[CS200]- Software Tool and Technologies Lab-II **16-09-2020**

Homework: 2

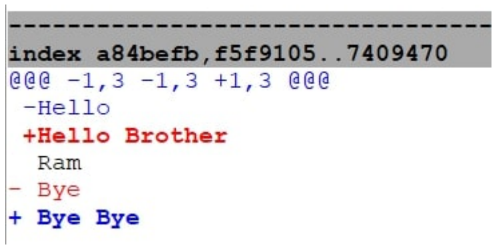
Team Name: Peaky Blinders Roll Numbers:11940150, 11940370, 11940380

Solution of Problem 3:

Git commands for repository ”Repo” and folder name ”q3BetterMerge” where Merge is better:-

```
cd
cd Repo/q3BetterMerge                          //Going to the location of file name ”q1” in repository ”Repo”
git init                                         //Initialization of git
touch sample.txt                                   //Creating new file with name ’sample’
echo ”Hello>Ram>Bye”>sample.txt          //Filling the file with content ”Hello in line-1, Ram in 2 and Bye in 3”
git add .                                       //Adding files to the repository
git commit -m ”3 Lines added to Master”          //Commiting 1st time
git checkout -b Branch                   //Adding a new feature to a new branch
echo ”Hello Brother>Ram>Bye”>sample.txt        //Adding some changes to the 1st line”
git add .                                       //Adding changes to the repository
git commit -m ”Brother Added to Line-1”          //Commiting 2nd time
git checkout master                    // Switched to ’master’
echo ”Hello>Ram>Bye Bye” >sample.txt        //Adding some changes to the 3rd line”
git add .                                       //Adding changes to the repository
git commit -m ”Bye Added to Line-3”          //Commiting 3rd time
git checkout Branch                    // Switched to ’Branch’
git merge master                     //Auto-merging made by the ’recursive’ strategy.
git graph                              //Showing Final Graph Graph (shown below in Figure-1.2 and Figure-1.3)
```

Brief Explaination:-
Rebasing and merging are both designed to integrate changes from one branch into another branch but in different ways.
The merge will result as a combination of commits,whereas rebase will add all the changes in feature branch starting from the last commit of the master branch.Merging takes the contents of the feature branch and integrates it with the master branch. As a result, only the master branch is changed.
So,if we want to see the history completely same as it happened,we should use merge. Merge preserves history whereas rebase rewrites it.
In the above Example, 1st Master file contains ’Hello’,’Ram’,’Bye’ -each in 1 line.Then a feature branch added to change 1st line as ’Hello Brother’ and also simultaneously master changed 3rd line to ’Bye Bye’. So, feature branch merged to master without any conflict since changes were in different lines and it became ’Hello Brother’,’Ram’,’Bye Bye’ -each in 1 line (shown in Figure-1.1). Hence, Merge is better here as it preserved the whole history.



```
--------------------------------
index a84befb,f5f9105..7409470
@@@ -1,3 -1,3 +1,3 @@@
 -Hello
 +Hello Brother
  Ram
- Bye
+ Bye Bye
```

Figure 1.1: All Changes while Merging

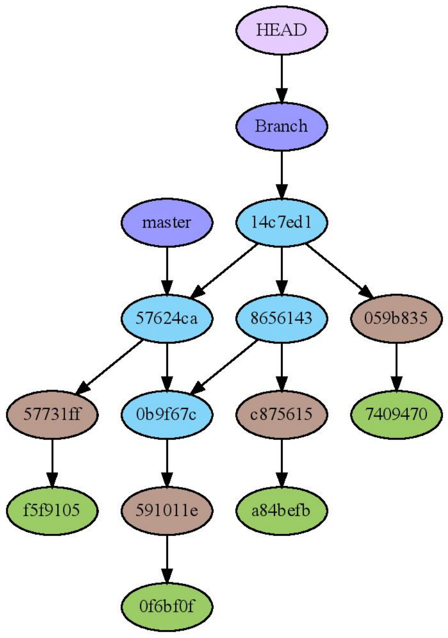
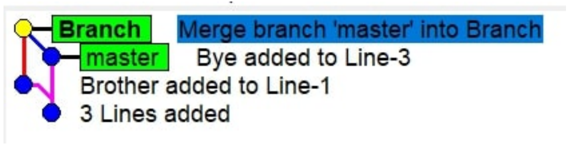


Figure 1.2: Final Git Graph after Merging



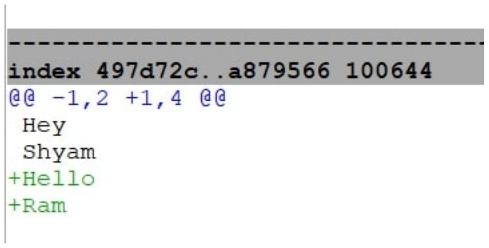
Branch Merge branch 'master' into Branch
master Bye added to Line-3
Brother added to Line-1
3 Lines added

Figure 1.3: Final Git Graph after Merging

Git commands for repository ”Repo” and folder name ”q3BetterRebase” where Rebase is better:-

```
cd
cd Repo/q3BetterRebase                              //Going to the location of file name ”q1” in repository ”Repo”
git init                                              //Initialization of git
touch sample.txt                                        //Creating new file with name ’sample’
echo ”Hi”>sample.txt            //Filling the file with content ”Hi”
git add .                                            //Adding files to the repository
git commit -m ”Hi added to Master”                 //Commiting 1st time
git checkout -b Branch                          //Adding a new feature to a new branch
echo ”Hello>Ram”>sample.txt           //Adding some changes to the 1st 2 lines”
git add .                                            //Adding changes to the repository
git commit -m ”Hello Ram Added to 1st 2 lines”              //Commiting 2nd time
git checkout master                          //Switched to ’master’
echo ”Hey>Shyam”>sample.txt           //Adding some changes to the 1st 2 lines”
git add .                                            //Adding changes to the repository
git commit -m ”Hey Shyam Added to 1st 2 lines”              //Commiting 3rd time
git checkout Branch                          // Switched to ’Branch’
git rebase master                          //Auto-merging made by the ’recursive’ strategy.
echo ”Hey>Shyam>Hello>Ram”>sample.txt          //Resolving Conflicts
git add .                                            //Adding changes to the repository
git rebase –continue                                  //Rebasing
git checkout master                         //Switched to master
git merge Branch                           //Rebasing to master
git graph                                  //Showing Final Graph Graph (shown below in Figure-1.5 and Figure-1.6)
```

Brief Explaination:-
When we do rebase a feature branch onto master, we move the base of the feature branch to master branch’s ending point.
Rebasing is better to streamline a complex history,we are able to change the commit history by interactive rebase.We can remove undesired commits, squash two or more commits into one or edit the commit message.
Rebase will present conflicts one commit at a time whereas merge will present them all at once. It is better and much easier to handle the conflicts but you shouldn’t forget that reverting a rebase is much more difficult than reverting a merge if there are many conflicts.
In the above Example, 1st Master file contains ’Hi’ in 1st line. Then a feature branch added to change 1st line as ’Hello’,’Ram’ -each in 1 line and also simultaneously master changed ’Hi’ to ’Hey’,’Shyam’ -each in 1 line. So when feature branch merged to master then conflict occurd since changes were in same lines. So using Rebasing , conflict resolved easily manually and it became ’Hey’,’Shyam’,’Hello’,’Ram’ -each in 1 line (shown in Figure-1.4). Hence, Rebase is better in this case.



```
---------------------------------
index 497d72c..a879566 100644
@@ -1,2 +1,4 @@
 Hey
 Shyam
+Hello
+Ram
```

Figure 1.4: All Changes while Rebasing

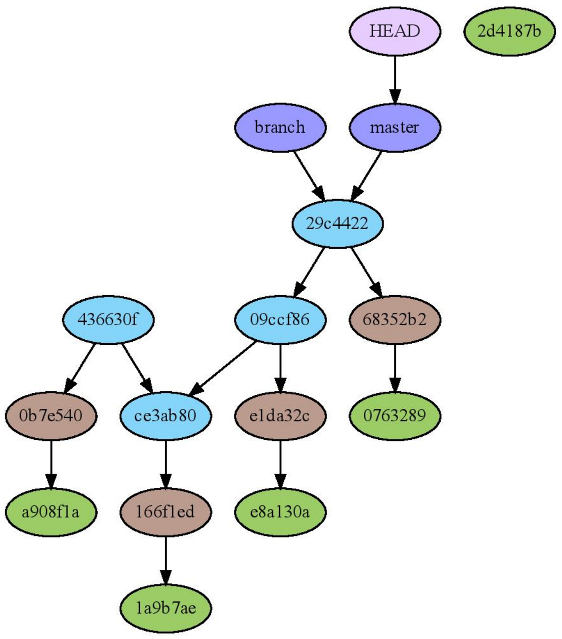
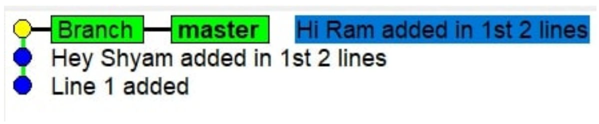


Figure 1.5: Final Git Graph after Rebasing



```
Branch master Hi Ram added in 1st 2 lines
Hey Shyam added in 1st 2 lines
Line 1 added
```

Figure 1.6: Final Git Graph after Rebasing