Assignment 1
Document of Git and GitHub

❖ Git

Git is a distributed version-control system for tracking changes in source code during software development.

It is designed for coordinating work among programmers, but it can be used to track changes in any set of files. Its goals include speed,data integrity, and support for distributed, non-linear work flows.

Git was created by Linus Torvalds in 2005 for development of the Linux kernel, with other kernel developers contributing to its initial development.

Its current maintainer since 2005 is Junio Hamano.

As with most other distributed version-control systems, and unlike most client–server systems, every Git directoryon every computer is a full-fledged repository with complete history and full version-tracking abilities, independent of network access or a central server.

Git is free and open-source software distributed under the terms of the GNU General Public License version 2.

Git basic commands

Git task	Notes	Git commands
Tell Git who you are	Configure the author name and email address to be used with your commits. Note that Git strips some characters (for example trailing periods) from `user.name`.	`git config --global user.name "Sam Smith"` `git config --global user.email sam@example.com`
Create a new local repository		`git init`
Check out a repository	Create a working copy of a local repository:	`git clone /path/to/repository`
	For a remote server, use:	`git clone username@host:/path/to/reposit`

		ory

Add files	Add one or more files to staging (index):	`git add <filename>`
		
		`git add *`

Commit	Commit changes to head (but not yet to the remote repository):	`git commit -m "Commit message"`
	Commit any files you've added with `git add`, and also commit any files you've changed since then:	`git commit -a`
Push	Send changes to the master branch of your remote repository:	`git push origin master`
Status	List the files you've changed	`git status`

	and those you still need to add or commit:	
Connect to a remote repository	If you haven't connected your local repository to a remote server, add the server to be able to push to it:	`git remote add origin <server>`
	List all currently configured remote repositories:	`git remote -v`
Branches	Create a new branch and switch to it:	`git checkout -b <branchname>`
	Switch from one branch to another:	`git checkout <branchname>`
	List all the branches in your repo, and also tell you what branch	`git branch`

	you're currently in:	
	Delete the feature branch:	`git branch -d <branchname>`
	Push the branch to your remote repository, so others can use it:	`git push origin <branchname>`
	Push all branches to your remote repository:	`git push --all origin`
	Delete a branch on your remote repository:	`git push origin :<branchname>`
Update from the remote repository	Fetch and merge changes on the remote server to your working directory:	`git pull`
	To merge a different	`git merge <branchname>`

	branch into your active branch:	

	View all the merge conflicts:
	View the conflicts against the base file:
	Preview changes, before merging:
		`git diff`
		`git diff --base <filename>`
		`git diff <sourcebranch> <targetbranch>`

	After you have manually resolved any conflicts, you mark the changed file:	`git add <filename>`
Tags	You can use tagging to mark a significant changeset, such as a release:	`git tag 1.0.0 <commitID>`

	CommitId is the leading characters of the changeset ID, up to 10, but must be unique. Get the ID using:	`git log`
	Push all tags to remote repository:	`git push --tags origin`
Undo local changes	If you mess up, you can replace the changes in your working tree with the last content in head: Changes already added to the index, as well as new files, will be kept.	`git checkout -- <filename>`

	Instead, to drop all your local changes	`git fetch origin`
		
		`git reset --hard origin/master`

	and commits, fetch the latest history from the server and point your local master branch at it, do this:	
Search	Search the working directory for `foo()`:	`git grep "foo()"`

❖ GitHub

GitHub is a web-based hosting service for version control using Git. It is mostly used for computer code.

It offers all of the distributed version control and source code management (SCM) functionality of Git as well as adding its own features.

It provides access control and several collaboration features such as bug tracking, feature requests, task management, and wikis for every project.

GitHub offers plans for enterprise, team, pro and free accounts which are commonly used to host open-source software projects.

As of January 2019, GitHub offers unlimited private repositories to all plans, including free accounts.

As of June 2018, GitHub reports having over 28 million users and 57 million repositories (including 28 million public repositories), making it the largest host of source code in the world.

❖ Difference between Git and GitHub