Subject Name: **Source Code Management**

Subject Code: **CS181**

Course: **CSE 1st Year**

Submitted By:
Name: Pratyukt Nag
2210990679
G-27/A

Submitted To:
Dr. Aditi Moudgil
Department of Computer Science & Engineering
Chitkara University Institute of Engineering and Technology
Rajpura, Punjab

< *Index* >

CHITKARA
UNIVERSITY
PUNJAB

Experiment No. 06

Aim: Add collaborators on GitHub Repo

Theory:

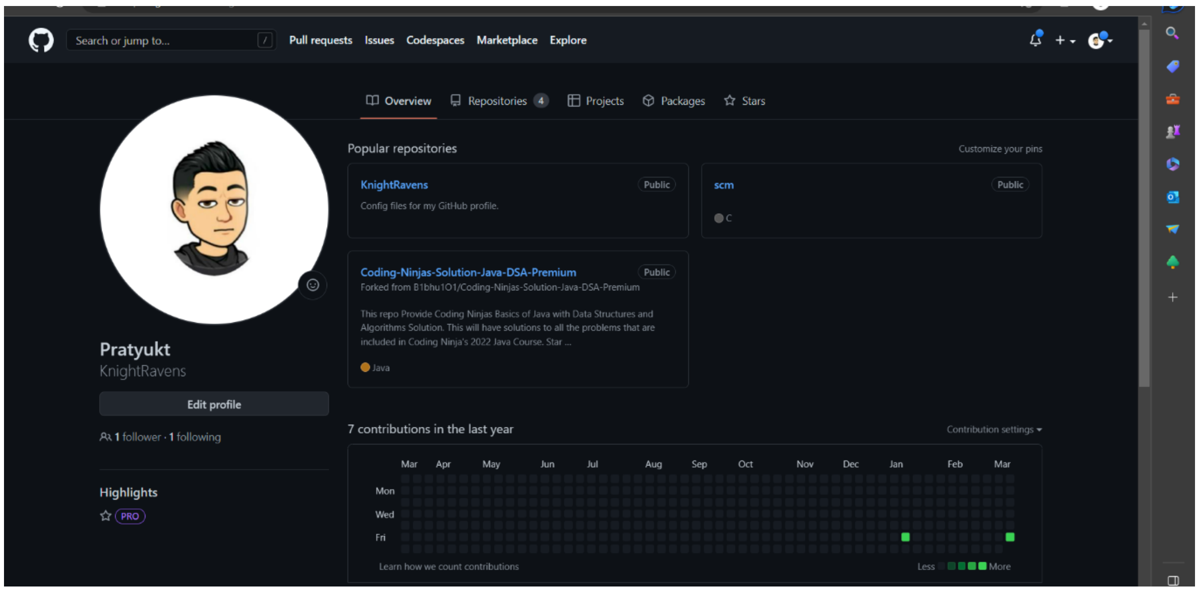
Whenever you make a repository in GitHub, not everyone has the permission to change or push codes into your repository. The users have a read-only access. In order to allow other individuals to make changes to your repository, you need to invite them to collaborate to the project.

GitHub also restricts the number of collaborators we can invite within a period of 24 hours. If we exceed the limit, then either we have to wait for 24-hours or we can also create an organization to collaborate with more people.

Being a collaborator, the user can create, merge and close pull requests in the repository. They can also remove them as the collaborator.

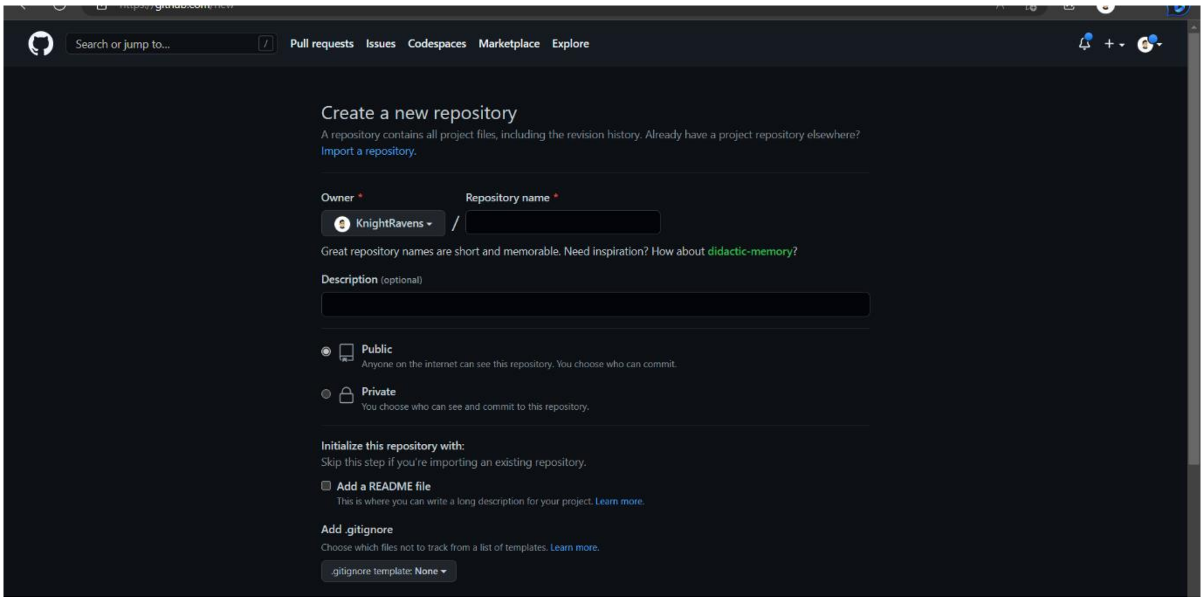
Procedure:

1. Login to your GitHub account and you will land on the home page as shown below. Click on Repositories option in the menu bar.



Search or jump to...
Pull requests
Issues
Codespaces
Marketplace
Explore
Overview
Repositories 4
Projects
Packages
Stars
Popular repositories
Customize your pins
KnightRavens
Public
Config files for my GitHub profile.
scm
Public
C
Coding-Ninjas-Solution-Java-DSA-Premium
Public
Forked from B1bhu101/Coding-Ninjas-Solution-Java-DSA-Premium
This repo Provide Coding Ninjas Basics of Java with Data Structures and Algorithms Solution. This will have solutions to all the problems that are included in Coding Ninja's 2022 Java Course. Star ...
Java
Pratyukt
KnightRavens
Edit profile
1 follower · 1 following
Highlights
PRO
7 contributions in the last year
Contribution settings
Mar Apr May Jun Jul Aug Sep Oct Nov Dec Jan Feb Mar
Mon
Wed
Fri
Learn how we count contributions
Less More

2. Click on the ‘New’ button in the top right corner.
3. Enter the Repository name and add the description of the repository.
4. Select if you want the repository to be public or private.



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Owner * KnightRavens / Repository name *

Great repository names are short and memorable. Need inspiration? How about didactic-memory?

Description (optional)

[✓ Public] Anyone on the internet can see this repository. You choose who can commit.

Private You choose who can see and commit to this repository.

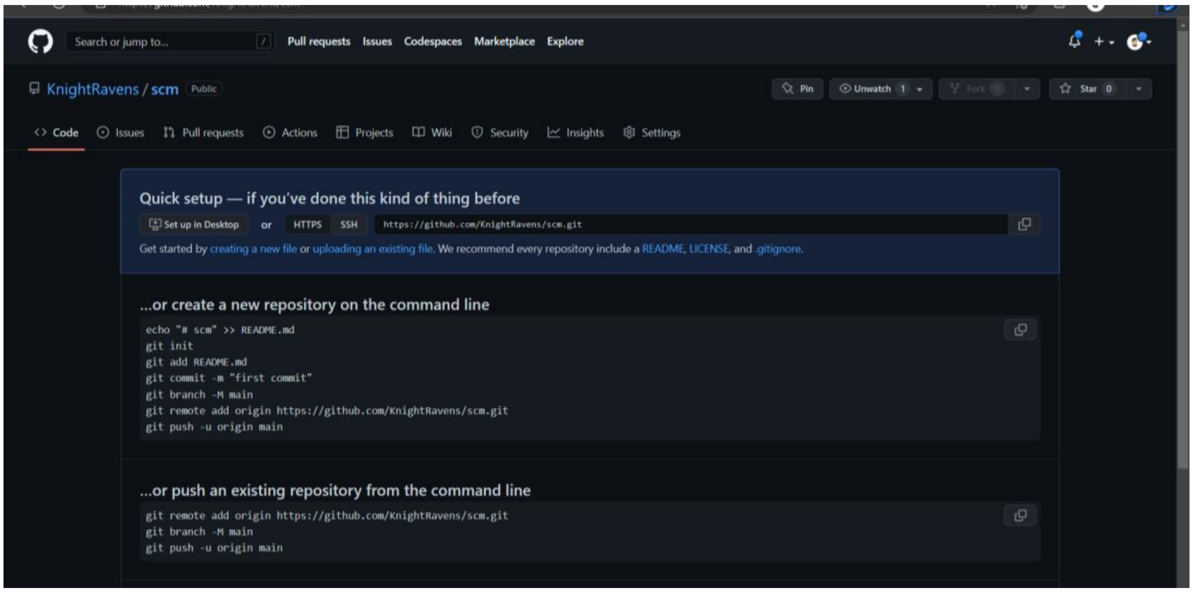
Initialize this repository with:
Skip this step if you're importing an existing repository.

☐ Add a README file
This is where you can write a long description for your project. Learn more.

Add .gitignore
Choose which files not to track from a list of templates. Learn more.

.gitignore template: None

5. If you want to import code from an existing repository select the import code option.



Quick setup — if you've done this kind of thing before

https://github.com/KnightRavens/scm.git

Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.

...or create a new repository on the command line

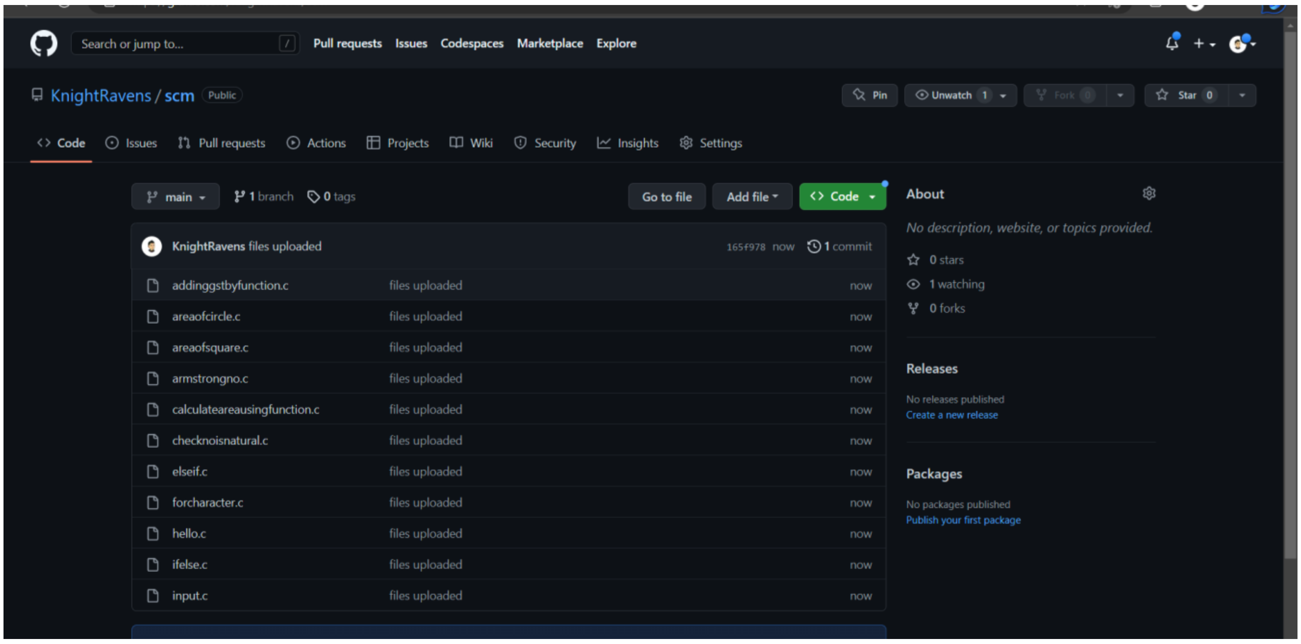
```
echo "# scm" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/KnightRavens/scm.git
git push -u origin main
```

...or push an existing repository from the command line

```
git remote add origin https://github.com/KnightRavens/scm.git
git branch -M main
git push -u origin main
```

6. Now, you have created your repository successfully.

7. To add collaborators to your repository, open your repository and select settings option in the navigation bar.



KnightRavens files uploaded 165f978 now 1 commit

addinggstbyfunction.c	files uploaded	now
areaofcircle.c	files uploaded	now
areaofsquare.c	files uploaded	now
armstrongno.c	files uploaded	now
calculateareausingfunction.c	files uploaded	now
checknoisnatural.c	files uploaded	now
elseif.c	files uploaded	now
forcharacter.c	files uploaded	now
hello.c	files uploaded	now
ifelse.c	files uploaded	now
input.c	files uploaded	now

About

No description, website, or topics provided.

0 stars
1 watching
0 forks

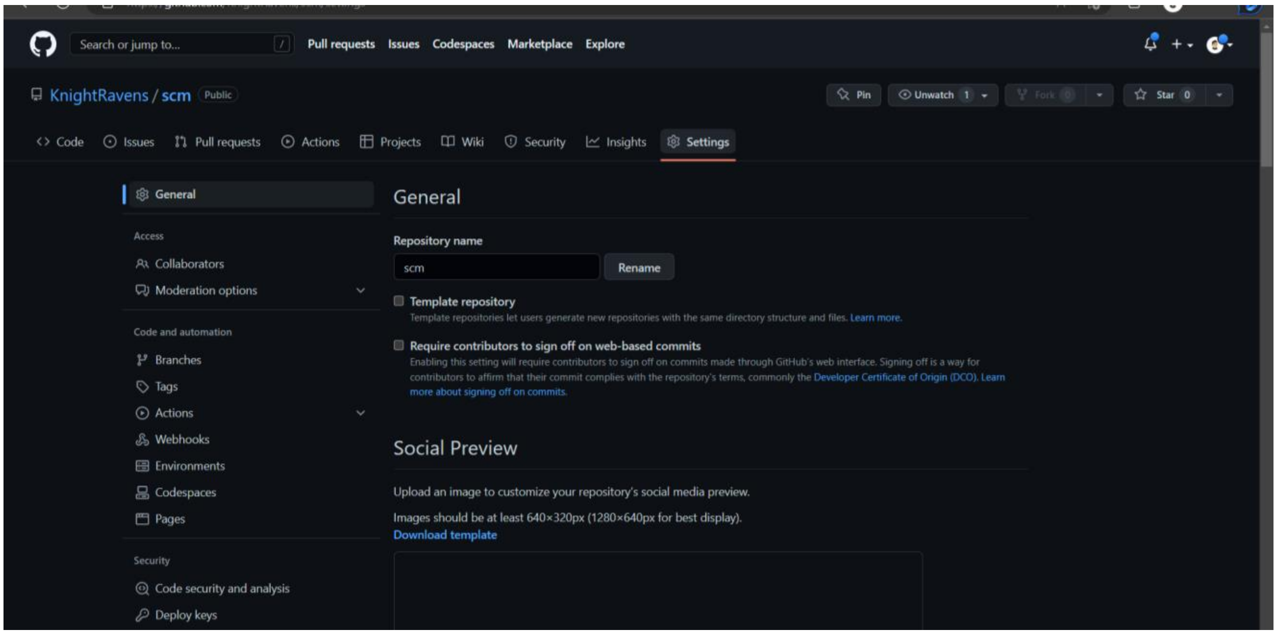
Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

8. Click on Collaborators option under the access tab.


Search or jump to...
Pull requests Issues Codespaces Marketplace Explore
KnightRavens / scm Public
Pin Unwatch 1 Fork 0 Star 0
Code Issues Pull requests Actions Projects Wiki Security Insights Settings
General
Access
Collaborators
Moderation options
Code and automation
Branches
Tags
Actions
Webhooks
Environments
Codespaces
Pages
Security
Code security and analysis
Deploy keys

General

Repository name

scm Rename

Template repository
Template repositories let users generate new repositories with the same directory structure and files. Learn more.

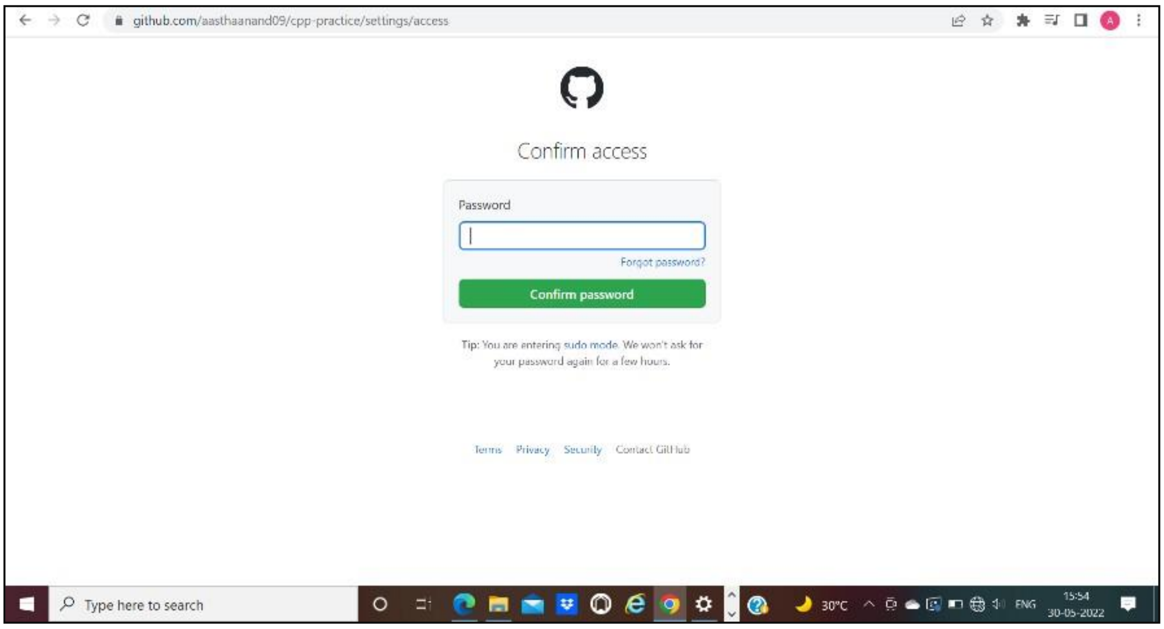
Require contributors to sign off on web-based commits
Enabling this setting will require contributors to sign off on commits made through GitHub's web interface. Signing off is a way for contributors to affirm that their commit complies with the repository's terms, commonly the Developer Certificate of Origin (DCO). Learn more about signing off on commits.

Social Preview

Upload an image to customize your repository's social media preview.

Images should be at least 640×320px (1280×640px for best display).
Download template

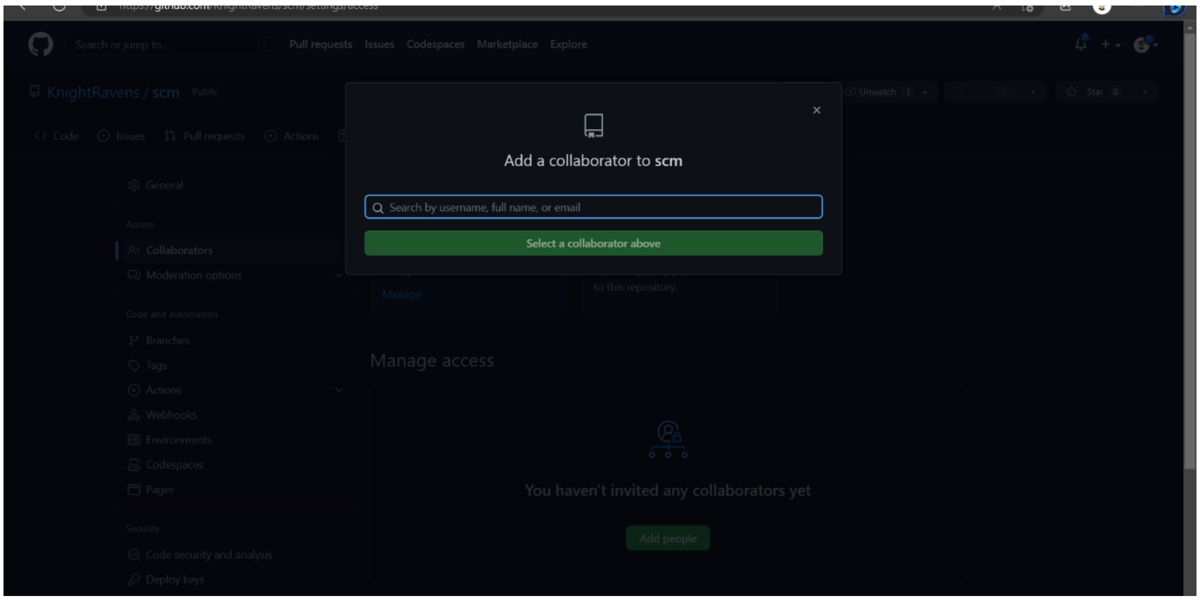
9. After clicking on collaborators, GitHub asks you to enter your password to confirm the access to the repository.


github.com/aasthaanand09/cpp-practice/settings/access
Confirm access
Password
Forgot password?
Confirm password
Terms Privacy Security Contact GitHub
Type here to search
30°C ENG 15:54 30-05-2022

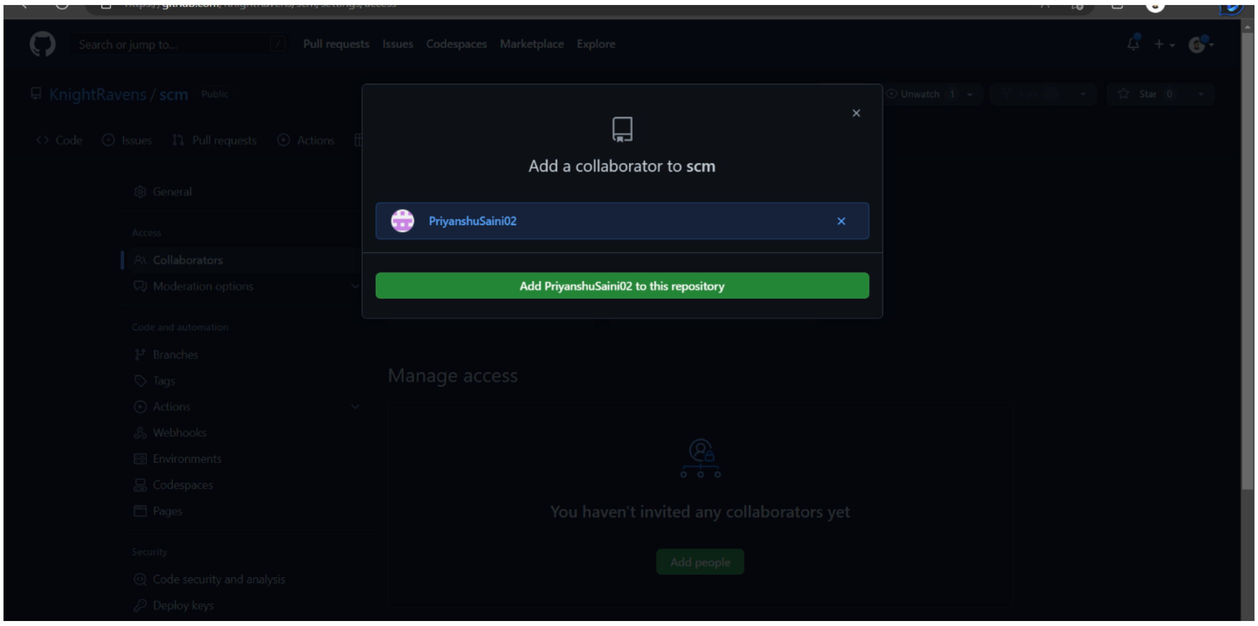
Tip: You are entering sudo mode. We won't ask for your password again for a few hours.

10. After entering the password, you can manage access and add/remove team members to your project.

11. To add members, click on the add people option and search the id of your respective team member.


Add a collaborator to scm
Search by username, full name, or email
Select a collaborator above
Manage access
You haven't invited any collaborators yet
Add people

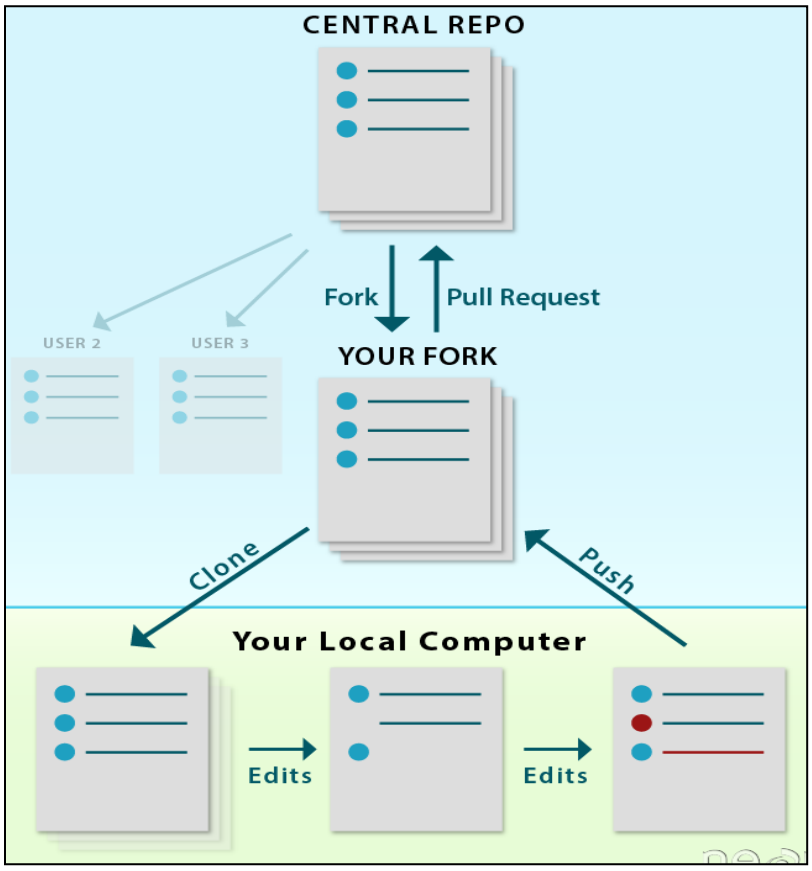
12. To remove any member, click on remove option available in the last column of member's respective row.


Add a collaborator to scm
PriyanshuSaini02
Add PriyanshuSaini02 to this repository
Manage access
You haven't invited any collaborators yet
Add people

Experiment No. 07

Aim: Fork and Commit

Theory: A fork is a copy of a repository that you manage. It allows us to freely experiment with the data. After creating a fork, we can make any desired change like adding collaborators, rename files, generate GitHub pages but all these changes won't be reflected in the original repository.



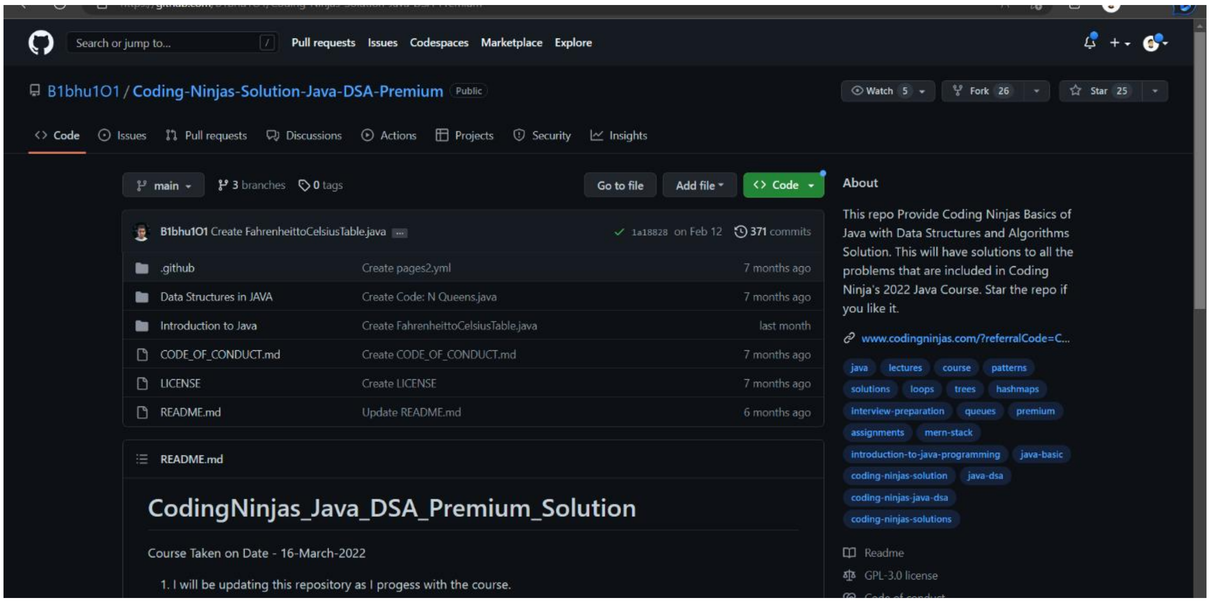
reference for picture: https://www.earthdatascience.org

To import the changes into the original repository, the user needs to send a pull request to the maintainer. If the maintainer closes the pull request only then the content can be added to the original repository.

Forking is a better method than directly cloning any repository, as in cloning only the default branch is cloned whereas forking creates a clone of the complete repo and also allows us to push the changes to the main repository by using open and close pull request.

Procedure:

1. To fork a repository first step you need to do is, sign in to your GitHub account and then you come to the repository you want to fork, so here for demo purpose am using **Lokesh-Garg-22/Temp-Repository** repository.



B1bhu1O1 Create FahrenheittoCelsiusTable.java ✓ 1a18828 on Feb 12 371 commits

.github	Create pages2.yml	7 months ago
Data Structures in JAVA	Create Code: N Queens.java	7 months ago
Introduction to Java	Create FahrenheittoCelsiusTable.java	last month
CODE_OF_CONDUCT.md	Create CODE_OF_CONDUCT.md	7 months ago
LICENSE	Create LICENSE	7 months ago
README.md	Update README.md	6 months ago

README.md

CodingNinjas_Java_DSA_Premium_Solution

Course Taken on Date - 16-March-2022

1. I will be updating this repository as I progess with the course.

About

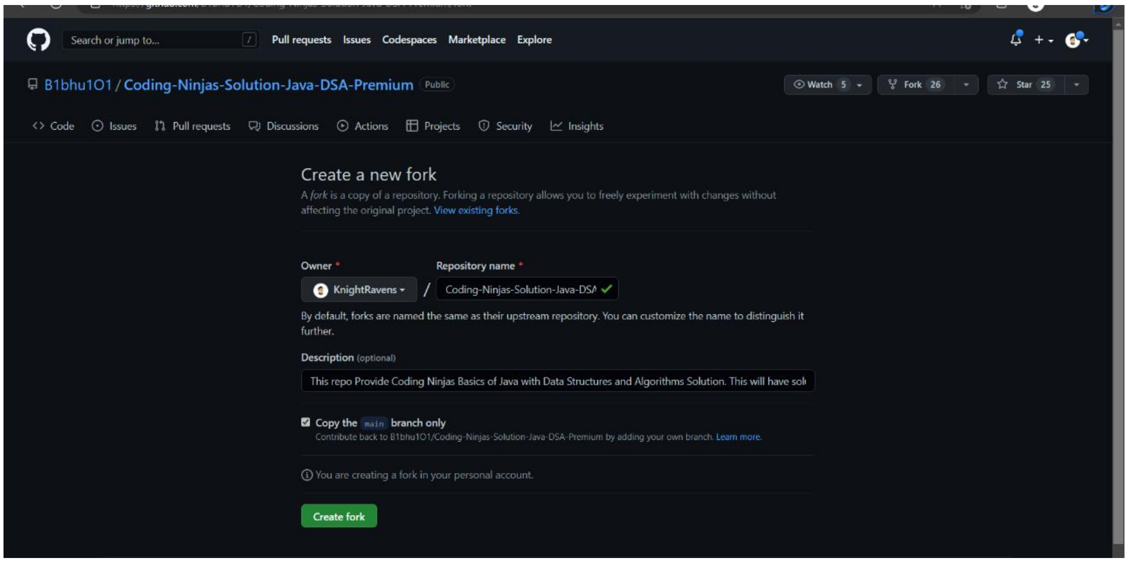
This repo Provide Coding Ninjas Basics of Java with Data Structures and Algorithms Solution. This will have solutions to all the problems that are included in Coding Ninja's 2022 Java Course. Star the repo if you like it.

www.codingninjas.com/?referralCode=C...

java lectures course patterns solutions loops trees hashmaps interview-preparation queues premium assignments mern-stack introduction-to-java-programming java-basic coding-ninjas-solution java-dsa coding-ninjas-java-dsa coding-ninjas-solutions

Readme
GPL-3.0 license

2. Click on the **Fork** button on right upside corner. Then it will ask to create a new fork, add description if you want and then click on create fork.



Create a new fork

A *fork* is a copy of a repository. Forking a repository allows you to freely experiment with changes without affecting the original project. View existing forks.

Owner * KnightRavens / Repository name * Coding-Ninjas-Solution-Java-DSA ✓

By default, forks are named the same as their upstream repository. You can customize the name to distinguish it further.

Description (optional)

This repo Provide Coding Ninjas Basics of Java with Data Structures and Algorithms Solution. This will have solu

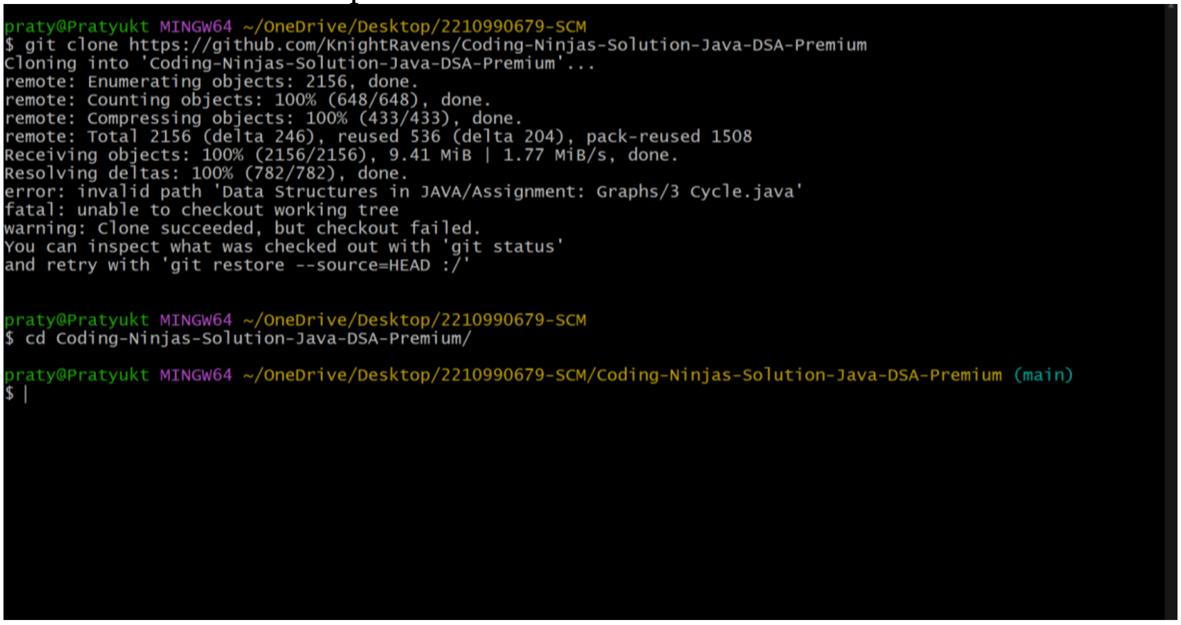
☑ Copy the `main` branch only
Contribute back to B1bhu1O1/Coding-Ninjas-Solution-Java-DSA-Premium by adding your own branch. Learn more.

ⓘ You are creating a fork in your personal account.

Create fork

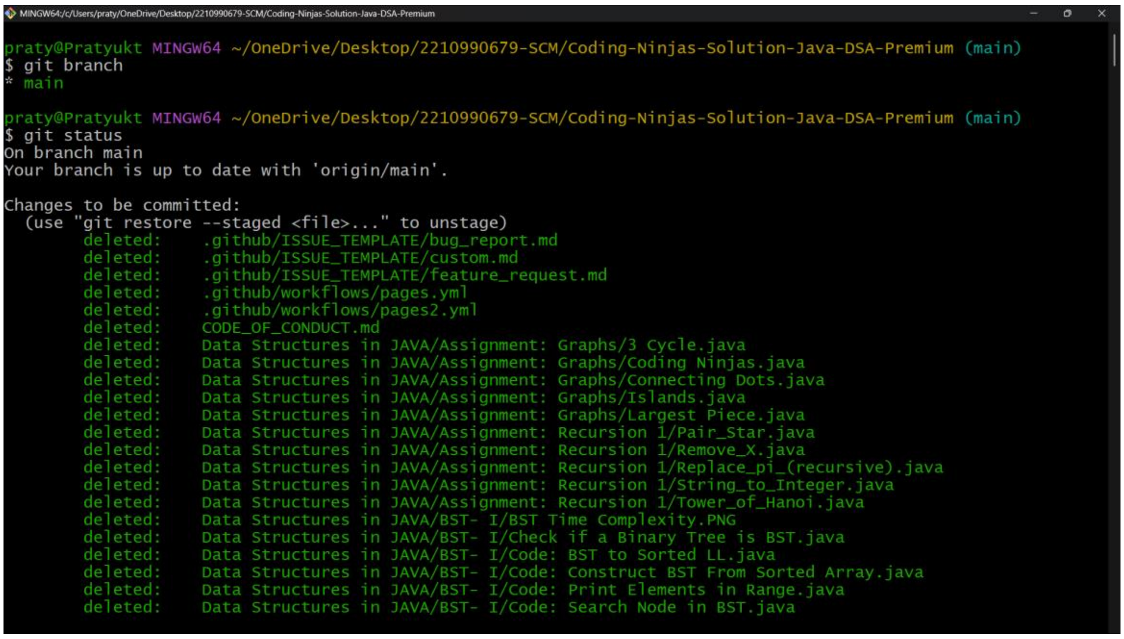
3. Now you will have a copy of the repo you have forked from other user. Now you can do any modification you want without making changes to main source code.

4. Now type git clone https://github.com/Lokesh544/Temp-Repository.git on git.
   Git clone <url> --> This command is used to fetch the remote repo or to clone the repo.



```
praty@Pratyukt MINGW64 ~/OneDrive/Desktop/2210990679-SCM
$ git clone https://github.com/KnightRavens/Coding-Ninjas-Solution-Java-DSA-Premium
Cloning into 'Coding-Ninjas-Solution-Java-DSA-Premium'...
remote: Enumerating objects: 2156, done.
remote: Counting objects: 100% (648/648), done.
remote: Compressing objects: 100% (433/433), done.
remote: Total 2156 (delta 246), reused 536 (delta 204), pack-reused 1508
Receiving objects: 100% (2156/2156), 9.41 MiB | 1.77 MiB/s, done.
Resolving deltas: 100% (782/782), done.
error: invalid path 'Data Structures in JAVA/Assignment: Graphs/3 Cycle.java'
fatal: unable to checkout working tree
warning: Clone succeeded, but checkout failed.
You can inspect what was checked out with 'git status'
and retry with 'git restore --source=HEAD :/'

praty@Pratyukt MINGW64 ~/OneDrive/Desktop/2210990679-SCM
$ cd Coding-Ninjas-Solution-Java-DSA-Premium/

praty@Pratyukt MINGW64 ~/OneDrive/Desktop/2210990679-SCM/Coding-Ninjas-Solution-Java-DSA-Premium (main)
$ |
```

5. Now Open the file make changes in it and commit it and push it to remote.



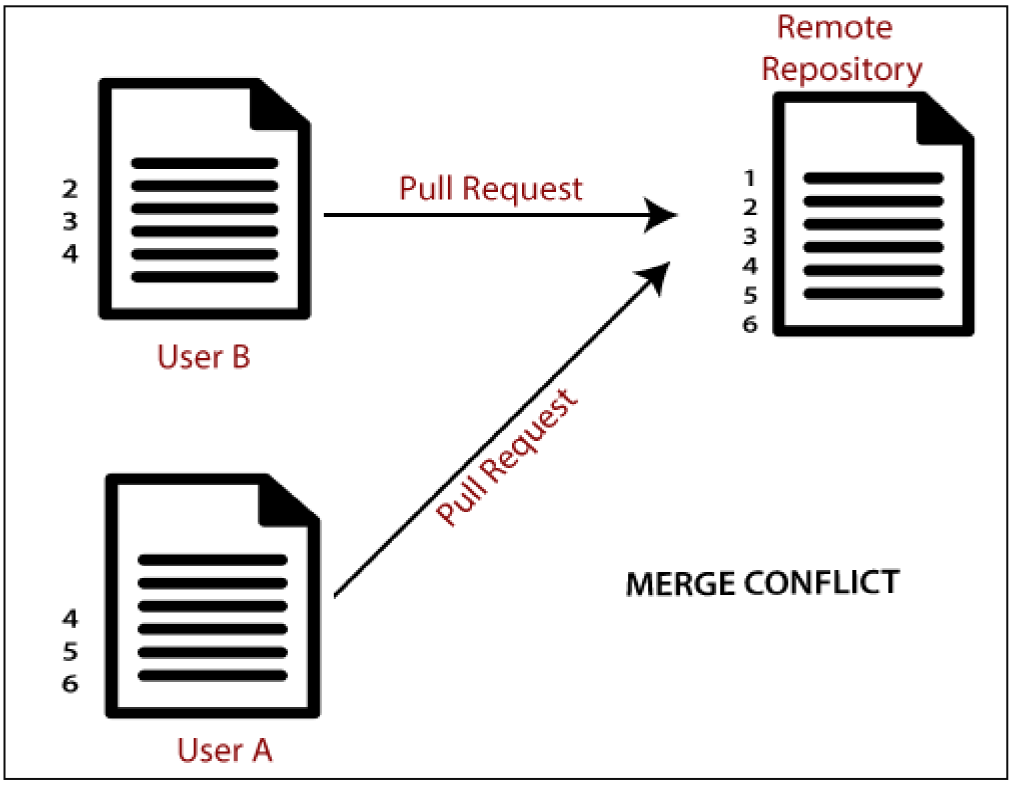
```
MINGW64:/c/Users/praty/OneDrive/Desktop/2210990679-SCM/Coding-Ninjas-Solution-Java-DSA-Premium

praty@Pratyukt MINGW64 ~/OneDrive/Desktop/2210990679-SCM/Coding-Ninjas-Solution-Java-DSA-Premium (main)
$ git branch
* main

praty@Pratyukt MINGW64 ~/OneDrive/Desktop/2210990679-SCM/Coding-Ninjas-Solution-Java-DSA-Premium (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        deleted:    .github/ISSUE_TEMPLATE/bug_report.md
        deleted:    .github/ISSUE_TEMPLATE/custom.md
        deleted:    .github/ISSUE_TEMPLATE/feature_request.md
        deleted:    .github/workflows/pages.yml
        deleted:    .github/workflows/pages2.yml
        deleted:    CODE_OF_CONDUCT.md
        deleted:    Data Structures in JAVA/Assignment: Graphs/3 Cycle.java
        deleted:    Data Structures in JAVA/Assignment: Graphs/Coding Ninjas.java
        deleted:    Data Structures in JAVA/Assignment: Graphs/Connecting Dots.java
        deleted:    Data Structures in JAVA/Assignment: Graphs/Islands.java
        deleted:    Data Structures in JAVA/Assignment: Graphs/Largest Piece.java
        deleted:    Data Structures in JAVA/Assignment: Recursion 1/Pair_Star.java
        deleted:    Data Structures in JAVA/Assignment: Recursion 1/Remove_X.java
        deleted:    Data Structures in JAVA/Assignment: Recursion 1/Replace_pi_(recursive).java
        deleted:    Data Structures in JAVA/Assignment: Recursion 1/String_to_Integer.java
        deleted:    Data Structures in JAVA/Assignment: Recursion 1/Tower_of_Hanoi.java
        deleted:    Data Structures in JAVA/BST- I/BST Time Complexity.PNG
        deleted:    Data Structures in JAVA/BST- I/Check if a Binary Tree is BST.java
        deleted:    Data Structures in JAVA/BST- I/Code: BST to Sorted LL.java
        deleted:    Data Structures in JAVA/BST- I/Code: Construct BST From Sorted Array.java
        deleted:    Data Structures in JAVA/BST- I/Code: Print Elements in Range.java
        deleted:    Data Structures in JAVA/BST- I/Code: Search Node in BST.java
```

Experiment No. 08

Aim: Merge and Resolve conflicts created due to own activity and collaborators activity.

Theory:

Version control systems are all about managing contributions between multiple distributed authors (usually developers). Sometimes multiple developers may try to edit the same content. If Developer A tries to edit code that Developer B is editing a conflict may occur.



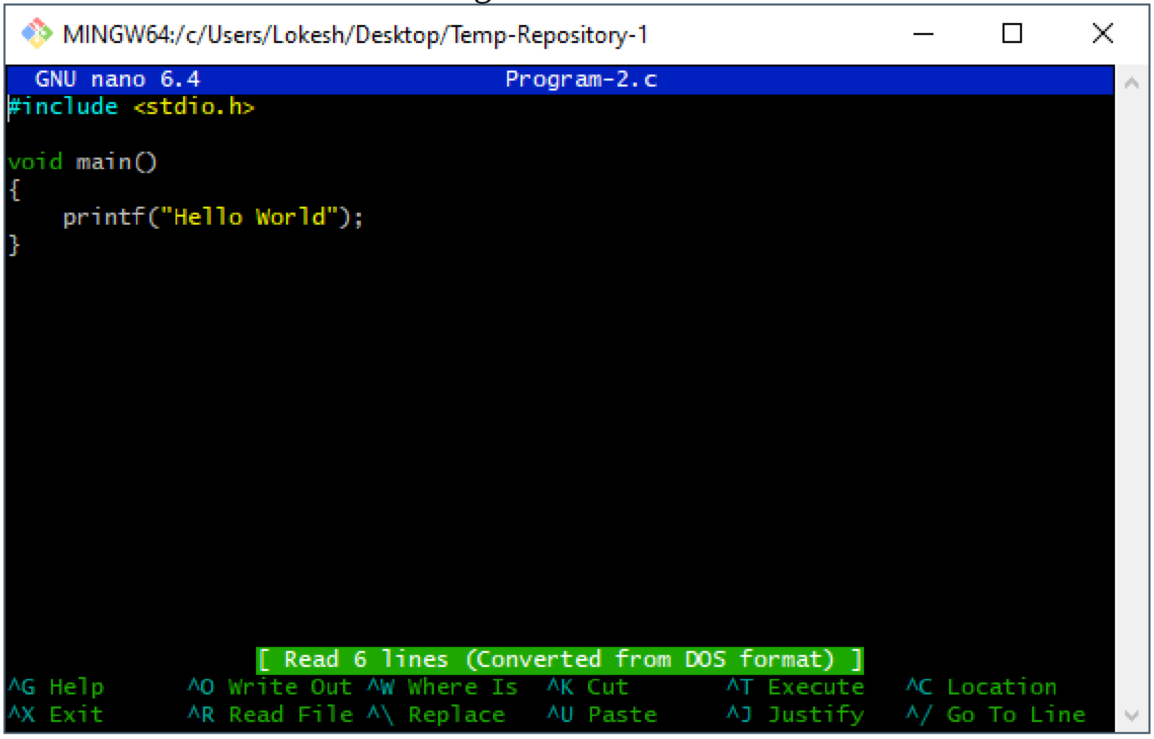
reference for picture: https://www.javatpoint.com/git-merge-and-merge-conflict

If you have a merge conflict on the command line, you cannot push your local changes to GitHub until you resolve the merge conflict locally on your computer.

To alleviate the occurrence of conflicts developers will work in separate isolated branches. If a merge conflict still arises between the compare branch and base branch in your pull request, you can view a list of the files with conflicting changes above the Merge pull request button. The Merge pull request button is deactivated until you've resolved all conflicts between the compare branch and base branch.

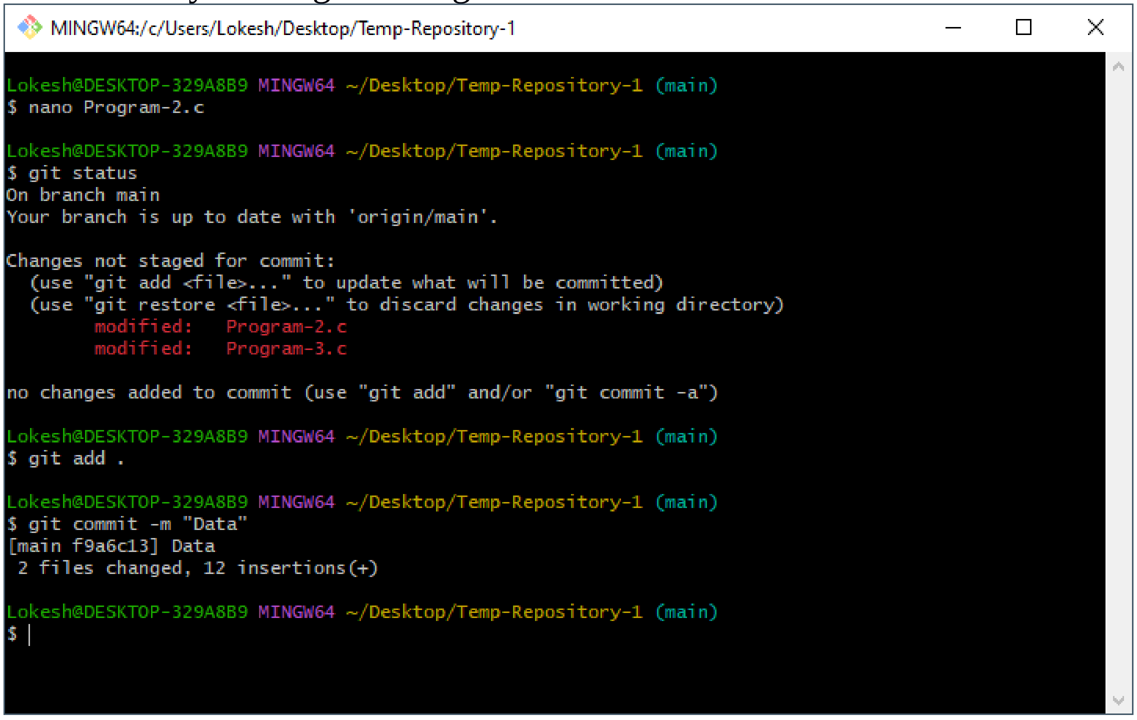
Procedure:

1) Do changes in master branch and commit those change. And checkout to different branch and again do changes and commit it. Now checkout to master branch and merge that branch in master.



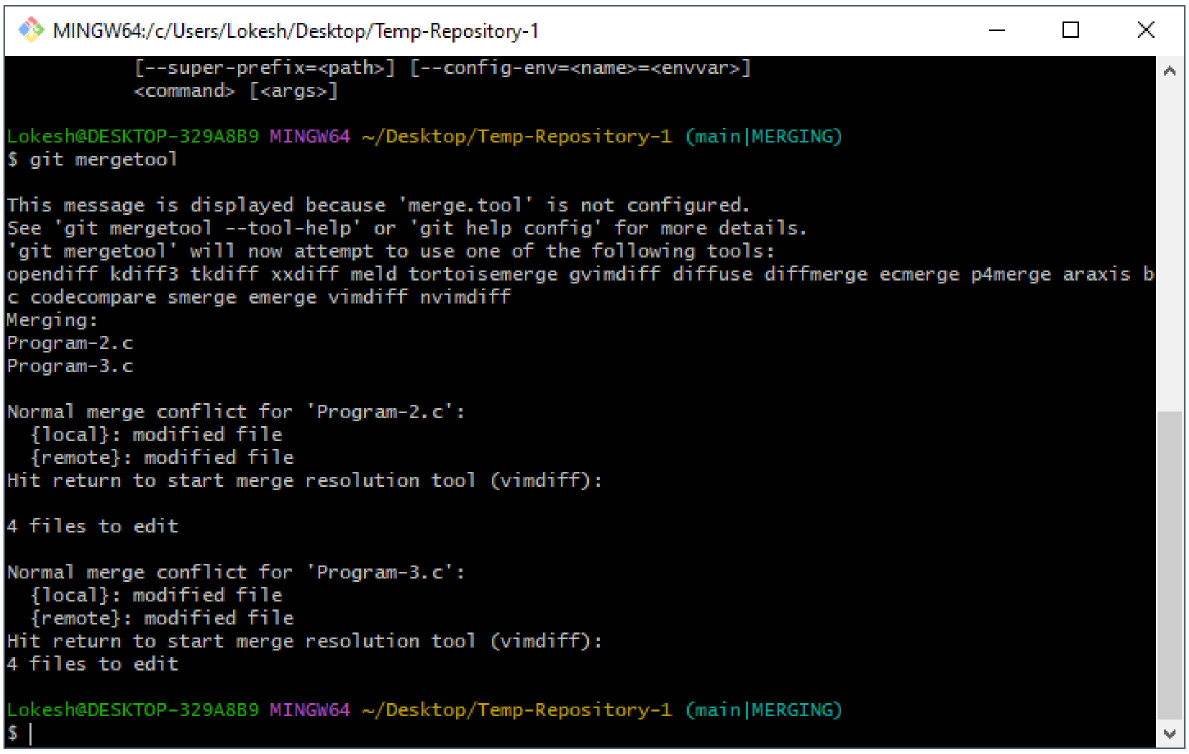
```
MINGW64:/c/Users/Lokesh/Desktop/Temp-Repository-1
  GNU nano 6.4                      Program-2.c
#include <stdio.h>

void main()
{
    printf("Hello World");
}

                   [ Read 6 lines (Converted from DOS format) ]
^G Help        ^O Write Out ^W Where Is  ^K Cut         ^T Execute    ^C Location
^X Exit        ^R Read File ^\ Replace   ^U Paste       ^J Justify    ^/ Go To Line
```

2. Now try to merge it will give Conflicts Error.



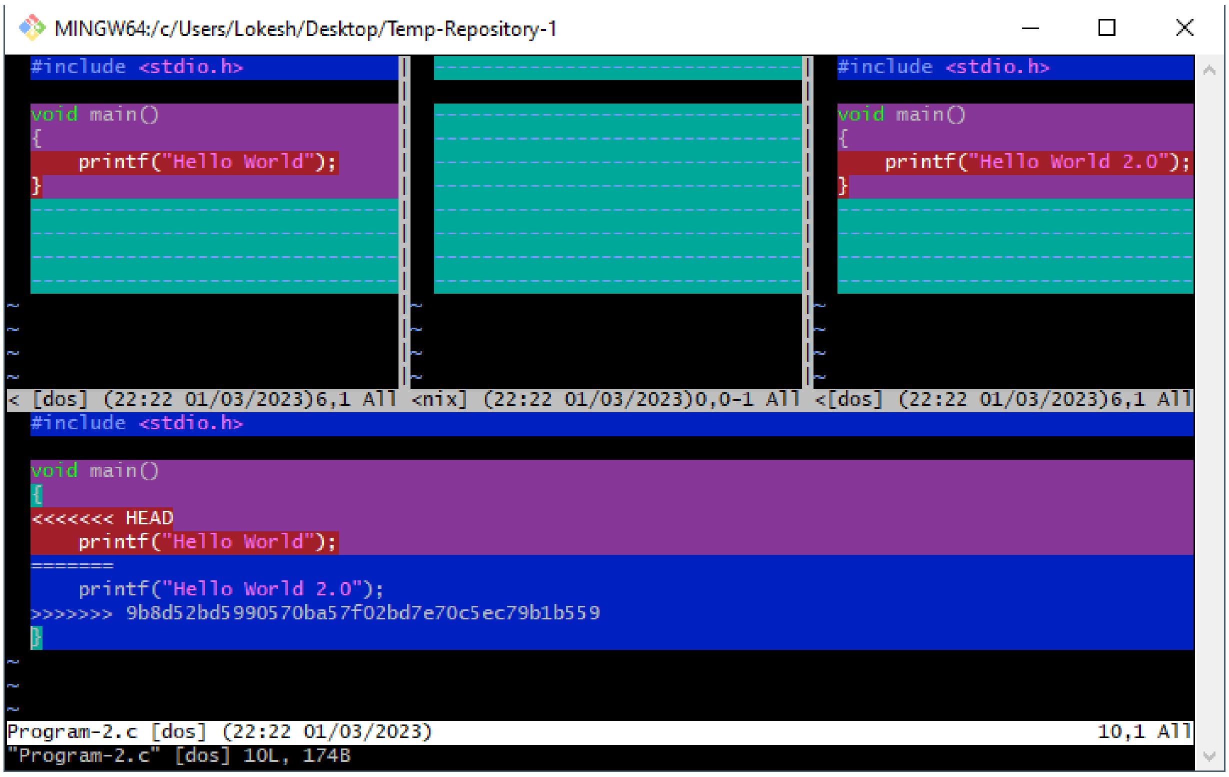
```
MINGW64:/c/Users/Lokesh/Desktop/Temp-Repository-1

Lokesh@DESKTOP-329A8B9 MINGW64 ~/Desktop/Temp-Repository-1 (main)
$ nano Program-2.c

Lokesh@DESKTOP-329A8B9 MINGW64 ~/Desktop/Temp-Repository-1 (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   Program-2.c
        modified:   Program-3.c

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

Lokesh@DESKTOP-329A8B9 MINGW64 ~/Desktop/Temp-Repository-1 (main)
$ git add .

Lokesh@DESKTOP-329A8B9 MINGW64 ~/Desktop/Temp-Repository-1 (main)
$ git commit -m "Data"
[main f9a6c13] Data
 2 files changed, 12 insertions(+)

Lokesh@DESKTOP-329A8B9 MINGW64 ~/Desktop/Temp-Repository-1 (main)
$ |
```

3. Use Command "git mergetool" to solve the conflict.
   git -mergetool – Run merge conflict resolution tools to resolve merge conflicts.



```
MINGW64:/c/Users/Lokesh/Desktop/Temp-Repository-1
           [--super-prefix=<path>] [--config-env=<name>=<envvar>]
           <command> [<args>]

Lokesh@DESKTOP-329A8B9 MINGW64 ~/Desktop/Temp-Repository-1 (main|MERGING)
$ git mergetool

This message is displayed because 'merge.tool' is not configured.
See 'git mergetool --tool-help' or 'git help config' for more details.
'git mergetool' will now attempt to use one of the following tools:
opendiff kdiff3 tkdiff xxdiff meld tortoisemerge gvimdiff diffuse diffmerge ecmerge p4merge araxis b
c codecompare smerge emerge vimdiff nvimdiff
Merging:
Program-2.c
Program-3.c

Normal merge conflict for 'Program-2.c':
  {local}: modified file
  {remote}: modified file
Hit return to start merge resolution tool (vimdiff):

4 files to edit

Normal merge conflict for 'Program-3.c':
  {local}: modified file
  {remote}: modified file
Hit return to start merge resolution tool (vimdiff):
4 files to edit

Lokesh@DESKTOP-329A8B9 MINGW64 ~/Desktop/Temp-Repository-1 (main|MERGING)
$ |
```

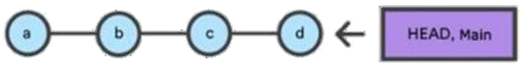
4. Press "I" to insert, after insertion. Press ":wq". The merge conflict is solved and aastha branch is merged to master branch.



```
MINGW64:/c/Users/Lokesh/Desktop/Temp-Repository-1
#include <stdio.h>              |  ---------------------------  | #include <stdio.h>
                                |                               |
void main()                     |  ---------------------------  | void main()
{                               |  ---------------------------  | {
    printf("Hello World");      |  ---------------------------  |     printf("Hello World 2.0");
}                               |  ---------------------------  | }
------------------------------  |  ---------------------------  | ------------------------------
------------------------------  |  ---------------------------  | ------------------------------
------------------------------  |  ---------------------------  | ------------------------------
------------------------------  |  ---------------------------  | ------------------------------
~                               |~                              |~
~                               |~                              |~
~                               |~                              |~
~                               |~                              |~
< [dos] (22:22 01/03/2023)6,1 All <nix] (22:22 01/03/2023)0,0-1 All <[dos] (22:22 01/03/2023)6,1 All
#include <stdio.h>

void main()
{
<<<<<<< HEAD
    printf("Hello World");
=======
    printf("Hello World 2.0");
>>>>>>> 9b8d52bd5990570ba57f02bd7e70c5ec79b1b559
}
~
~
~
Program-2.c [dos] (22:22 01/03/2023)                                                          10,1 All
"Program-2.c" [dos] 10L, 174B
```

Experiment No. 09

Aim: Reset and Revert



Theory:

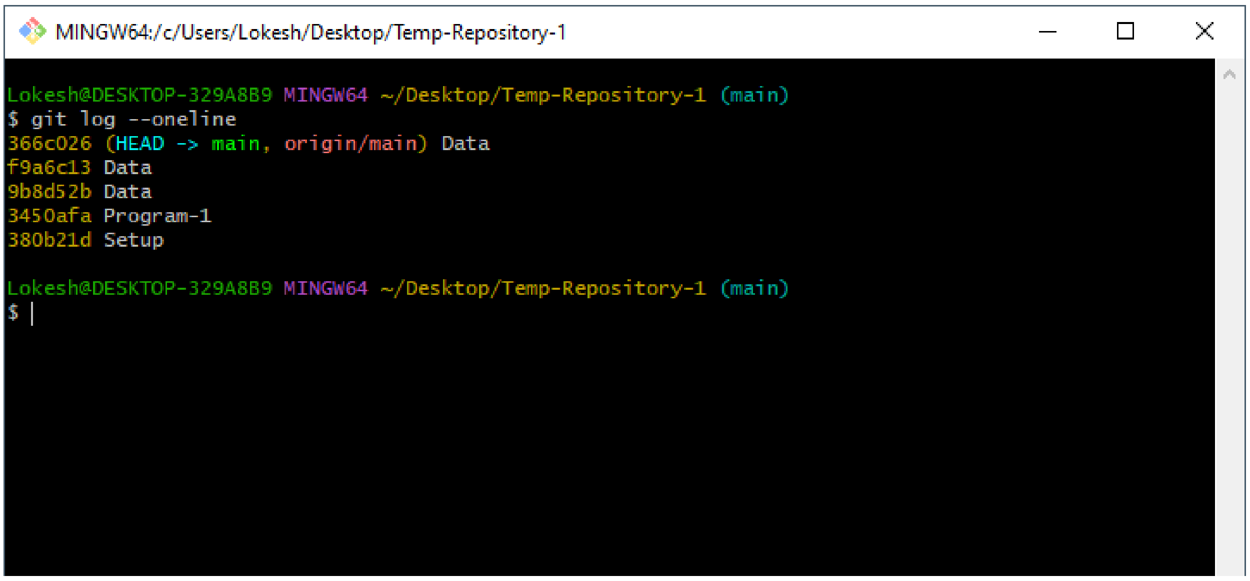
Git-revert – Revert some existing commits.
A reset is an operation that takes a specified commit and resets the "three trees" to match the state of the repository at that specified commit. A reset can be invoked in three different modes which correspond to the three trees. In reset, rest of the commits wash out after the mentioned commit. This is a limitation of reset command that we cannot have any random access.
A revert is an operation that takes a specified commit and creates a new commit which inverses the specified commit. git revert can only be run at a commit level scope and has no file level functionality.
These two features justify the Version- controlled feature of the git as we can rollback to any version at any time.

PROCEDURE:

Firstly, prepare a log of multiple commits to make the reset and revert command function.

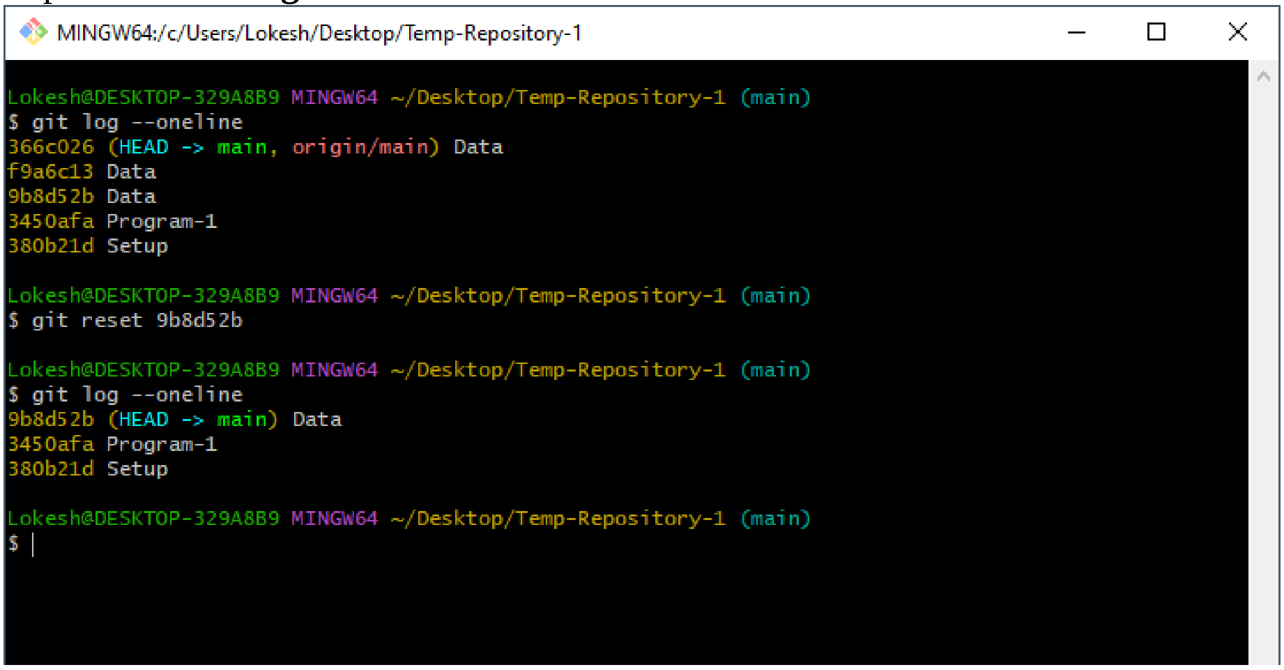
Reset: reset is the command we use when we want to move the repository back to a previous commit, discarding any changes made after that commit.

1) Create few files, stage them and commit.
2) Check the git log



```
Lokesh@DESKTOP-329A8B9 MINGW64 ~/Desktop/Temp-Repository-1 (main)
$ git log --oneline
366c026 (HEAD -> main, origin/main) Data
f9a6c13 Data
9b8d52b Data
3450afa Program-1
380b21d Setup

Lokesh@DESKTOP-329A8B9 MINGW64 ~/Desktop/Temp-Repository-1 (main)
$ |
```

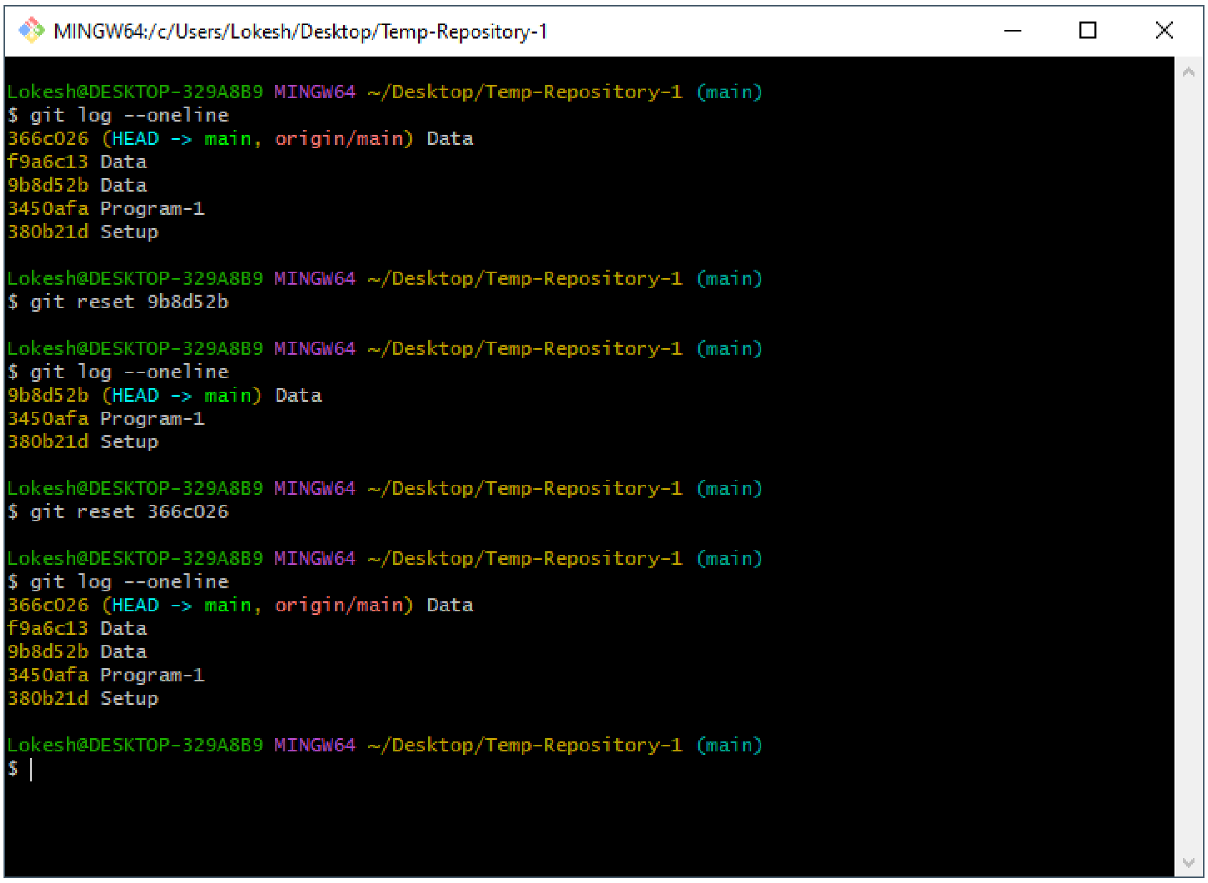
3) Pick any commit where you want the repository to rollback. Copy its checksum and paste it in the $ **git reset checksum** command.



```
MINGW64:/c/Users/Lokesh/Desktop/Temp-Repository-1

Lokesh@DESKTOP-329A8B9 MINGW64 ~/Desktop/Temp-Repository-1 (main)
$ git log --oneline
366c026 (HEAD -> main, origin/main) Data
f9a6c13 Data
9b8d52b Data
3450afa Program-1
380b21d Setup

Lokesh@DESKTOP-329A8B9 MINGW64 ~/Desktop/Temp-Repository-1 (main)
$ git reset 9b8d52b

Lokesh@DESKTOP-329A8B9 MINGW64 ~/Desktop/Temp-Repository-1 (main)
$ git log --oneline
9b8d52b (HEAD -> main) Data
3450afa Program-1
380b21d Setup

Lokesh@DESKTOP-329A8B9 MINGW64 ~/Desktop/Temp-Repository-1 (main)
$ |
```

The head is now pointing the commit whose checksum we have provided that means the commits that followed vanished.

4) In you want undo this change, you copy the checksum of the commit you want back and run the same command again.

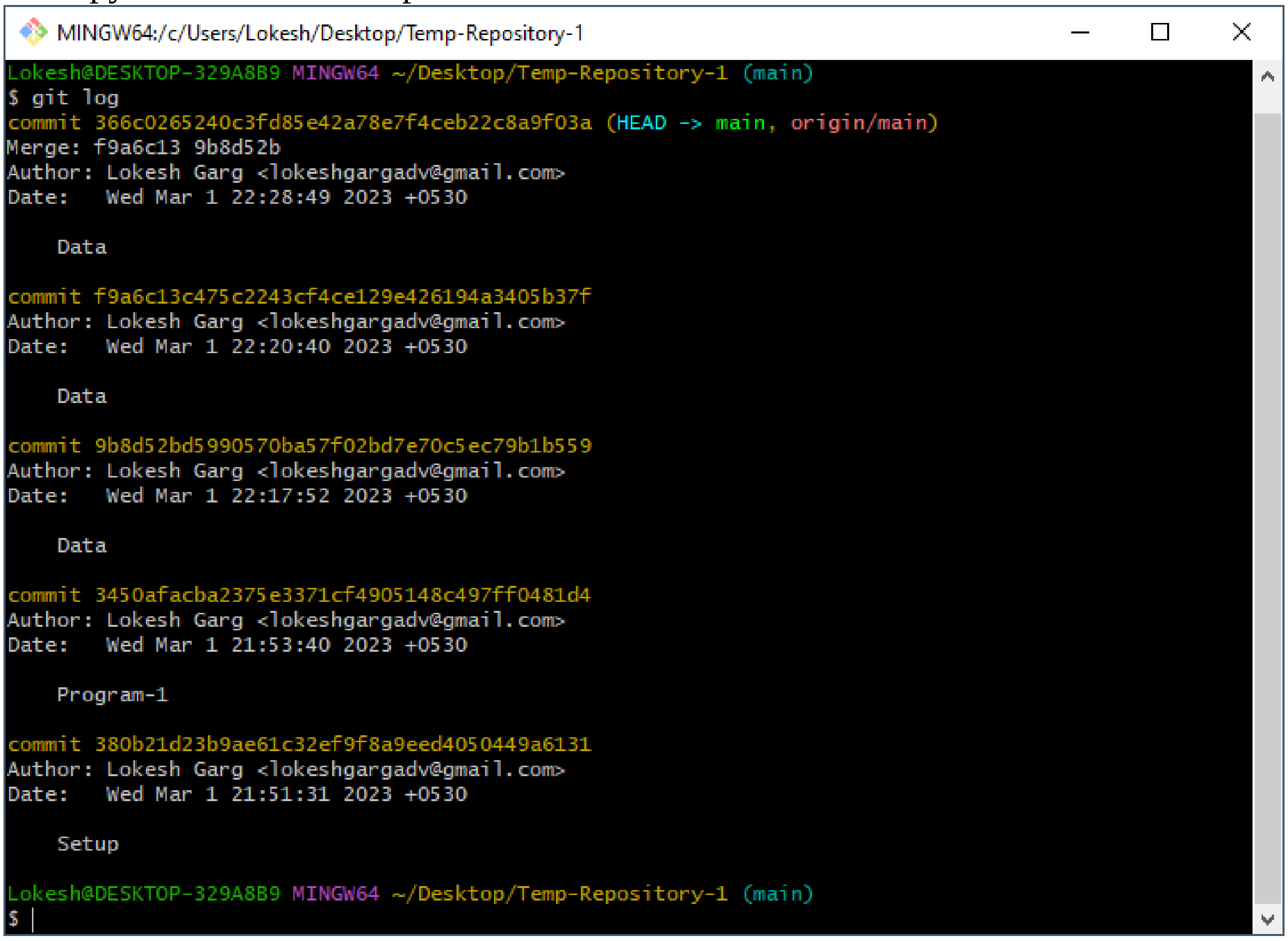


```
MINGW64:/c/Users/Lokesh/Desktop/Temp-Repository-1

Lokesh@DESKTOP-329A8B9 MINGW64 ~/Desktop/Temp-Repository-1 (main)
$ git log --oneline
366c026 (HEAD -> main, origin/main) Data
f9a6c13 Data
9b8d52b Data
3450afa Program-1
380b21d Setup

Lokesh@DESKTOP-329A8B9 MINGW64 ~/Desktop/Temp-Repository-1 (main)
$ git reset 9b8d52b

Lokesh@DESKTOP-329A8B9 MINGW64 ~/Desktop/Temp-Repository-1 (main)
$ git log --oneline
9b8d52b (HEAD -> main) Data
3450afa Program-1
380b21d Setup

Lokesh@DESKTOP-329A8B9 MINGW64 ~/Desktop/Temp-Repository-1 (main)
$ git reset 366c026

Lokesh@DESKTOP-329A8B9 MINGW64 ~/Desktop/Temp-Repository-1 (main)
$ git log --oneline
366c026 (HEAD -> main, origin/main) Data
f9a6c13 Data
9b8d52b Data
3450afa Program-1
380b21d Setup

Lokesh@DESKTOP-329A8B9 MINGW64 ~/Desktop/Temp-Repository-1 (main)
$ |
```

Revert:

Follow these steps to revert any change:

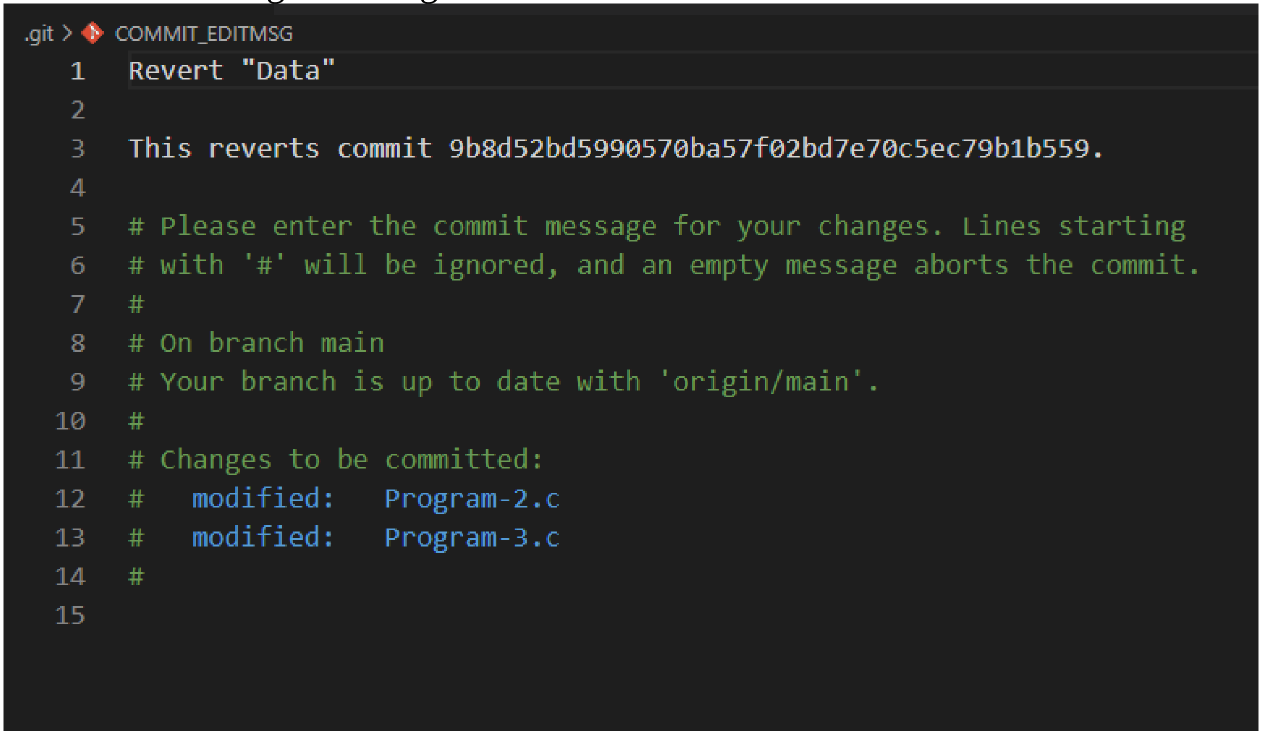
1) Pick the change where you want the project to revert back.
Copy its checksum and paste it in the revert command.



MINGW64:/c/Users/Lokesh/Desktop/Temp-Repository-1

```
Lokesh@DESKTOP-329A8B9 MINGW64 ~/Desktop/Temp-Repository-1 (main)
$ git log
commit 366c0265240c3fd85e42a78e7f4ceb22c8a9f03a (HEAD -> main, origin/main)
Merge: f9a6c13 9b8d52b
Author: Lokesh Garg <lokeshgargadv@gmail.com>
Date:   Wed Mar 1 22:28:49 2023 +0530

    Data

commit f9a6c13c475c2243cf4ce129e426194a3405b37f
Author: Lokesh Garg <lokeshgargadv@gmail.com>
Date:   Wed Mar 1 22:20:40 2023 +0530

    Data

commit 9b8d52bd5990570ba57f02bd7e70c5ec79b1b559
Author: Lokesh Garg <lokeshgargadv@gmail.com>
Date:   Wed Mar 1 22:17:52 2023 +0530

    Data

commit 3450afacba2375e3371cf4905148c497ff0481d4
Author: Lokesh Garg <lokeshgargadv@gmail.com>
Date:   Wed Mar 1 21:53:40 2023 +0530

    Program-1

commit 380b21d23b9ae61c32ef9f8a9eed4050449a6131
Author: Lokesh Garg <lokeshgargadv@gmail.com>
Date:   Wed Mar 1 21:51:31 2023 +0530

    Setup

Lokesh@DESKTOP-329A8B9 MINGW64 ~/Desktop/Temp-Repository-1 (main)
$ |
```

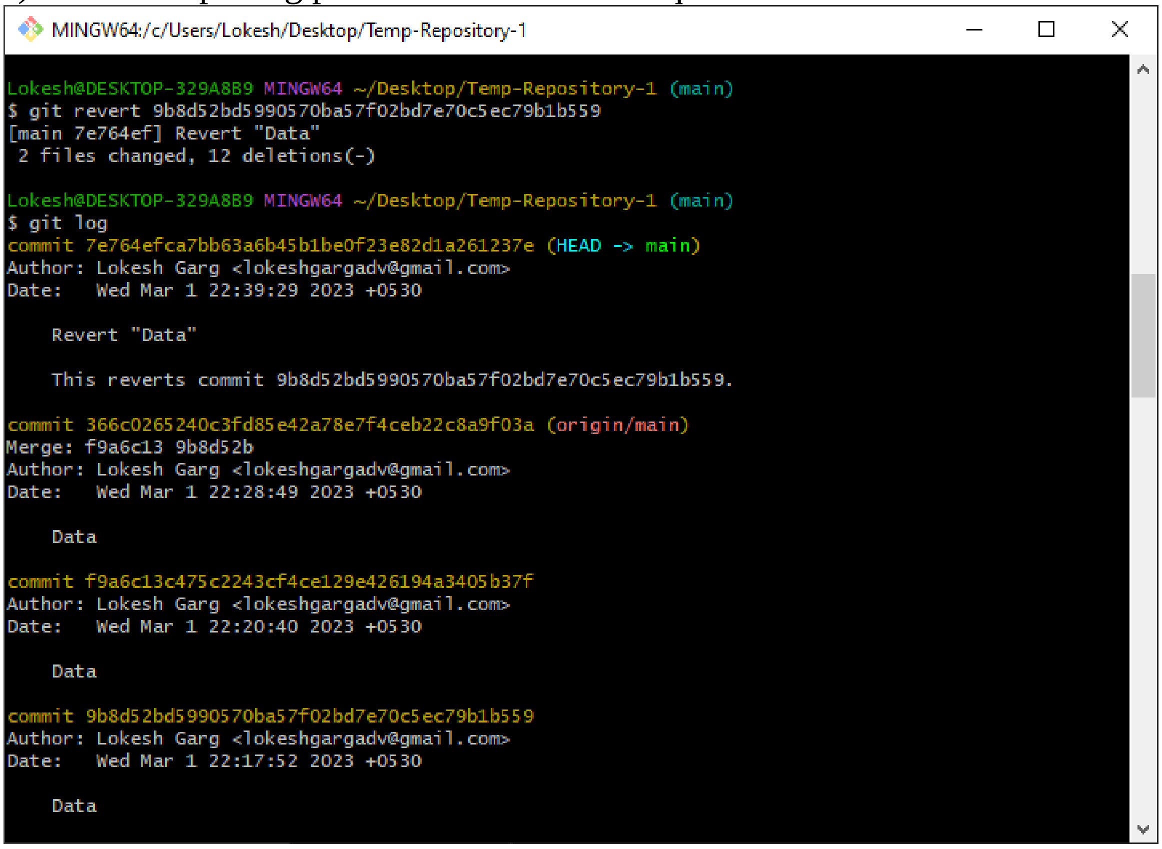
2) A window will appear. Press 'I" and write the statement you want to be displayed for reverting the change.



.git > COMMIT_EDITMSG

```
Revert "Data"

This reverts commit 9b8d52bd5990570ba57f02bd7e70c5ec79b1b559.

# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# On branch main
# Your branch is up to date with 'origin/main'.
#
# Changes to be committed:
#	modified:   Program-2.c
#	modified:   Program-3.c
#

```

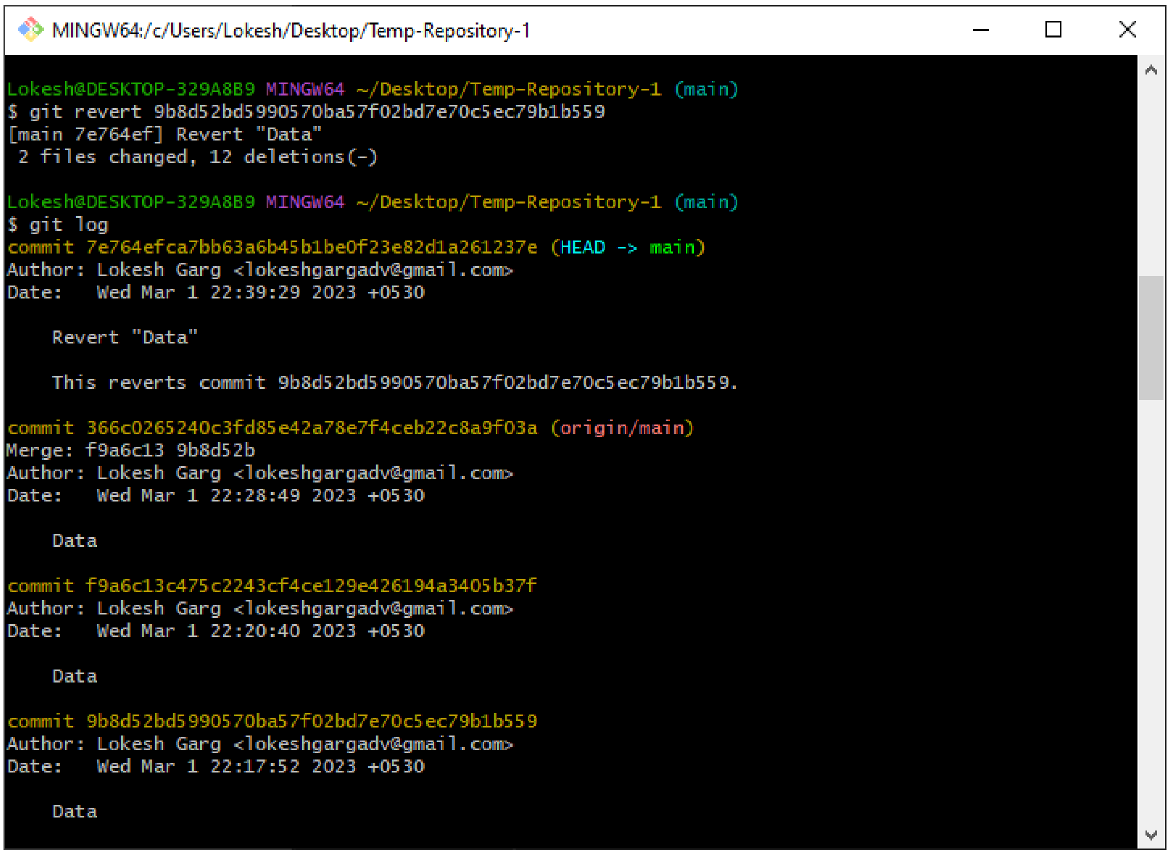
3) After completing press ‘esc’ and write: wq in the terminal.



MINGW64:/c/Users/Lokesh/Desktop/Temp-Repository-1

```
Lokesh@DESKTOP-329A8B9 MINGW64 ~/Desktop/Temp-Repository-1 (main)
$ git revert 9b8d52bd5990570ba57f02bd7e70c5ec79b1b559
[main 7e764ef] Revert "Data"
 2 files changed, 12 deletions(-)

Lokesh@DESKTOP-329A8B9 MINGW64 ~/Desktop/Temp-Repository-1 (main)
$ git log
commit 7e764efca7bb63a6b45b1be0f23e82d1a261237e (HEAD -> main)
Author: Lokesh Garg <lokeshgargadv@gmail.com>
Date:   Wed Mar 1 22:39:29 2023 +0530

    Revert "Data"

    This reverts commit 9b8d52bd5990570ba57f02bd7e70c5ec79b1b559.

commit 366c0265240c3fd85e42a78e7f4ceb22c8a9f03a (origin/main)
Merge: f9a6c13 9b8d52b
Author: Lokesh Garg <lokeshgargadv@gmail.com>
Date:   Wed Mar 1 22:28:49 2023 +0530

    Data

commit f9a6c13c475c2243cf4ce129e426194a3405b37f
Author: Lokesh Garg <lokeshgargadv@gmail.com>
Date:   Wed Mar 1 22:20:40 2023 +0530

    Data

commit 9b8d52bd5990570ba57f02bd7e70c5ec79b1b559
Author: Lokesh Garg <lokeshgargadv@gmail.com>
Date:   Wed Mar 1 22:17:52 2023 +0530

    Data
```

4) Check the git log and you will find another commit is added without affecting the rest commits.



MINGW64:/c/Users/Lokesh/Desktop/Temp-Repository-1

```
Lokesh@DESKTOP-329A8B9 MINGW64 ~/Desktop/Temp-Repository-1 (main)
$ git revert 9b8d52bd5990570ba57f02bd7e70c5ec79b1b559
[main 7e764ef] Revert "Data"
 2 files changed, 12 deletions(-)

Lokesh@DESKTOP-329A8B9 MINGW64 ~/Desktop/Temp-Repository-1 (main)
$ git log
commit 7e764efca7bb63a6b45b1be0f23e82d1a261237e (HEAD -> main)
Author: Lokesh Garg <lokeshgargadv@gmail.com>
Date:   Wed Mar 1 22:39:29 2023 +0530

    Revert "Data"

    This reverts commit 9b8d52bd5990570ba57f02bd7e70c5ec79b1b559.

commit 366c0265240c3fd85e42a78e7f4ceb22c8a9f03a (origin/main)
Merge: f9a6c13 9b8d52b
Author: Lokesh Garg <lokeshgargadv@gmail.com>
Date:   Wed Mar 1 22:28:49 2023 +0530

    Data

commit f9a6c13c475c2243cf4ce129e426194a3405b37f
Author: Lokesh Garg <lokeshgargadv@gmail.com>
Date:   Wed Mar 1 22:20:40 2023 +0530

    Data

commit 9b8d52bd5990570ba57f02bd7e70c5ec79b1b559
Author: Lokesh Garg <lokeshgargadv@gmail.com>
Date:   Wed Mar 1 22:17:52 2023 +0530

    Data
```

5) The change associated to the reverted commit has disappeared.