1. Start with a local Git repository having application code.

```
PS D:\lab\Git> cd GitDemo
PS D:\lab\Git\GitDemo> dir -Force


    Directory: D:\lab\Git\GitDemo


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d--h--        2/23/2019   1:24 PM                .git
-a----        2/23/2019   1:24 PM             21 hello.py


PS D:\lab\Git\GitDemo> type hello.py
print ("Hello World")
```

---

2. Set the identity of developer globally so that the same will be used across multiple local git repositories in different folders in developers machine.

```
PS D:\lab\Git\GitDemo> git config --global user.email
prakash.ramamurthy@wipro.com

PS D:\lab\Git\GitDemo> git config --global user.name prakaram

PS D:\lab\Git\GitDemo> type C:\Users\ACER\.gitconfig
[user]
        name = prakaram
        email = prakash.ramamurthy@wipro.com
```

---

3. Create new branch named “newfeature”

```
PS D:\lab\Git\GitDemo> git branch
* master

PS D:\lab\Git\GitDemo> git branch newfeature

PS D:\lab\Git\GitDemo> git branch
* master
  newfeature

PS D:\lab\Git\GitDemo> git checkout newfeature
Switched to branch 'newfeature'
```

---

4. Make change to the application code and commit the same on new branch.

```
PS D:\lab\Git\GitDemo> git branch
  master
* newfeature

PS D:\lab\Git\GitDemo> notepad .\hello.py

PS D:\lab\Git\GitDemo> type .\hello.py
print ("Hello World")
print ("Adding new feature")

PS D:\lab\Git\GitDemo> git status -s
 M hello.py

PS D:\lab\Git\GitDemo> git add .\hello.py

PS D:\lab\Git\GitDemo> git commit -m "newfeature"
[newfeature 4cfd68e] newfeature
 1 file changed, 2 insertions(+), 1 deletion(-)

PS D:\lab\Git\GitDemo> git log --pretty=format:"%h %cd | %s%d
[%an]" --date=short
4cfd68e 2019-02-23 | newfeature (HEAD -> newfeature) [Prakash]
6739347 2019-02-23 | first commit (origin/master, origin/HEAD,
master) [Prakash]
```

---

5. Verify to find that this change applies only on the commit made in “newfeature” branch, while the code and its versions on “master” branch remain intact.

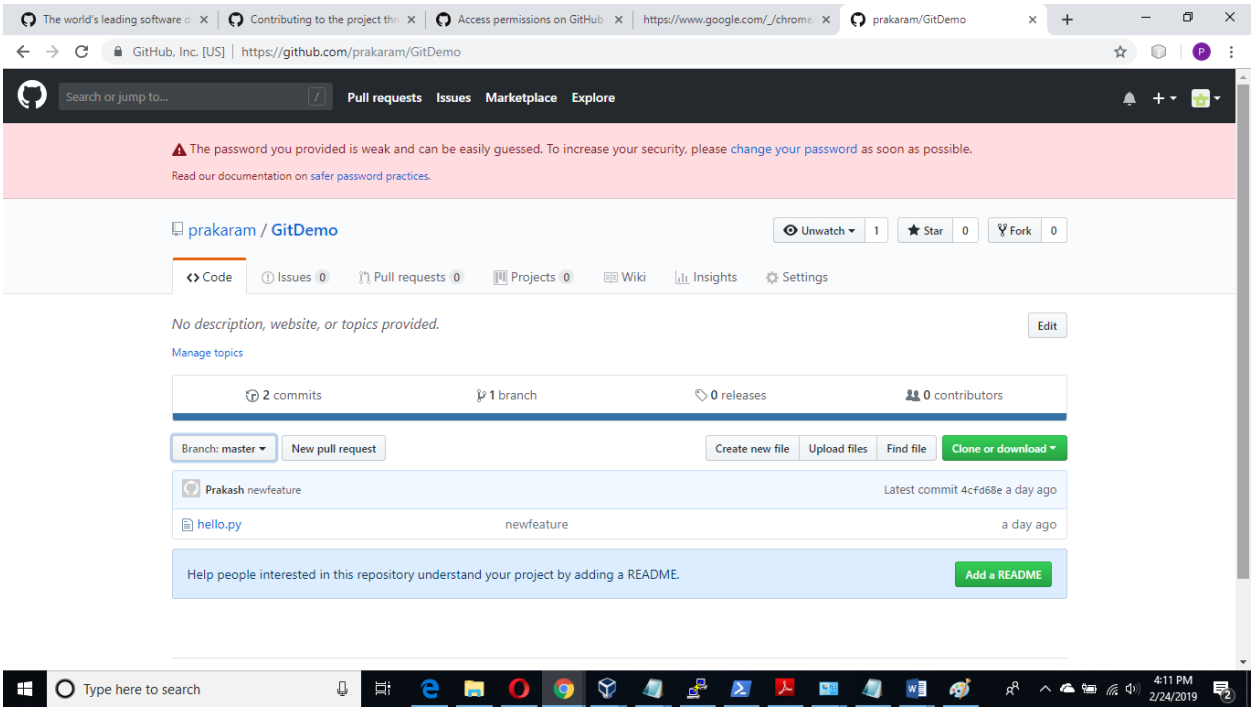
```
PS D:\lab\Git\GitDemo> type hello.py
print ("Hello World")
print ("Adding new feature")

PS D:\lab\Git\GitDemo> git branch
  master
* newfeature

PS D:\lab\Git\GitDemo> git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.

PS D:\lab\Git\GitDemo> git branch
* master
  newfeature

PS D:\lab\Git\GitDemo> type .\hello.py
print ("Hello World")
```

6. Now merge the changes made in newfeature branch to master branch. You must be in master branch before executing merge command.

```
PS D:\lab\Git\GitDemo> git branch
* master
  Newfeature

PS D:\lab\Git\GitDemo> git merge newfeature
Updating 6739347..4cfd68e
Fast-forward
 hello.py | 3 ++-
 1 file changed, 2 insertions(+), 1 deletion(-)

PS D:\lab\Git\GitDemo> git branch
* master
  newfeature

PS D:\lab\Git\GitDemo> type .\hello.py
print ("Hello World")
print ("Adding new feature")
```

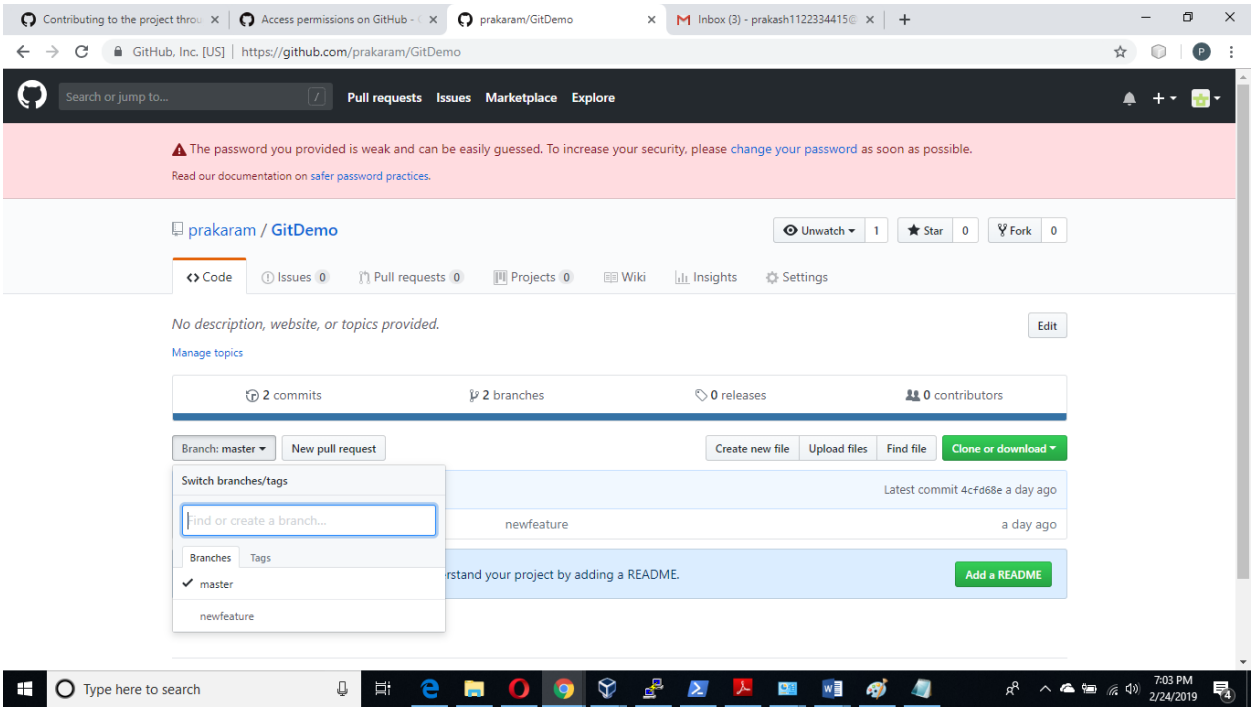
7. Push the changes made to remote repository. Following command push master branch which has commits from newfeature branch. Same is visible in the screenshot below.

```
PS D:\lab\Git\GitDemo> git push origin master
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 283 bytes | 283.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/prakaram/GitDemo.git
   6739347..4cfd68e  master -> master
```

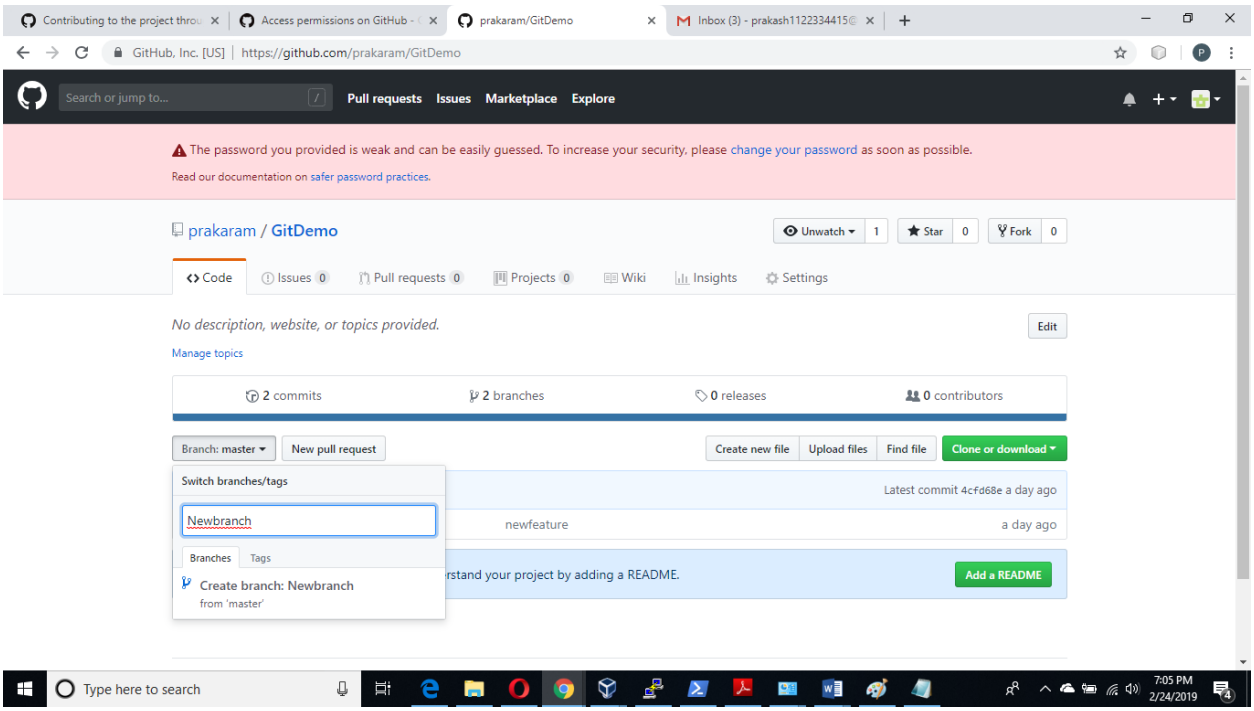


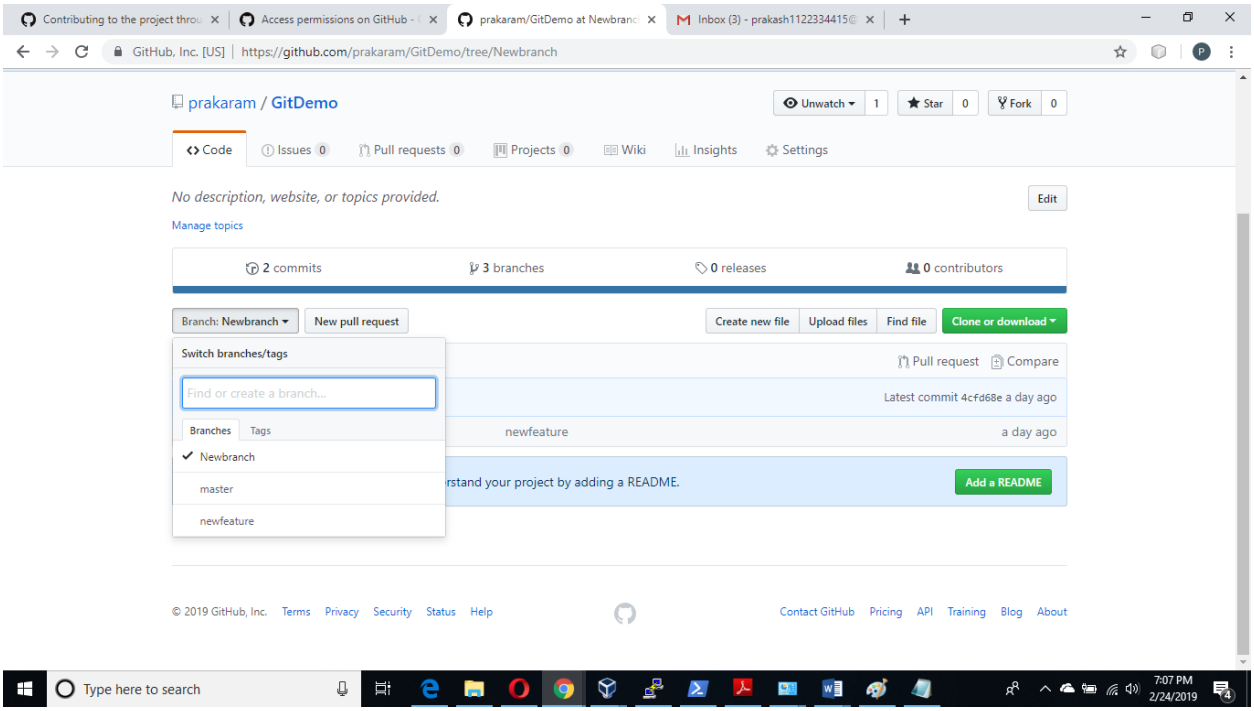
8. You may also push “newfeature” branch to remote repository. This is visible in the screenshot below.

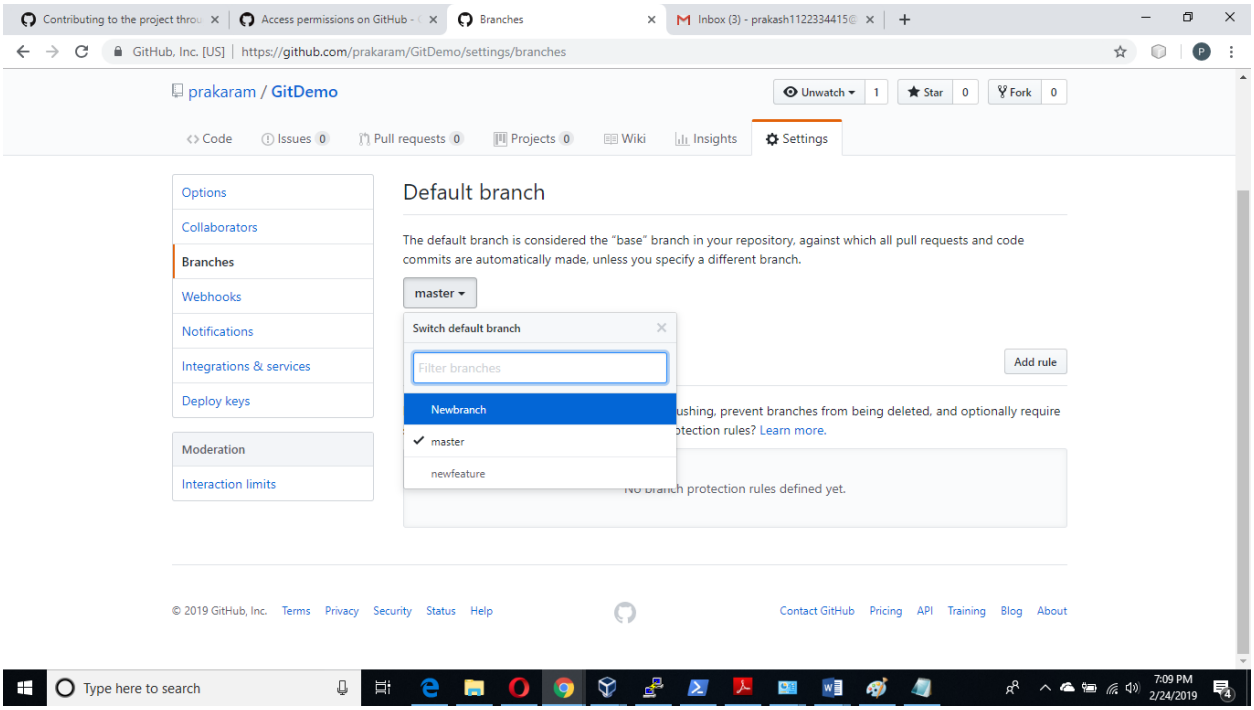
```
PS D:\lab\Git\GitDemo> git push origin newfeature
Total 0 (delta 0), reused 0 (delta 0)
remote:
remote: Create a pull request for 'newfeature' on GitHub by
visiting:
remote:
https://github.com/prakaram/GitDemo/pull/new/newfeature
remote:
To https://github.com/prakaram/GitDemo.git
 * [new branch]      newfeature -> newfeature
```



9. Similarly you may also create a “Newbranch” on the remote repository, and can push or pull the code with this branch. This is visible in the below screenshots.




Contributing to the project throu
Access permissions on GitHub -
prakaram/GitDemo at Newbranc
Inbox (3) - prakash1122334415@
GitHub, Inc. [US] | https://github.com/prakaram/GitDemo/tree/Newbranch
prakaram / GitDemo
Unwatch 1
Star 0
Fork 0
Code
Issues 0
Pull requests 0
Projects 0
Wiki
Insights
Settings
No description, website, or topics provided.
Edit
Manage topics
2 commits
3 branches
0 releases
0 contributors
Branch: Newbranch
New pull request
Create new file
Upload files
Find file
Clone or download
Switch branches/tags
Find or create a branch...
Branches
Tags
Newbranch
master
newfeature
Pull request
Compare
Latest commit 4cfd68e a day ago
newfeature
a day ago
rstand your project by adding a README.
Add a README
© 2019 GitHub, Inc.
Terms
Privacy
Security
Status
Help
Contact GitHub
Pricing
API
Training
Blog
About
Type here to search
7:07 PM
2/24/2019


Contributing to the project throu
Access permissions on GitHub -
Branches
Inbox (3) - prakash1122334415@
GitHub, Inc. [US] | https://github.com/prakaram/GitDemo/settings/branches
prakaram / GitDemo
Unwatch 1
Star 0
Fork 0
Code
Issues 0
Pull requests 0
Projects 0
Wiki
Insights
Settings
Options
Collaborators
Branches
Webhooks
Notifications
Integrations & services
Deploy keys
Moderation
Interaction limits
Default branch
The default branch is considered the "base" branch in your repository, against which all pull requests and code commits are automatically made, unless you specify a different branch.
master
Switch default branch
Filter branches
Newbranch
master
newfeature
Add rule
ushing, prevent branches from being deleted, and optionally require
otection rules? Learn more.
No branch protection rules defined yet.
© 2019 GitHub, Inc.
Terms
Privacy
Security
Status
Help
Contact GitHub
Pricing
API
Training
Blog
About
Type here to search
7:09 PM
2/24/2019

10. Let us now check for possibility of conflict. Switch to newfeature branch and make a small change and commit.

```
PS D:\lab\Git\GitDemo> type hello.py
print ("Hello World")
print ("Adding new feature")

PS D:\lab\Git\GitDemo> git branch
* master
  newfeature

PS D:\lab\Git\GitDemo> git checkout newfeature
Switched to branch 'newfeature'

PS D:\lab\Git\GitDemo> git branch
  master
* newfeature

PS D:\lab\Git\GitDemo> notepad hello.py
PS D:\lab\Git\GitDemo> type .\hello.py
print ("Hello World")
print ("Adding new feature")
print ("Adding another feature")

PS D:\lab\Git\GitDemo> git commit -a -m "another feature"
[newfeature 50248fa] another feature
 1 file changed, 2 insertions(+), 1 deletion(-)
```

---

11. While the branch is being modified, let us come back to master branch and make a modification before merging newfeature branch.

```
PS D:\lab\Git\GitDemo> git branch
  master
* newfeature

PS D:\lab\Git\GitDemo> git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.

PS D:\lab\Git\GitDemo> type hello.py
print ("Hello World")
print ("Adding new feature")

PS D:\lab\Git\GitDemo> git branch
* master
  newfeature

PS D:\lab\Git\GitDemo> notepad hello.py
```

```
PS D:\lab\Git\GitDemo> type hello.py
print ("Hello World")
print ("Adding new feature")
print ("Additional line on master")

PS D:\lab\Git\GitDemo> git commit -a -m "another line"
[master 0aa9270] another line
 1 file changed, 2 insertions(+), 1 deletion(-)
```

---

12. Examine both the branches to watch possibility of conflict.

```
PS D:\lab\Git\GitDemo> git log --pretty=format:"%h %cd |
%s%d [%an]" --date=short
0aa9270 2019-02-25 | another line (HEAD -> master)
[Prakash]
4cfd68e 2019-02-23 | newfeature (origin/newfeature,
origin/master, origin/HEAD) [Prakash]
6739347 2019-02-23 | first commit [Prakash]

PS D:\lab\Git\GitDemo> git branch
* master
  newfeature

PS D:\lab\Git\GitDemo> type hello.py
print ("Hello World")
print ("Adding new feature")
print ("Additional line on master")

PS D:\lab\Git\GitDemo> git checkout newfeature
Switched to branch 'newfeature'

PS D:\lab\Git\GitDemo> git branch
  master
* newfeature

PS D:\lab\Git\GitDemo> type hello.py
print ("Hello World")
print ("Adding new feature")
print ("Adding another feature")
```

---

13. Let us now switch to master branch and try to merge newfeature branch as we did before.

```
PS D:\lab\Git\GitDemo> git checkout master
Switched to branch 'master'
Your branch is ahead of 'origin/master' by 1 commit.
  (use "git push" to publish your local commits)
```

```
PS D:\lab\Git\GitDemo> git branch
* master
  newfeature

PS D:\lab\Git\GitDemo> git merge newfeature
Auto-merging hello.py
CONFLICT (content): Merge conflict in hello.py
Automatic merge failed; fix conflicts and then commit the
result.
```

---

14. Watch which lines are in conflict. Once the conflict happen, unless this merge-conflict is resolved you can not switch branch.

```
PS D:\lab\Git\GitDemo> type hello.py
print ("Hello World")
print ("Adding new feature")
<<<<<<< HEAD
print ("Additional line on master")
=======
print ("Adding another feature")
>>>>>>> newfeature

PS D:\lab\Git\GitDemo> git checkout newfeature
error: you need to resolve your current index first
hello.py: needs merge
```

---

15. Abort the merge effort so that you can switch branch to correct your code.

```
PS D:\lab\Git\GitDemo> git merge --abort

PS D:\lab\Git\GitDemo> git branch
* master
  newfeature

PS D:\lab\Git\GitDemo> type hello.py
print ("Hello World")
print ("Adding new feature")
print ("Additional line on master")

PS D:\lab\Git\GitDemo> git checkout newfeature
Switched to branch 'newfeature'

PS D:\lab\Git\GitDemo> git branch
  master
* newfeature

PS D:\lab\Git\GitDemo> type hello.py
```

```
print ("Hello World")
print ("Adding new feature")
print ("Adding another feature")
```

---

16. Let us switch back to master branch and shall merge keeping “ours”, that would retain content modified on master branch and reject changes from newfeature branch. Alternately you may also try with “theirs”

```
PS D:\lab\Git\GitDemo> git checkout master
Switched to branch 'master'
Your branch is ahead of 'origin/master' by 1 commit.
  (use "git push" to publish your local commits)

PS D:\lab\Git\GitDemo> git branch
* master
  newfeature

PS D:\lab\Git\GitDemo> git merge -s ours newfeature
Merge made by the 'ours' strategy.
PS D:\lab\Git\GitDemo> cat hello.py
print ("Hello World")
print ("Adding new feature")
print ("Additional line on master")
```

---

17. Let us examine the way to avoid such possibilities of merge conflicts.

```
PS D:\lab\Git\GitDemo> git branch
* master
  newfeature

PS D:\lab\Git\GitDemo> ls


    Directory: D:\lab\Git\GitDemo


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----        2/25/2019  12:44 AM             88 hello.py


PS D:\lab\Git\GitDemo> type hello.py
print ("Hello World")
print ("Adding new feature")
print ("Additional line on master")
```

---

18. Let the original code on master be available in a separate file “file_on_master”.

```
PS D:\lab\Git\GitDemo> notepad file_on_master

PS D:\lab\Git\GitDemo> type .\file_on_master.txt
This file has original code

PS D:\lab\Git\GitDemo> git add .\file_on_master.txt

PS D:\lab\Git\GitDemo> git commit -m "file on master"
[master 69227df] file on master
 1 file changed, 1 insertion(+)
 create mode 100644 file_on_master.txt

PS D:\lab\Git\GitDemo> git branch
* master
  Newfeature

PS D:\lab\Git\GitDemo> ls


    Directory: D:\lab\Git\GitDemo


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----        2/25/2019   1:11 AM             27 file_on_master.txt
-a----        2/25/2019  12:44 AM             88 hello.py
```

---

19. Let us switch to newfeature branch and code the new feature in a separate file.

```
PS D:\lab\Git\GitDemo> git checkout newfeature
Switched to branch 'newfeature'

PS D:\lab\Git\GitDemo> git branch
  master
* newfeature

PS D:\lab\Git\GitDemo> notepad added_feature
PS D:\lab\Git\GitDemo> type .\added_feature.txt
This file has code of added feature

PS D:\lab\Git\GitDemo> git add .\added_feature.txt

PS D:\lab\Git\GitDemo> git commit -m "added feature"
[newfeature e0dd229] added feature
 1 file changed, 1 insertion(+)
 create mode 100644 added_feature.txt
```

```
PS D:\lab\Git\GitDemo> ls


    Directory: D:\lab\Git\GitDemo


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----        2/25/2019   1:13 AM             35 added_feature.txt
-a----        2/25/2019   1:13 AM             85 hello.py
```

---

20. Watch that master and newfeature branches have code available in separate files.

```
PS D:\lab\Git\GitDemo> git branch
* master
  Newfeature

PS D:\lab\Git\GitDemo> ls


    Directory: D:\lab\Git\GitDemo


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----        2/25/2019   1:11 AM             27 file_on_master.txt
-a----        2/25/2019  12:44 AM             88 hello.py

PS D:\lab\Git\GitDemo> git branch
  master
* newfeature

PS D:\lab\Git\GitDemo> ls


    Directory: D:\lab\Git\GitDemo


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----        2/25/2019   1:13 AM             35 added_feature.txt
-a----        2/25/2019   1:13 AM             85 hello.py
```

---

21. Switch to master branch and merge the content from newfeature branch.

```
PS D:\lab\Git\GitDemo> git checkout master
Switched to branch 'master'
Your branch is ahead of 'origin/master' by 4 commits.
```

```
  (use "git push" to publish your local commits)

PS D:\lab\Git\GitDemo> ls


    Directory: D:\lab\Git\GitDemo


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----        2/25/2019   1:15 AM             27 file_on_master.txt
-a----        2/25/2019   1:15 AM             88 hello.py


PS D:\lab\Git\GitDemo> git branch
* master
  newfeature

PS D:\lab\Git\GitDemo> git merge newfeature
Merge made by the 'recursive' strategy.
 added_feature.txt | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 added_feature.txt

PS D:\lab\Git\GitDemo> ls


    Directory: D:\lab\Git\GitDemo


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----        2/25/2019   1:16 AM             35 added_feature.txt
-a----        2/25/2019   1:15 AM             27 file_on_master.txt
-a----        2/25/2019   1:15 AM             88 hello.py
```