2.

1. Git –version
2. Git init
3. Git clone
4. Git status
5. Git branch
6. Git branch -r
7. Git checkout -b <new-branch-name>
8. Git add .
9. Git commit -m “”
10. Git push -u origin main
11. Git pull origin main
12. Git log
13. Git log –online
14. Git config –global user.name ‘Riya’
15. Git config –global user.email ‘riyaindap2000’
16. Git branch -d branch_name
17. Git clone “”
18. Git fetch origin main
19. Git merge origin/main
20. Git stash
21. Git stash apply
22.

3.

Make repo
Git init
Make folder
Git remote add origin “repo_link”
Git remote show origin
git checkout -b main
(make few txt files in folder)Git push origin main
Git pull origin main
(makefew txt files in rpo)git fetch origin main
Git merge origin/main

4.

```
public class MathOperations {
   public static void main(String[] args) {
      int num1 = Integer.parseInt(args[0]);
      int num2 = Integer.parseInt(args[1]);
         int result=num1+num2;
         System.out.println("The sum is:"+result);

}
```

}

5.

6.

(python factorial)

```
num = int(input("Enter a number: "))
factorial = 1
if num < 0:
   print(" Factorial does not exist for negative numbers")
elif num == 0:
   print("The factorial of 0 is 1")
else:
   for i in range(1,num + 1):
      factorial = factorial*i
   print("The factorial of",num,"is",factorial)
```

(sum of numbers)

```
num = int(input("Enter a number: "))

if num < 0:
   print("Enter a positive number")
else:
   sum = 0
   # use while loop to iterate un till zero
   while(num > 0):
      sum += num
      num -= 1
   print("The sum is",sum)
```

7.

8.

Pipeline script consisting stages and parameters

```
pipeline{
agent any
stages{
stage ('Plan phase'){
steps{
echo 'Hi. This is Shree Jaswal'
}
}
```

```
stage ('code phase'){
steps{
input('Do you want to continue?')
}
}
stage ('integrate phase'){
when{
not{
branch "master"
}
}
steps {
echo 'Integration test passed'

}
}
stage ('testing phase'){
parallel{
stage ('unit test'){
steps{
echo 'running unit test'
}
}
}
}
}
}
```

9.

11.

```
FROM ubuntu:latest
MAINTAINER "Shree Jaswal"
RUN apt update -y
RUN apt install nginx -y
EXPOSE 80
COPY index.html /usr/share/nginx/html/index.html
COPY index.html /var/www/html/index.html
CMD ["nginx","-g","daemon off;"]
```

Docker build -t riyanginx
Docker run -it -p 8888:80 riyanginx

13.
File and Directory Management:

- `ls`: List files and directories.
- `cd <directory>`: Change directory.
- `mkdir <directory>`: Create a new directory.
- `rm <file>`: Delete a file.
- `rm -r <directory>`: Delete a directory recursively.
- `cp <source> <destination>`: Copy files or directories.
- `mv <source> <destination>`: Move or rename files or directories.
- `touch <file>`: Create an empty file.

File Viewing and Editing:

- `cat <file>`: View file contents.
- `nano <file>` / `vi <file>`: Edit a file with text editors.
- `less <file>`: View file contents page by page.
- `grep <pattern> <file>`: Search for a pattern within a file.

System and User Management:

- `pwd`: Display the current directory path.
- `chmod <permissions> <file>`: Change file permissions.
- `chown <user>:<group> <file>`: Change file ownership.
- `top`: Display running processes and system resource usage.
- `ps`: List currently running processes.
- `sudo <command>`: Execute a command with superuser privileges.
- `df -h`: Display disk usage.

Networking:

- `ping <host>`: Check connectivity to a host.

14.
Basic Commands:

- `docker --version`: Check Docker version.
- `docker run <image>`: Run a container from an image.
- `docker ps`: List running containers.
- `docker stop <container>` / `docker start <container>`: Stop or start a container.

- `docker rm <container>`: Remove a container.
- `docker images`: List images.
- `docker pull <image>`: Pull an image from Docker Hub.
- `docker rmi <image>`: Remove an image.
- `docker build -t <image> <path>`: Build an image from a Dockerfile.

Advanced Commands:

- `docker exec -it <container> <command>`: Run a command in a container.
- `docker logs <container>`: View container logs.
- `docker inspect <container>`: Inspect container details.
- `docker network create <network>`: Create a custom network.
- `docker commit <container> <image>`: Create an image from a container.
- `docker export -o <file>.tar <container>`: Export container to tar file.
- `docker import <file>.tar`: Import an image from tar.
- `docker system prune`: Clean up unused Docker data.