1) What are the different approach to operate git account ?
   We can access it using console and command prompt.Console is slower than compared to command prompt

1) Introduction to git

Git is a distributed version control system used to track changes in source code during software development.
It helps:

- Track file changes
- Collaborate with teams
- Maintain version history
- Manage branches

2) Installation and Configuration

1) **Check git version**

   git - - version

2) **Configuring User Details**

   git config - - global user.name “username” (name and email from our git account)
   git config - - global user.email “useremail”

3) **Checking configuration**

   Git config - - list

3) Create a Local Repository

1) **Create a project folder**
   mkdir directoryname
   Cd directoryname

2) **Initialize Git Repository**
   git init
   This creates a .git hidden folder to track changes

Check Status: git status

4) Commit To a Local Repository

1) **Create a file**
   touch app.py

2) **Add file to staging area**
   git add filename

Or git add . to add all the files in the folder

3) **Commit the file**
   Git commit -m “commit message”

5) Create a Remote Repository

Remote repositories are typically created on platform like

- GitHub
- GitLab
- Bitbucket

Steps to create Repository

- Login to github account
- profile>repository
- Click new repository>enter repo name >create

6) Push to a Remote Repositary

Connect Local repo to remote repo

```
git remote add origin https://github.com/nivethaprabhakaran/Gitpractice.git
```

Check remote

```
git  remote -v
```

Push Code

```
git push -u origin main
```

After first push next time simply git push

7) Git References(Important Commands)

Command	Description
git status	Check current status
git add	Add file to staging
git commit	Save changes
git log	View commit history
git diff	View file changes
git branch	List branches
git checkout	Switch branch

git merge	Merge branches
git pull	Fetch+ merge remote changes
git clone	Copy remote repository

8) Branching and Merging

Branching allows parallel development without affecting main code

Create New Branch

```
git branch branchname
```

Switch to branch:

```
git checkout branchname
```

Or (short cut):

```
git checkout -b branchname (creates and switch to the new branch)
```

Make Changes and Commit

```
git add.
Git commit -m “Added new feature”
```

Merge Branch into Main

Switch to main branch

```
git checkout main
```

Merge:

```
git merge branchname
```

Delete Branch(optional)

```
git branch -d branchname
```

9) Basic Git Workflow

```
Working Directory
          |
Staging Area (git add)
          |
Local Repository(git commit)
          |
Remote Repository(git push)
```