Task 1 – Code Management System

1 Git – Source Code Management

Git is a popular source code management system. Every Git working directory is a full-fledged repository, with complete history and full version tracking capabilities, not dependent on network access or a central server. In this sense, it trades off space for speed. Git is primarily developed on Linux, although it also supports most major operating systems including BSD, OS X, and Microsoft Windows. Pertinent information about Git can be found on the project website.

- `http://git-scm.com/`

Action Items

- **Read:** Git documentation.
  `http://git-scm.com/doc/`
- **Complete:** Try git tutorial.
  `https://try.github.io/levels/1/challenges/1`
- **Download and Install:** Git (if needed).
  `http://git-scm.com/downloads`

2 GitHub

GitHub is a web-based hosting service for source code management (SCM). It is built on the Git revision control system and offers free accounts for open source projects. According to the terms of service, if bandwidth usage of an account significantly exceeds the average of other GitHub customers, the associated file hosting service may be immediately disabled.

To facilitate integration of our projects, we will employ a Git archive. Our master Git repository is hosted on GitHub.

`https://github.com/CourseReps/ECEN489-Spring2015.git`

Active participants in this course will be given access to our master repository. You will find below a short list of frequently used commands.

Common Actions

- **Init:** The `init` command creates a new local repository.
- **Clone:** Use `clone` to instantiate a working copy from a master repository. This is usually the first command employed to establish a local working hierarchy under this paradigm. Within Eclipse, you may use the Git Repository Exploring perspective to clone a master repository.
- **Add:** The `add` command is used to add one or more files to staging. Only add pertinent files to the repository.

- **Commit:** The `commit` command incorporates changes to your working copy of the repository.
- **Push:** The `push` command sends changes to the master branch, typically a remote repository.
- **Pull:** The `pull` command fetches and merges changes on the remote server to the local working directory.
- **Mergetool:** Sometimes, there may be a discrepancy between the latest version of a file and its working copy on a given host. In such cases, the developer may need to take action to resolve these issues. This can be achieve through normal editing, followed by the Git `add` command. Alternatively, one can use the `mergetool` command, which initiates a visual tool.
- **Status:** The `status` command lists the status of working files and directories.

Action Items

- **Account:** Go to `https://github.com` and create a developer account (if needed). Email your GitHubID to your instructor via your TAMU account. You will need write permission before you can proceed.
- **Web:** Locate the `README.md` under `Students`. Add your GitHubID next to you name.

```
* <Full Name>, [GitHubID](https://github.com/GitHubID)
```

- **Clone:** Once you have been added to the list of collaborators, use `git` to clone the master repository.
- **Directory:** Under `Students` (in the master branch), make a directory named `<GitHubID>`. This location is where you will commit all your individual work. Within this directory, create a file labeled `README.md` that contains the following information.

```
# Identity

* Name: <Full Name>
* GitHubID: <GitHub ID>
* NetID: <TAMU NetID>
* Preferred email: username@domain
```

  Finally, `add` and `commit` your modifications to your local repository, then `pull` the latest revisions and `push` this information on the master repository.