Source Code Management

Task 1.2

(CS181)

Submitted by

Name-Aadithya Roll No.- 2110990002

Department of Computer Science & Engineering

Chitkara University Institute of Engineering and Technology, Punjab

Jan- June
(2021-22)

CHITKARA UNIVERSITY

Institute/School Name	**Chitkara University Institute of Engineering and Technology**		
Department Name	**Department of Computer Science & Engineering**		
Programme Name	**Bachelor of Engineering (B.E.), Computer Science & Engineering**		
Course Name	**Source Code Management**	Session	**2021-22**
Course Code	**CS181**	Semester/Batch	**2nd/2021**
Vertical Name	**Beta**	Group No	G01
Course Coordinator	**Dr. Neeraj Singla**		
Faculty Name	**Mr. Monit Kapoor**		

CHITKARA UNIVERSITY

INDEX

CHITKARA UNIVERSITY

ADD COLLABORATORS ON GITHUB REPO

In GitHub, we can invite other GitHub users to become collaborators to our private repositories (which expires after 7 days if not accepted, restoring any unclaimed licenses). Being a collaborator of a personal repository you can pull (read) the contents of the repository and push (write) changes to the repository. You can add unlimited collaborators on public and private repositories.

Collaborators can perform a number of actions into someone else's personal repositories, they have gained access to. Some of them are,

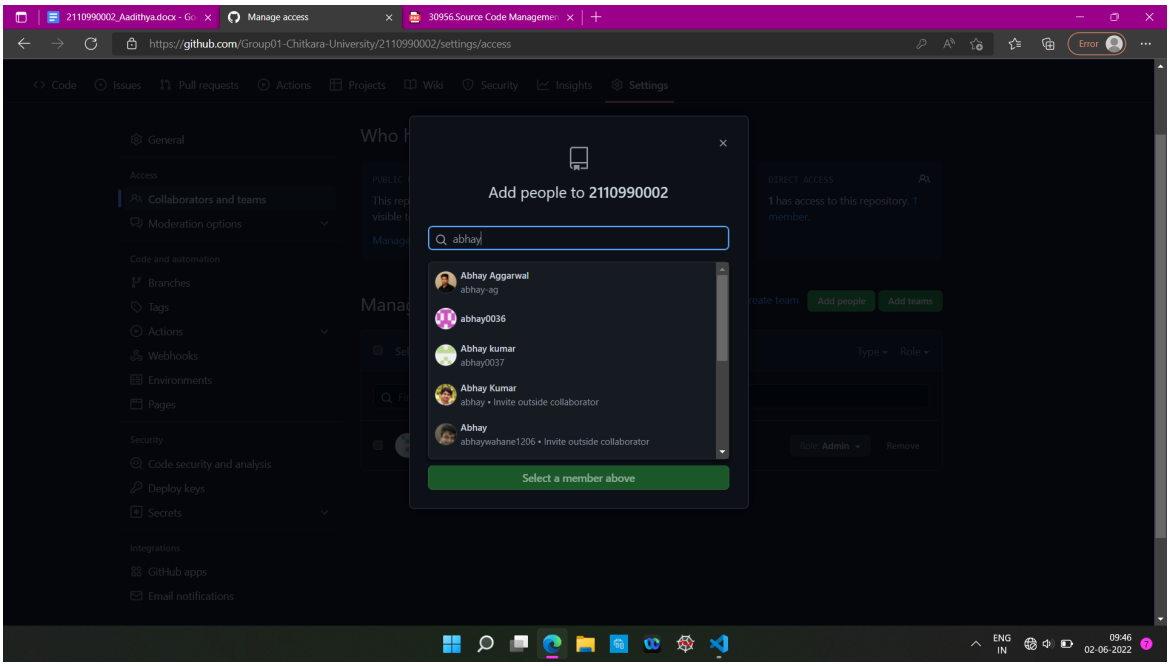
1. Create, merge, and close pull requests in the repository
2. Publish, view, install the packages
3. Fork the repositories
4. Make the changes on the repositories as suggested by the Pull requests.
5. Mark issues or pull requests as duplicate
6. Create, edit, and delete any comments on commits, pull requests, and issues in the repository
7. Removing themselves as collaborators on the repositories.
8. Manage releases in the repositories.

STEPS TO ADD COLLABORATORS:

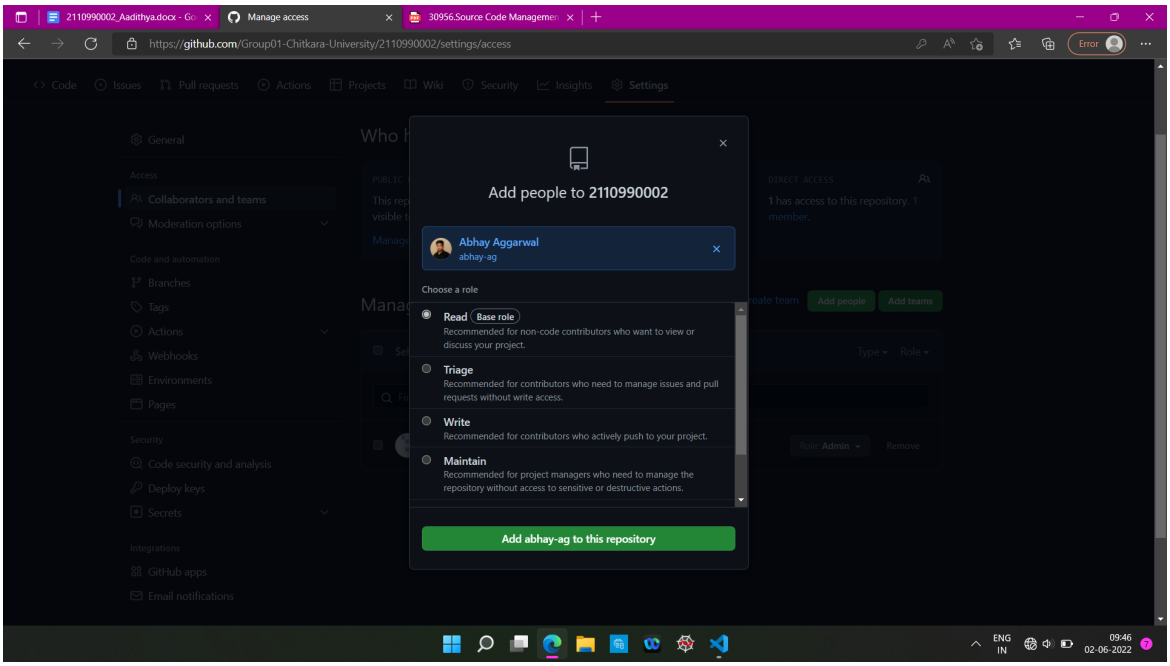
1. Navigate to the repository on Github you wish to share with your collaborator.
2. Click on the "Settings" tab on the right side of the menu at the top of the screen.

3.On the new page, click the "Collaborators" menu item on the left side of the page.

4. Start typing the new collaborator's GitHub username into the text box.



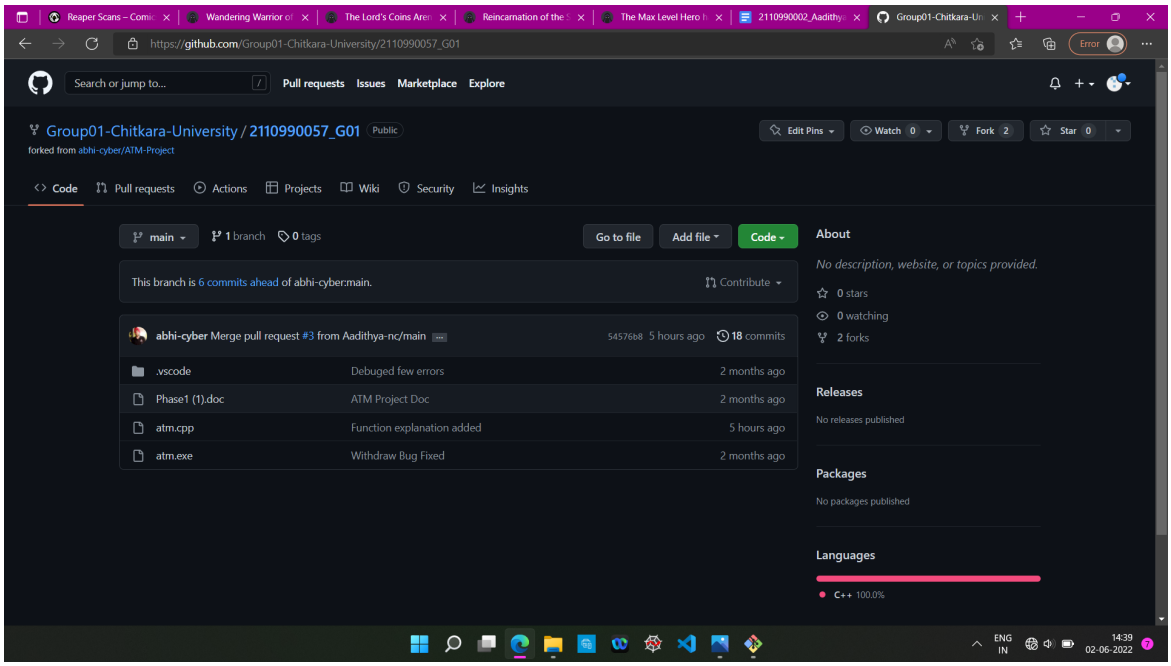
5. Select the GitHub user from the list that appears below the text box.
6. Click the "Add" button.



FORK AND COMMIT

A fork is a copy of a repository. Forking a repository allows you to freely experiment with changes without affecting the original project. Most commonly, forks are used to either propose changes to someone else's project to which you do not have write access, or to use someone else's project as a starting point for your own idea.

STEPS TO FORK A REPO-



1. Go to the repository that you wish to fork.

2.Click on the option ‘Fork’ in the top right corner.

3.You now have a forked repository.

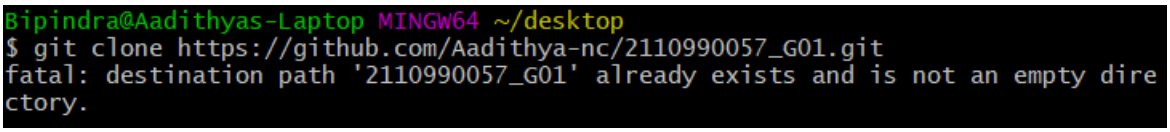
CLONING THE REPO INTO YOUR DEVICE

When you create a repository on GitHub.com, it exists as a remote repository. You can clone your repository to create a local copy on your computer and sync between the two locations.

1. Once you have forked the repository, you can clone it into your computer using directly the option given on github or through running git clone command in git bash.

2. Copy the URL of the forked repository

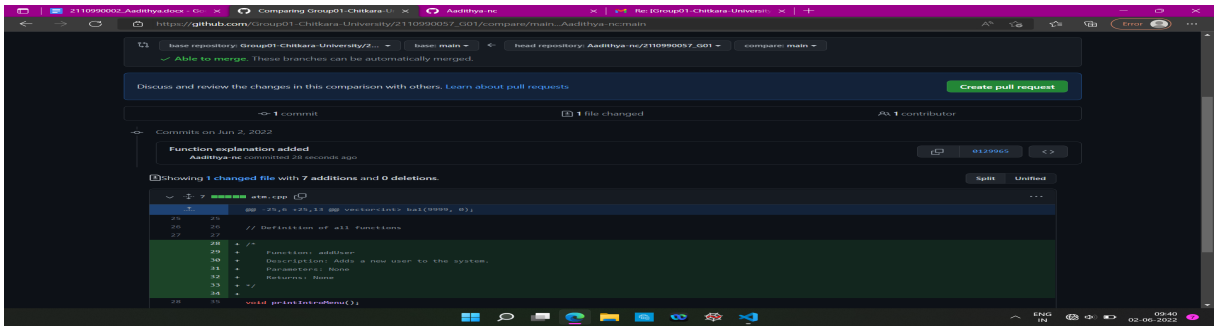
3. Open git bash and type the command " git clone <url of the forked repository>"

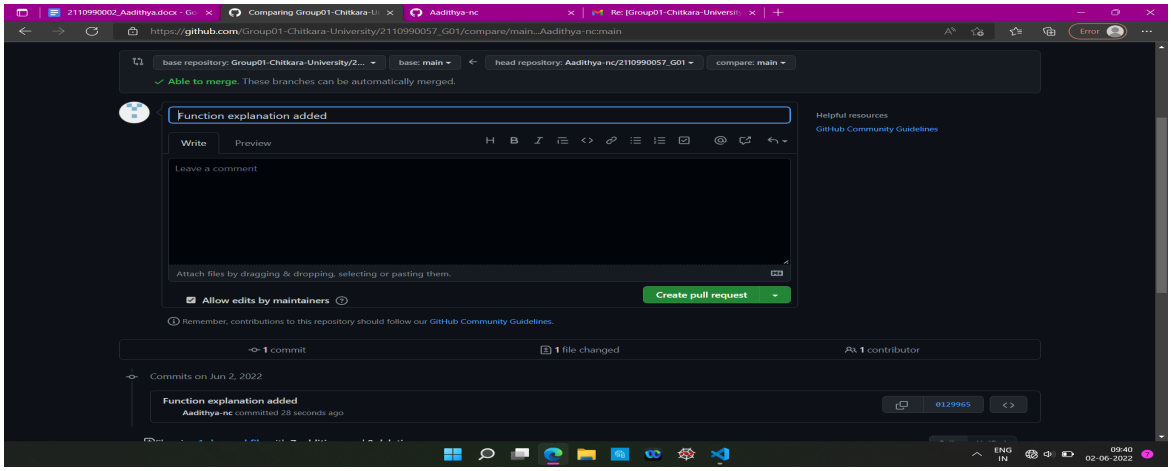


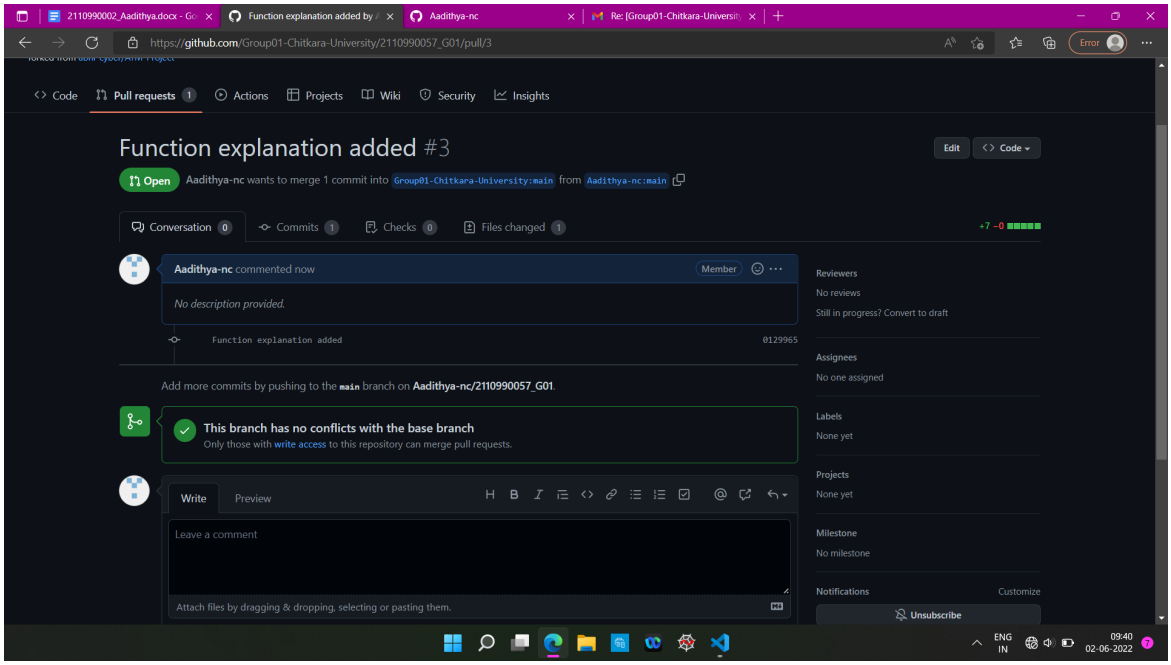
(here it says destination path already exists because i had cloned the repo few days before i took the screenshot)

COMMITING CHANGES TO THE FORKED REPOSITORY

1. Once you have cloned the repository you can introduce changes to it as per your wish.
2. After changing it you have to stage the file and then commit it.
3. After committing changes push it to your remote repository.




Function explanation added
Write
Preview
Leave a comment
Attach files by dragging & dropping, selecting or pasting them.
Allow edits by maintainers
Create pull request
Remember, contributions to this repository should follow our GitHub Community Guidelines.
1 commit
1 file changed
1 contributor
Commits on Jun 2, 2022
Function explanation added
Aadithya-nc committed 28 seconds ago
0129965


Code
Pull requests 1
Actions
Projects
Wiki
Security
Insights
Function explanation added #3
Edit
Code
Open
Aadithya-nc wants to merge 1 commit into Group01-Chitkara-University:main from Aadithya-nc:main
Conversation 0
Commits 1
Checks 0
Files changed 1
+7 −0
Aadithya-nc commented now
Member
No description provided.
Function explanation added
0129965
Add more commits by pushing to the main branch on Aadithya-nc/2110990057_G01.
This branch has no conflicts with the base branch
Only those with write access to this repository can merge pull requests.
Write
Preview
Leave a comment
Attach files by dragging & dropping, selecting or pasting them.
Reviewers
No reviews
Still in progress? Convert to draft
Assignees
No one assigned
Labels
None yet
Projects
None yet
Milestone
No milestone
Notifications
Customize
Unsubscribe

MERGE AND RESOLVE CONFLICTS CREATED DUE TO OWN ACTIVITY AND COLLABORATORS ACTIVITY

Merging and conflicts are a common part of the Git experience. Conflicts generally arise when two people have changed the same lines in a file, or if one developer deleted a file while another developer was modifying it. In these cases, Git cannot automatically determine what is correct. Conflicts only affect the developer conducting the merge, the rest of the team is unaware of the conflict. Git will mark the file as being conflicted and halt the merging process. It is then the developers' responsibility to resolve the conflict.

1.To understand the merging concept of branches, create a branch named “feature” in your repository.

```
Bipindra@Aadithyas-Laptop MINGW64 /a/git (master)
$ git branch
  feature
* master

Bipindra@Aadithyas-Laptop MINGW64 /a/git (master)
$ git switch feature
Switched to branch 'feature'

Bipindra@Aadithyas-Laptop MINGW64 /a/git (feature)
$ git log --oneline
3eb8771 (HEAD -> feature, origin/master, master) Corrected code, bugs
7648148 Corrected code, bugs
6e34cd4 fucked up some shit
e9ff7a2 removed 1 comment
820f6cc added main body, new comments
8dae878 added 4 lines of code to file
767065b first commit|g1 repo|demo for class
```

2.Here, there is a file called ‘day2.cpp’. Make changes to it, add and commit them.

```
Bipindra@Aadithyas-Laptop MINGW64 /a/git (feature)
$ vi first.cpp

Bipindra@Aadithyas-Laptop MINGW64 /a/git (feature)
$ git add first.cpp
warning: LF will be replaced by CRLF in first.cpp.
The file will have its original line endings in your working directory

Bipindra@Aadithyas-Laptop MINGW64 /a/git (feature)
$ git commit -m "made new program on file"
[feature 1f68c44] made new program on file
 1 file changed, 65 insertions(+), 10 deletions(-)
 rewrite first.cpp (91%)

Bipindra@Aadithyas-Laptop MINGW64 /a/git (feature)
$ git checkout master
Switched to branch 'master'
D       .first.cpp.swp
Your branch is up to date with 'origin/master'.

Bipindra@Aadithyas-Laptop MINGW64 /a/git (master)
$ ls
first.cpp  fork/  meh.html  meh10.html  meh2.html  meh3.html  meh4.html  meh5.html  meh6.html  meh6_files/  meh7.html  meh8.html  meh9.html

Bipindra@Aadithyas-Laptop MINGW64 /a/git (master)
$ vi first.cpp
```

3.Similarly, change the same lines of day2.cpp file in the master branch.

4.If you are not already on the branch that you want the other one to merged in (in this example master branch), then switch to it.

5.Using the command try merging feature branch into master branch using the “git merge <branch name>”

```
Bipindra@Aadithyas-Laptop MINGW64 /a/git (master)
$ git add first.cpp

Bipindra@Aadithyas-Laptop MINGW64 /a/git (master)
$ git commit -m "code corrected"
[master 37e402f] code corrected
 1 file changed, 68 insertions(+), 10 deletions(-)
 rewrite first.cpp (79%)
```

```
Bipindra@Aadithyas-Laptop MINGW64 /a/git (master)
$ git merge feature
Auto-merging first.cpp
CONFLICT (content): Merge conflict in first.cpp
Automatic merge failed; fix conflicts and then commit the result.
```

6.Auto merging fails and conflict arises. In order to resolve it we make use of the mergetool by running the command “git mergetool”. The mergetool editor will open.

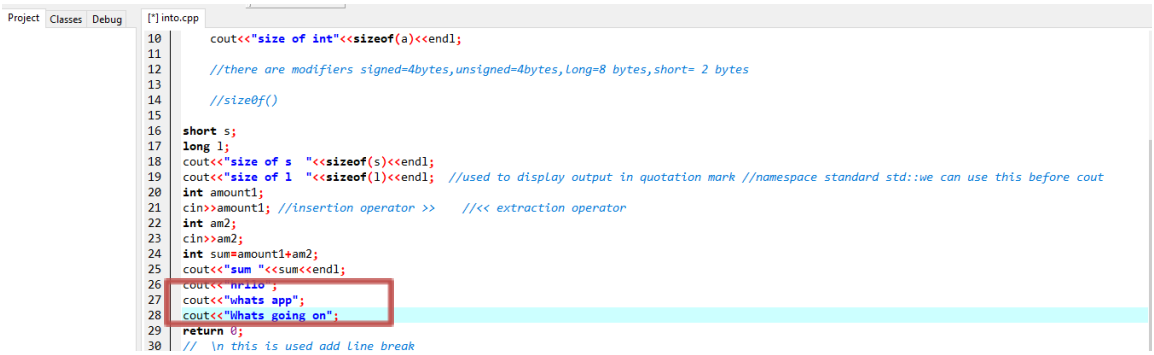
7.Make changes as per requirement in order to resolve the conflicts and exit the editor.

```
Bipindra@Aadithyas-Laptop MINGW64 /a/git (master|MERGING)
$ git status
On branch master
Your branch is ahead of 'origin/master' by 1 commit.
  (use "git push" to publish your local commits)

You have unmerged paths.
  (fix conflicts and run "git commit")
  (use "git merge --abort" to abort the merge)

Unmerged paths:
  (use "git add <file>..." to mark resolution)
        both modified:   first.cpp

Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        deleted:    .first.cpp.swp

no changes added to commit (use "git add" and/or "git commit -a")
```

RESET AND REVERT

While Working with Git in certain situations we want to undo changes in the working area or index area, sometimes remove commits locally or remotely and we need to reverse those changes. We can do it by using the git reset, git revert, git checkout commands.

RESET-
git reset is used when we want to unstage a file and bring our changes back to the working directory. Git reset can also be used to remove commits from the local repository.
Suppose we make edits to a file, stage it and commit it




```
Bipindra@Aadithyas-Laptop MINGW64 /a/git (master|MERGING)
$ git log
commit 37e402fc42ba5d12583926c3fb860b2022adda1e (HEAD -> master)
Author: Aadithya-nc <aadithya002.be21@chitkara@edu.in>
Date:   Thu Jun 2 15:30:16 2022 +0530

    code corrected

commit 3eb87710e2824d27ed8515d358806dafee7ee15e (origin/master)
Author: Aadithya-nc <aadithya002.be21@chitkara@edu.in>
Date:   Fri Apr 8 10:22:47 2022 +0530

    Corrected code, bugs

commit 7648148447261b5ff7e10b5298b1e9c8dd42c75a
Author: Aadithya-nc <aadithya002.be21@chitkara@edu.in>
Date:   Fri Apr 8 10:21:34 2022 +0530

    Corrected code, bugs
```

In order to reset the changes made in the recent commit, run the “git reset --hard HEAD~1” command. Or a command git “reset commit no.”

```
Bipindra@Aadithyas-Laptop MINGW64 /a/git (master|MERGING)
$ git reset
Unstaged changes after reset:
D       .first.cpp.swp
M       first.cpp

Bipindra@Aadithyas-Laptop MINGW64 /a/git (master)
$ git reset 37e402fc42ba5d12583926c3fb860b2022adda1e
Unstaged changes after reset:
D       .first.cpp.swp
M       first.cpp

Bipindra@Aadithyas-Laptop MINGW64 /a/git (master)
$ git log
commit 37e402fc42ba5d12583926c3fb860b2022adda1e (HEAD -> master)
Author: Aadithya-nc <aadithya002.be21@chitkara@edu.in>
Date:   Thu Jun 2 15:30:16 2022 +0530

    code corrected

commit 3eb87710e2824d27ed8515d358806dafee7ee15e (origin/master)
Author: Aadithya-nc <aadithya002.be21@chitkara@edu.in>
Date:   Fri Apr 8 10:22:47 2022 +0530

    Corrected code, bugs
```

The HEAD returns to the previous commit and the changes made are reset.

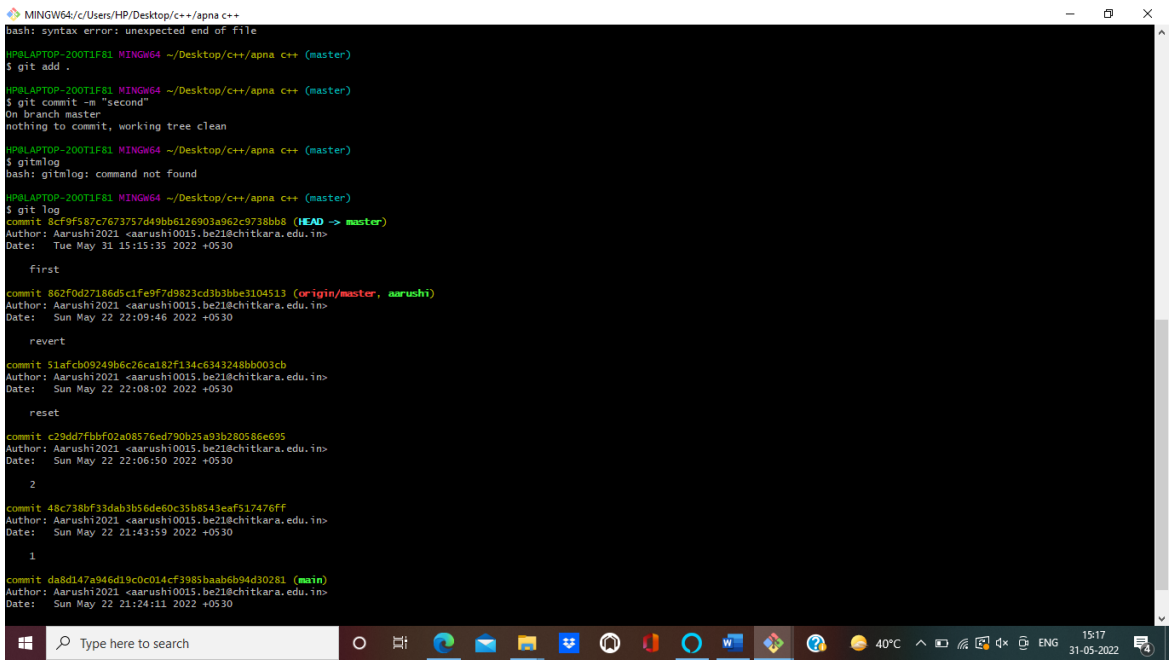
REVERT-

git revert is used to remove the commits from the remote repository. git revert removes the commit that we have done but adds one more commit which tells us that the revert has been done.

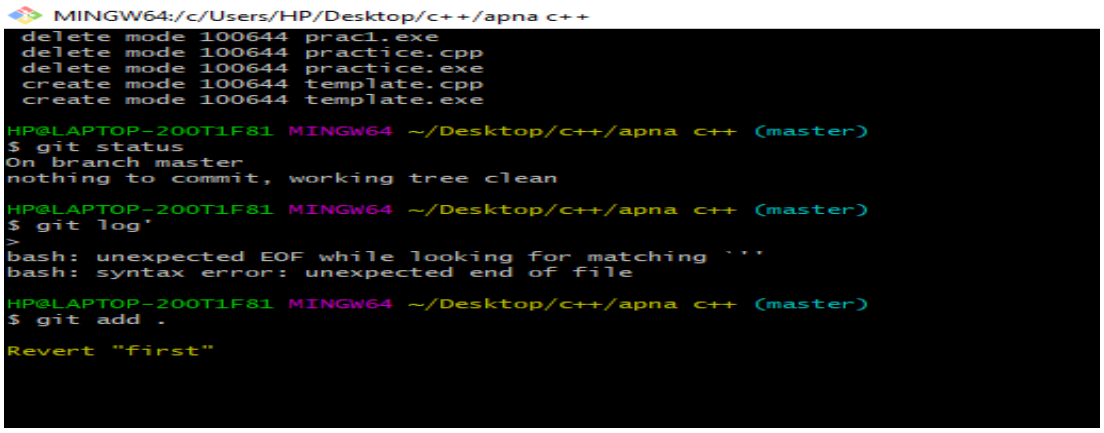
```
    //sizeOf()

short s;
long l;
cout<<"size of s  "<<sizeof(s)<<endl;
cout<<"size of l  "<<sizeof(l)<<endl;  //used to display output in quotation mark //namespace standard std::we can use this before cout
int amount1;
cin>>amount1; //insertion operator >>    //<< extraction operator
int am2;
cin>>am2;
int sum=amount1+am2;
cout<<"sum "<<sum<<endl;
cout<<"hrllo";


return 0;
```

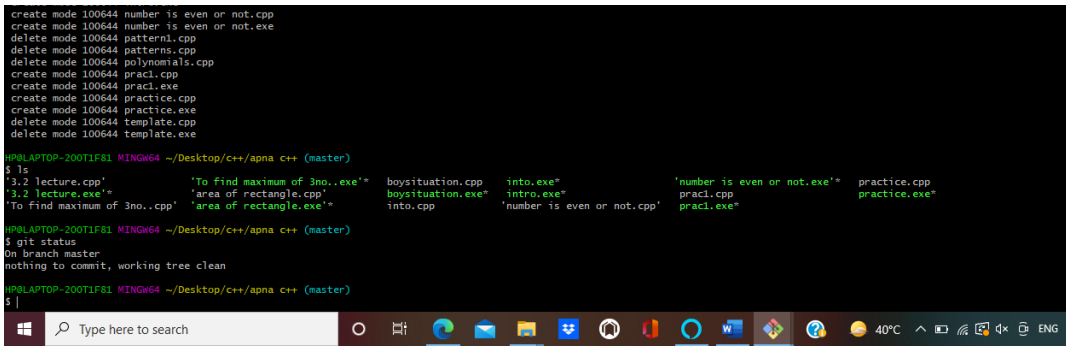
```
MINGW64:/c/Users/HP/Desktop/c++/apna c++
//      }
//      int space=2*n-2*i;;
//
//      for(int j=1;j<=space;j++){
//              cout<<" ";
//
//      }
//      for(int j=1;j<=i;j++){
//              cout<<"*";
//      }
//      cout<<endl;
//}
//
//for(int j=1;j<=i;j++){
//              cout<<"*";
//      }
//      int space=2*n-2*i;;
//
//      for(int j=1;j<=space;j++){
//              cout<<" ";
//
//      }
//      for(int j=1;j<=i;j++){
//              cout<<"*";
//      }
//      cout<<endl;
//}
//
//
//
//}

#include <iostream>
using namespace std;
int main(){
int n;
cin>>n;
for(int i=1;i<=n;i++){
        for (int j=1;j<=n+1-i;j++){
                cout<<j<<" ";
        }
        cout<<endl;
}
      cout<<"Hello world";
return 0;
}
pattern1.cpp[+] [dos] (18:30 28/05/2022)                                155,10 Bot
-- INSERT --
```

In order to understand it add changes to a file, stage and commit it.



Now to revert the changes made in the commit run the “git revert <commit id>” command.





You can see that a new commit as ‘revert “changes made”’ is there and the file has returned to its previous state.