2. Accept a sentence from the user and Count the number of words in the sentence and Display the words in uppercase separated by commas.

```
#!/bin/bash

# Prompt user to enter a sentence
read -p "Enter a sentence: " sentence

# Count number of words
word_count=$(echo "$sentence" | wc -w)

# Convert sentence to uppercase and replace spaces with commas
uppercase_commas=$(echo "$sentence" | tr '[:lower:]' '[:upper:]' | sed 's/ /,/g')

# Display results
echo "Number of words: $word_count"
echo "Words in uppercase separated by commas: $uppercase_commas"
```

----

3. Create a text file name sunbeam.txt and insert the following line.

```
Failed to connect to server
Connection established successfully
User admin logged in
Failed login attempt detected
Server restarted at 3AM
```

1. Extract all lines containing words that start with "est"?
Ans - `grep -o "\best\w*" sunbeam.txt`

2. Find lines containing the word "User" and display only the usernames (second word in the line)
Ans- `grep "^User" sunbeam.txt | awk '{print $2}'`

3. Show lines that end with a time (e.g., "3AM")
Ans- `grep "[0-9]\+AM$" sunbeam.txt`

4. Find lines that contain words with double letters (e.g., "ss", "ll")
Ans- `grep -E "\b\w*(.)\1\w*\b" sunbeam.txt`

5. Print lines that contain both the words "login" and "Failed"
Ans- `grep "login" sunbeam.txt | grep "Failed"`

Git Commands

1. Initialize a local Git repository.

```
git init
```

2. Create and commit a new file intro.txt using VIM editor, in the current branch. Use commit message: "initial commit with intro".

```
Ans- vim intro.txt      # (Add some text, save and exit)
git add intro.txt
git commit -m "initial commit with intro"
```

3. Create a new branch named feature1 and switch to it.
Ans- `git checkout -b feature1`

4. On feature1, create and commit file feature.txt with commit message: "feature added in feature1".
Ans- vim feature.txt   # (Add some content)
git add feature.txt
git commit -m "feature added in feature1"

5. Create another branch feature2 from master. Create and commit file module.txt with message: "added module in feature2".
Ans- git checkout master
git checkout -b feature2
vim module.txt   # (Edit file)
git add module.txt
git commit -m "added module in feature2"

6. Merge both branches (feature1 and feature2) into the master branch.
Ans- git checkout master
git merge feature1
git merge feature2

7. Show the branching graph using terminal
Ans- git log --oneline --graph --all

8. Rename the master branch to main.
Ans- git branch -m master main

9. In the repository, list all files that were changed in the last 2 commits and show the total number of lines added and removed.
Ans- git diff --stat HEAD~2 HEAD
git diff --numstat HEAD~2 HEAD

10. Show the difference between the last two commits.
Ans- git diff HEAD~1 HEAD