# Getting started with git

1. **1st question**

Generating ssh key

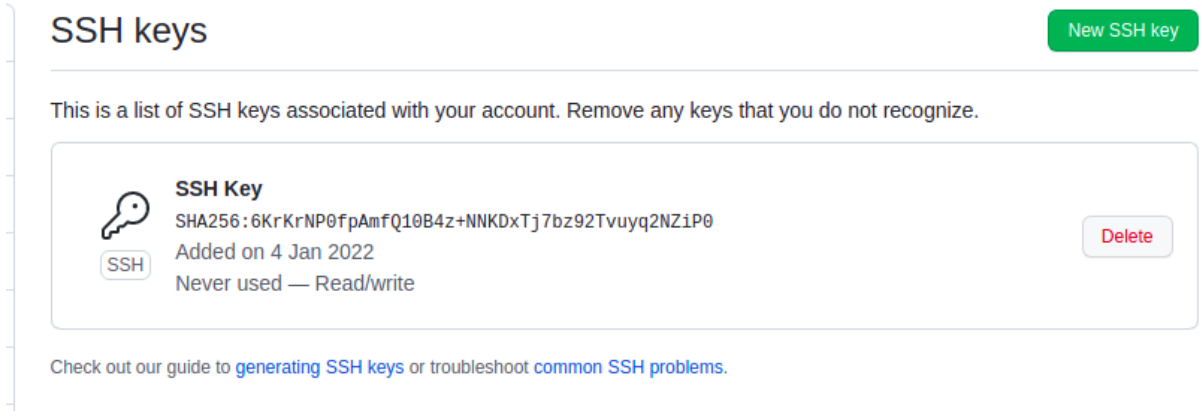
```
ignitarium@IGN-BLR-LP-215:~/git_getting_started$ ssh-keygen -t rsa -b 4096 -C "vegimohnish@gmail.com"
Generating public/private rsa key pair.
Enter file in which to save the key (/home/ignitarium/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ignitarium/.ssh/id_rsa
Your public key has been saved in /home/ignitarium/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:6KrKrNP0fpAmfQ10B4z+NNKDxTj7bz92Tvuyq2NZiP0 vegimohnish@gmail.com
The key's randomart image is:
+---[RSA 4096]----+
|       =..       |
|      = = .      |
|     o B .       |
|      *.=        |
|   . ..BSo o .   |
|  o =.. + . o .  |
| o + o.  .   +.  |
|+ . ...   o *.E. |
|+=.oo.    . +o*==.|
+----[SHA256]-----+
```

So, here public and private key pairs are generated.

```
ignitarium@IGN-BLR-LP-215:~$ cd /home/ignitarium/.ssh
ignitarium@IGN-BLR-LP-215:~/.ssh$ ls
id_rsa  id_rsa.pub
ignitarium@IGN-BLR-LP-215:~/.ssh$
```

Here both keys are generated in the given path. id_rsa is private key and id_rsa.pub is a public key

Updating in the server




---

2. **2nd question**

Configuring username and email using git config

```
ignitarium@IGN-BLR-LP-215:~/git_getting_started$ git config --global user.name VegiMohnish
ignitarium@IGN-BLR-LP-215:~/git_getting_started$ git config --global user.email vegimohnish@gmail.com
ignitarium@IGN-BLR-LP-215:~/git_getting_started$ git config --list
user.name=VegiMohnish
user.email=vegimohnish@gmail.com
core.repositoryformatversion=0
core.filemode=true
core.bare=false
core.logallrefupdates=true
ignitarium@IGN-BLR-LP-215:~/git_getting_started$
```

---

3. **3rd question**

Clone empty repository in local system

```
ignitarium@IGN-BLR-LP-215:~/git_getting_started$ git clone https://github.com/VegiMohnish/training.git
Cloning into 'training'...
warning: You appear to have cloned an empty repository.
ignitarium@IGN-BLR-LP-215:~/git_getting_started$
```

---

4. **4th question**

```
ignitarium@IGN-BLR-LP-215:~/git$ git init
Initialized empty Git repository in /home/ignitarium/git/.git/
ignitarium@IGN-BLR-LP-215:~/git$ git remote add origin "/home/ignitarium/editing_and_browsing_code.pdf
> /home/ignitarium/git_getting_started.pdf
> /home/ignitarium/linux_shell.pdf^C
ignitarium@IGN-BLR-LP-215:~/git$ git remote add origin "https://github.com/VegiMohnish/training.git"
ignitarium@IGN-BLR-LP-215:~/git$ git remote -v
origin  https://github.com/VegiMohnish/training.git (fetch)
origin  https://github.com/VegiMohnish/training.git (push)
ignitarium@IGN-BLR-LP-215:~/git$ git add .
ignitarium@IGN-BLR-LP-215:~/git$ git commit -m "my training exercises"
[master (root-commit) 29e0e53] my training exercises
 22 files changed, 86 insertions(+)
 create mode 100644 editing_and_browsing_code.pdf
 create mode 100644 git_getting_started.pdf
 create mode 100644 linux_shell.pdf
 create mode 100644 linuxshell/2ndquestion.png
 create mode 100644 linuxshell/3rdquestion.png
 create mode 100644 linuxshell/6thquestion.png
 create mode 100755 linuxshell/8.sh
 create mode 100644 linuxshell/8thquestion.png
 create mode 100755 linuxshell/9.sh
 create mode 100644 linuxshell/9thquestion.png
 create mode 100644 linuxshell/Bash
 create mode 100644 linuxshell/Linux Commands.pdf
 create mode 100644 linuxshell/Linux_shell_solutions.pdf
 create mode 100644 linuxshell/Shortcuts
 create mode 100644 linuxshell/a.patch
 create mode 100644 linuxshell/a.txt
 create mode 100644 linuxshell/a_sort.txt
 create mode 100644 linuxshell/b.txt
 create mode 100644 linuxshell/b_sort.txt
 create mode 100644 linuxshell/listOfFiles.txt
 create mode 100644 linuxshell/result.txt
 create mode 100644 linuxshell/sorted_result.txt
```

---

5. **5th question**

Added my_exercise.txt file in the root directory

---

6. **6th question**

Guidelines for commit

---

7. **7th question**

➔ add : add files content to the index. takes a modified file in your working directory and places the modified version in a staging area.

➔ commit: takes everything from the staging area and makes a permanent snapshot of the current state of your repository that is associated with a unique identifier.

```
ignitarium@IGN-BLR-LP-215:~/git/linuxshell$ git add 8.sh
ignitarium@IGN-BLR-LP-215:~/git/linuxshell$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   8.sh

ignitarium@IGN-BLR-LP-215:~/git/linuxshell$ git commit -m "file 8.sh is modified"
[master 872087d] file 8.sh is modified
 1 file changed, 2 insertions(+)
ignitarium@IGN-BLR-LP-215:~/git/linuxshell$ git status
On branch master
nothing to commit, working tree clean
ignitarium@IGN-BLR-LP-215:~/git/linuxshell$
ignitarium@IGN-BLR-LP-215:~/git/linuxshell$
```

➔ rm : used to remove individual or group of files

➔ log: Git log is a utility tool to review and read a history of everything that happens to a repository

```
ignitarium@IGN-BLR-LP-215:~/git/linuxshell$ git log
commit 872087d58e690875f70484d9026a81a0b4983553 (HEAD -> master)
Author: VegiMohnish <vegimohnish@gmail.com>
Date:   Wed Jan 5 18:33:04 2022 +0530

    file 8.sh is modified

commit 29e0e53d35faeb6ba452f8ad2f73492054a28c36 (origin/master)
Author: VegiMohnish <vegimohnish@gmail.com>
Date:   Tue Jan 4 22:30:09 2022 +0530

    my training exercises
ignitarium@IGN-BLR-LP-215:~/git/linuxshell$
```

➔ show: Show various types of objects

➔ diff: Show changes between commits, commit and working tree, etc

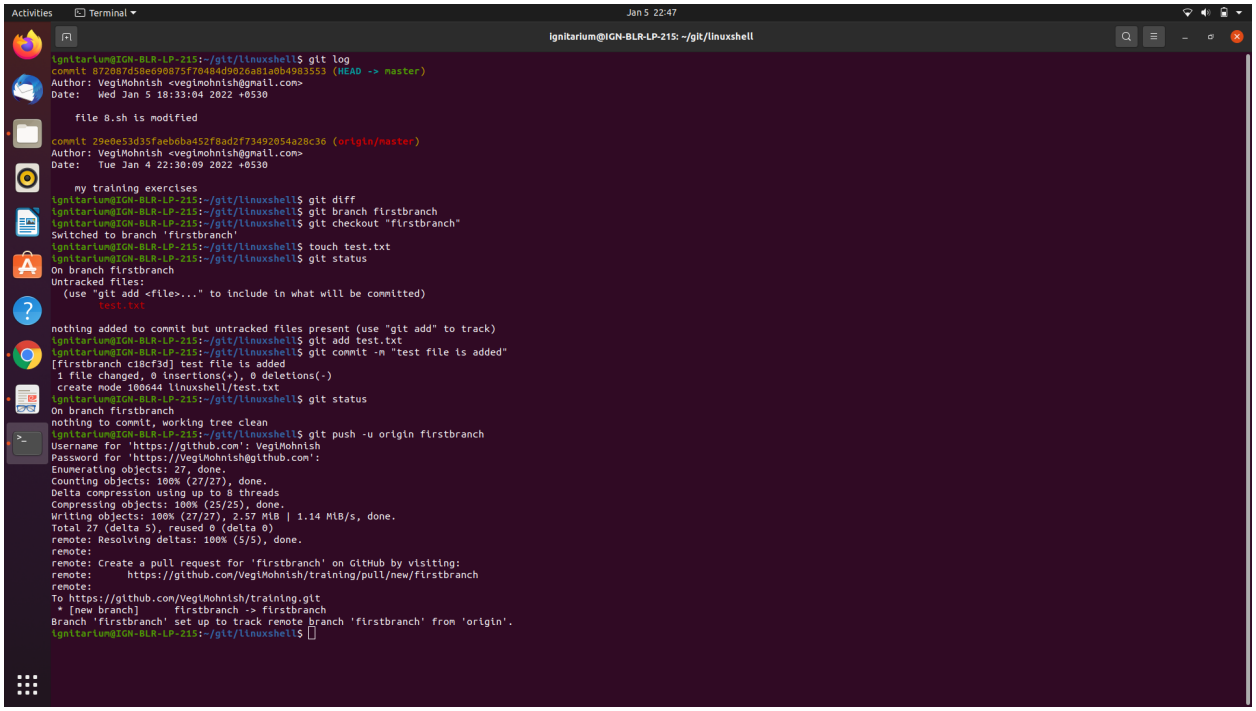
➔ push: The git push command is used to upload local repository content to a remote repository.

Referred link:

https://www.earthdatascience.org/workshops/intro-version-control-git/basic-git-commands/

---

## 8. 8th question




---

## 9. 9th question

1. git log --merge

The git log --merge command helps to produce the list of commits that are causing the conflict

2. git diff

The git diff command helps to identify the differences between the states repositories or files

3. git checkout

The git checkout command is used to undo the changes made to the file, or for changing branches

4. git reset --mixed

The git reset --mixed command is used to undo changes to the working directory and staging area

5. git merge --abort

The git merge --abort command helps in exiting the merge process and returning back to the state before the merging began

6. git reset

The git reset command is used at the time of merge conflict to reset the conflicted files to their original state

---

**10. 10th question**

**git fetch** is the command that tells your local git to retrieve the latest meta-data info from the original (yet doesn't do any file transferring. It's more like just checking to see if there are any changes available).
**git pull** on the other hand does that AND brings (copy) those changes from the remote repository.

Referred this: https://www.freecodecamp.org/news/git-fetch-vs-pull/

---

**11. 11th question**

.gitignore file -> text file that tells git which files or folders to ignore in the project.

Referred this for .gitignore :
https://www.freecodecamp.org/news/gitignore-what-is-it-and-how-to-add-to-repo/

.gitattribute file -> simple text file that gives attributes to pathnames

Referred this for .gitattribute: https://git-scm.com/docs/gitattributes

---

**12. 12th question**

Referred this link : https://git-scm.com/docs/git-config

***THANK YOU***