the changes to everyone.

1.Download this repo here and unzip it. Do the following to the repo, then zip it up and send it with your email. Link to Repo: <https://www.dropbox.com/s/ldh806d1xiog7qq/hw2repo.zip?dl=0>

* Make some modification to only file1.txt, then commit it.
* Make some modification to both file1.txt and file2.txt, then commit it.
* Create some file file3.txt, and commit it.
* In lecture we discussed that the general workflow for making commits is edit-add-commit. It seems like you should just be able to modify files, then commit your changes. Why is the git add step necessary? In other words, why do we need or want a staging area? Explain in a few sentences (intentionally open-ended, I just want you to think about it).

**The staging area is a file, generally contained in your Git directory, that stores information about what will go into your next commit. It’s sometimes referred to as the “index”, but it’s also common to refer to it as the staging area.**

* Give an example of a bad commit message.

**git commit –m “added”**

**Its Bad commit message because its not clear message. We don’t know what is added.**

* Give an example of a good commit message.

**git commit –m “ file01.txt modified in repos repository“**

**It’s a good commit message because its clear.**

* We've seen how to update files in the staging area with files from the working directory via git add. Why does it make less sense to update files in the working directory with files from the staging area? Explain briefly.

**Staging helps you "check off" individual changes as you review a complex commit, and to concentrate on the stuff that has not yet passed your review.**

**Before you commit, you'll probably review the whole change by using git diff. If you stage each change as you review it, you'll find that you can concentrate better on the changes that are not yet staged.**

1. After a long vacation in Bermuda, Charlie returns to work on his project but discovers that someone has deleted all the files in his project with rm -rf \*! Fortunately, the unknown adversary didn't realize Charlie followed good software practices and was using version control with git, so they forgot to delete the .git directory.

* Describe in words what the output would generally look like if Charlie runs git status.

**If Charlie runs git status then it looks something like below**

Archanas-MacBook-Pro:hw2repo Archana$ git status

On branch master

Changes not staged for commit:

(use "git add/rm <file>..." to update what will be committed)

(use "git checkout -- <file>..." to discard changes in working directory)

deleted: file1.txt

deleted: file2.txt

Untracked files:

(use "git add <file>..." to include in what will be committed)

.DS\_Store

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

* Give a command or a series of commands that would restore Charlie's project from the last commit (HEAD).

**Archanas-MacBook-Pro:hw2repo Archana$ git checkout .**

**Archanas-MacBook-Pro:hw2repo Archana$ git reset HEAD –hard**

3.After reading StackOverflow, Lucas decides that the correct way to make commits in git is to do git add .. Upon running git status, he discovers that as a result, he's staged the following files that he doesn't want to include in his next commit: .DS\_Store, npm-debug.log, passwords.txt, mydiary.txt.

Advise Lucas

1. what to do to fix the immediate problem and also

**git add . stages new and modified, without deleted. To fix this problem he should reset it.**

**git reset <filePath> will unstage any staged changes for the given file(s).**

1. what he could do specifically to prevent this from happening in the future (saying "don't trust everything you read on StackOverflow" is not sufficient).

To prevent this from happening in the future he should do

**git add <file name>**

4. After editing a source file for a few days, Ana decides that she likes the version she had a few days ago better than what she has now. She now wishes to replace the current version of the file with that previous version.

* Ana doesn't remember the short hash of the most recent commit the previous version was last part of. Practically speaking, what would Ana have to do to find this out?

Ana should type **git log** by doing this she will get short hash of the commit she requires.

* Once Ana finds this commit, how can she restore that previous version of the file? Note that she wants to restore just that file, not every file in the commit.

**git checkout <hash number> <file name>**

5.A git repo is in the following state:

|  |
| --- |
| **1. Which of the descriptions below accurately describes Git?** |
| a type of corporate repository |
| an error tracker |
| a distributed version control system |
| a file server |
| **2. When wanting to use a Git client to create local commits, the minimum requirement for settings information which a user must first supply to Git are/is...** |
| username and password |
| name and email address |
| there is no minimum requirement |
| email address |
| **3. Regarding the command 'git checkout', which statement is true?** |
| it is normally used to create a local copy of a repo |
| it can be used to switch between local branches |
| it involves communication with a remote |
| it never takes additional parameters |
| **4. Regarding the command 'git log' (without additional parameters) which statement is true?** |
| it provides information about all commits reachable from HEAD |
| it provides information about all commits associated with the repo |
| it provides information about the latest commit only |
| it provides information about the user's commits only |
| **5. I have two local Git repositories and wish to merge them together into a single repository. Which statement is true?** |
| it is normally simple to do, but all commit IDs in the combined repo will differ from the original commits |
| git-filter-branch must be used to achieve this |
| there are no simple ways to achieve this |
| it is normally possible to merge them without affecting any existing commit IDs from either repository |
| **6. Complete the sentence correctly. A git branch is ...** |
| always stored on a server |
| a reference to a single commit |
| a series of related commits |
| normally stored in a separate folder on the server |
| **7. Regarding 'git fetch', 'git pull' and 'git push' which of the following statements is true?** |
| 'git fetch' does not communicate with a remote, 'git pull' does |
| both 'git fetch' and 'git pull' retrieve all new commits from a remote |
| 'git pull' merges content for all branches, 'git fetch' does not |
| 'git pull' is the opposite of 'git push' |
| **8. We wish Git to ignore some files in our working copy. Which statement is true?** |
| a .gitignore file could be used with a pattern which matches tracked files |
| a .gitignore file could be used with a pattern which matches untracked files |
| the files could be moved to a .gitignore directory |
| the file names to be ignored should be specified on the central server |
| **9. A developer's Git repository can be considered to consist of which elements?** |
| Working Copy and Log |
| Database and Working Copy |
| Index, Working Copy, Log |
| Database, Index, Working Copy |
| **10. Regarding Git and binary files, which of the following statements is most accurate?** |
| You can use large binaries with Git but it is not recomended due to the impact on performance |
| Git stops you storing very large binaries in a repo |
| Git uses locking to avoid merge problems with binary files |
| The latest versions of Git have good support for large binary file usage |

How do you create a copy of a lab under your own GitHub account so that you can solve the lab?

Forking it via the GitHub interface.

git fork

git clone

git pull-request

What's the opposite of git clone, instead of downloading your code from GitHub, uploads your changes and code back to GitHub?

git push

git add

git upload

git status

How do you check the state of your local git repository since your last commit?

git check

git status

git commit

git diff

How do you stage files for a commit?

git stage

git commit

git add

git reset

How do you save the current state of your code into the git version control?

By committing the staged changes with git commit

By adding all changes and staging them with git stage

By adding all changes and staging them with git add

By creating a new commit with git init

What's a shortcut to staging all the changes you have?

git commit add .

git commit .

git add .

git push -am "Message"

How do you supply a commit message to a commit?

git message "I'm coding"

git add "I'm coding"

git commit "I'm coding"

git commit -m "I'm coding"

What is the correct commit syntax for all changes with a message?

git message -am "I'm coding"

git add -a "I'm coding"

git commit -a "I'm coding"

git commit -am "I'm coding"

How do you submit a solution to Learn?

git submit

git pull-request

By creating a Pull Request through the GitHub interface

git commit -am "Done with Lab"

What comes first, staging with git add . or committing with git commit?

Staging your commits with git add

Committing with git commit

**https://www.toptal.com/git/interview-questions**

**Short Questions:**

How do you revert a commit that has already been pushed and made public?

→

How do you squash last N commits into a single commit?

→

How do you find a list of files that has changed in a particular commit?

→

How do you setup a script to run every time a repository receives new commits through push?

→

What is git bisect? How can you use it to determine the source of a (regression) bug?

→

What are the different ways you can refer to a commit?

→

What is git rebase and how can it be used to resolve conflicts in a feature branch before merge?

→

How do you configure a Git repository to run code sanity checking tools right before making commits, and preventing them if the test fails?

**Ans: https://www.toptal.com/git/interview-questions**

**GURU99 45 question and answer**

**http://career.guru99.com/top-40-interview-questions-on-git/**

1) What is GIT?

GIT is a distributed version control system and source code management (SCM) system with an emphasis to handle small and large projects with speed and efficiency.

2) What is a repository in GIT?

A repository contains a directory named .git, where git keeps all of its metadata for the repository. The content of the .git directory are private to git.

3) What is the command you can use to write a commit message?

Git

The command that is used to write a commit message is “git commit –a”. The –a on the command line instructs git to commit the new content of all tracked files that have been modified. You can use “git add<file>” before git commit –a if new files need to be committed for the first time.

4) What is the difference between GIT and SVN?

The difference between GIT and SVN is

a) Git is less preferred for handling extremely large files or frequently changing binary files while SVN can handle multiple projects stored in the same repository.

b) GIT does not support ‘commits’ across multiple branches or tags. Subversion allows the creation of folders at any location in the repository layout.

c) Gits are unchangeable, while Subversion allows committers to treat a tag as a branch and to create multiple revisions under a tag root.

5) What are the advantages of using GIT?

a) Data redundancy and replication

b) High availability

c) Only one.git directory per repository

d) Superior disk utilization and network performance

e) Collaboration friendly

f) Any sort of projects can use GIT

6) What language is used in GIT?

GIT is fast, and ‘C’ language makes this possible by reducing the overhead of runtimes associated with higher languages.

7) What is the function of ‘GIT PUSH’ in GIT?

‘GIT PUSH’ updates remote refs along with associated objects.

8) Why GIT better than Subversion?

GIT is an open source version control system; it will allow you to run ‘versions’ of a project, which show the changes that were made to the code overtime also it allows you keep the backtrack if necessary and undo those changes. Multiple developers can checkout, and upload changes and each change can then be attributed to a specific developer.

9) What is “Staging Area” or “Index” in GIT?

Before completing the commits, it can be formatted and reviewed in an intermediate area known as ‘Staging Area’ or ‘Index’.

10) What is GIT stash?

GIT stash takes the current state of the working directory and index and puts in on the stack for later and gives you back a clean working directory. So in case if you are in the middle of something and need to jump over to the other job, and at the same time you don’t want to lose your current edits then you can use GIT stash.

11) What is GIT stash drop?

When you are done with the stashed item or want to remove it from the list, run the git ‘stash drop’ command. It will remove the last added stash item by default, and it can also remove a specific item if you include as an argument.

12) How will you know in GIT if a branch has been already merged into master?

Git branch—merged lists the branches that have been merged into the current branch

Git branch—-no merged lists the branches that have not been merged

13) What is the function of git clone?

The git clone command creates a copy of an existing Git repository. To get the copy of a central repository, ‘cloning’ is the most common way used by programmers.

14) What is the function of ‘git config’?

The ‘git config’ command is a convenient way to set configuration options for your Git installation. Behaviour of a repository, user info, preferences etc. can be defined through this command.

15) What does commit object contain?

a) A set of files, representing the state of a project at a given point of time

b) Reference to parent commit objects

c) An SHAI name, a 40 character string that uniquely identifies the commit object.

16) How can you create a repository in Git?

In Git, to create a repository, create a directory for the project if it does not exist, and then run command “git init”. By running this command .git directory will be created in the project directory, the directory does not need to be empty.

17) What is ‘head’ in git and how many heads can be created in a repository?

A ‘head’ is simply a reference to a commit object. In every repository, there is a default head referred as “Master”. A repository can contain any number of heads.

18) What is the purpose of branching in GIT?

The purpose of branching in GIT is that you can create your own branch and jump between those branches. It will allow you to go to your previous work keeping your recent work intact.

19) What is the common branching pattern in GIT?

The common way of creating branch in GIT is to maintain one as “Main“

branch and create another branch to implement new features. This pattern is particularly useful when there are multiple developers working on a single project.

20) How can you bring a new feature in the main branch?

To bring a new feature in the main branch, you can use a command “git merge” or “git pull command”.

21) What is a ‘conflict’ in git?

A ‘conflict’ arises when the commit that has to be merged has some change in one place, and the current commit also has a change at the same place. Git will not be able to predict which change should take precedence.

22) How can conflict in git resolved?

To resolve the conflict in git, edit the files to fix the conflicting changes and then add the resolved files by running “git add” after that to commit the repaired merge, run “git commit”. Git remembers that you are in the middle of a merger, so it sets the parents of the commit correctly.

23) To delete a branch what is the command that is used?

Once your development branch is merged into the main branch, you don’t need

development branch. To delete a branch use, the command “git branch –d [head]”.

24) What is another option for merging in git?

“Rebasing” is an alternative to merging in git.

25) What is the syntax for “Rebasing” in Git?

The syntax used for rebase is “git rebase [new-commit] “

26) What is the difference between ‘git remote’ and ‘git clone’?

‘git remote add’ just creates an entry in your git config that specifies a name for a particular URL. While, ‘git clone’ creates a new git repository by copying and existing one located at the URI.

27) What is GIT version control?

With the help of GIT version control, you can track the history of a collection of files and includes the functionality to revert the collection of files to another version. Each version captures a snapshot of the file system at a certain point of time. A collection of files and their complete history are stored in a repository.

28) Mention some of the best graphical GIT client for LINUX?

Some of the best GIT client for LINUX is

a) Git Cola

b) Git-g

c) Smart git

d) Giggle

e) Git GUI

f) qGit

29) What is Subgit? Why to use Subgit?

‘Subgit’ is a tool for a smooth, stress-free SVN to Git migration. Subgit is a solution for a company -wide migration from SVN to Git that is:

a) It is much better than git-svn

b) No requirement to change the infrastructure that is already placed

c) Allows to use all git and all sub-version features

d) Provides genuine stress –free migration experience.

30) What is the function of ‘git diff ’ in git?

‘git diff ’ shows the changes between commits, commit and working tree etc.

31) What is ‘git status’ is used for?

As ‘Git Status’ shows you the difference between the working directory and the index, it is helpful in understanding a git more comprehensively.

32) What is the difference between the ‘git diff ’and ‘git status’?

‘git diff’ is similar to ‘git status’, but it shows the differences between various commits and also between the working directory and index.

33) What is the function of ‘git checkout’ in git?

A ‘git checkout’ command is used to update directories or specific files in your working tree with those from another branch without merging it in the whole branch.

34) What is the function of ‘git rm’?

To remove the file from the staging area and also off your disk ‘git rm’ is used.

35) What is the function of ‘git stash apply’?

When you want to continue working where you have left your work, ‘git stash apply’ command is used to bring back the saved changes onto the working directory.

36) What is the use of ‘git log’?

To find specific commits in your project history- by author, date, content or history ‘git log’ is used.

37) What is ‘git add’ is used for?

‘git add’ adds file changes in your existing directory to your index.

38) What is the function of ‘git reset’?

The function of ‘Git Reset’ is to reset your index as well as the working directory to the state of your last commit.

39) What is git Is-tree?

‘git Is-tree’ represents a tree object including the mode and the name of each item and the SHA-1 value of the blob or the tree.

40) How git instaweb is used?

‘Git Instaweb’ automatically directs a web browser and runs webserver with an interface into your local repository.

41) What does ‘hooks’ consist of in git?

This directory consists of Shell scripts which are activated after running the corresponding Git commands. For example, git will try to execute the post-commit script after you run a commit.

42) Explain what is commit message?

Commit message is a feature of git which appears when you commit a change. Git provides you a text editor where you can enter the modifications made in commits.

43) How can you fix a broken commit?

To fix any broken commit, you will use the command “git commit—amend”. By running this command, you can fix the broken commit message in the editor.

44) Why is it advisable to create an additional commit rather than amending an existing commit?

There are couple of reason

a) The amend operation will destroy the state that was previously saved in a commit. If it’s just the commit message being changed then that’s not an issue. But if the contents are being amended then chances of eliminating something important remains more.

b) Abusing “git commit- amend” can cause a small commit to grow and acquire unrelated changes.

45) What is ‘bare repository’ in GIT?

To co-ordinate with the distributed development and developers team, especially when you are working on a project from multiple computers ‘Bare Repository’ is used. A bare repository comprises of a version history of your code.

[**http://career.guru99.com/top-40-interview-questions-on-git/**](http://career.guru99.com/top-40-interview-questions-on-git/)