Ex3 - Collaboration Made Easy

Please read the instructions carefully before completing this exercise.

Up until now we've been working on a local Git repository. While working locally can be fun for tracking your own personal software project, Git is especially useful for collaboration of multiple developers on the same project. In this exercise we are going to open a new account on github.com, a hosting server for source code (storing repositories). Then we will create our first new remote repository on GitHub and push our project to it.

Please submit your answer as a PDF file. For each question, include in your answer the question number, the command(s) you used to solve the question, and the output of the command(s) (text or screenshot).

Part 1: Getting Started

Questions:

1. Using `git --version`, report the version number of Git that you are using to solve this exercise.

```
gadi@DESKTOP-IEPN1TQ:/mnt/d/Academy/Studying/Bootcamp/ITC Feb 23/Assignments/Git_Fundamentals/Solutions/Ex1 - Gitting Started$ git --
version
git version 2.25.1
```

2. Sign up on GitHub, and write your GitHub username in your answer. Make sure to pick a meaningful username, as it will follow you for the rest of your life :)

   Gadi G. Ezer

3. Set **Personal Access Token (PAT) (recommended)** or **SSH key authentication** for your GitHub account. See the appendix at the end of this document for instructions. Do not include commands or output in your answer to this question; just write that you did this step.

   DONE

4. Create a new public repository on GitHub. In order to check it is public, try to access the repository URL from an incognito browser window. Include the repository's URL in your answer.
   https://github.com/Gadi-G-Ezer/Git_HW.git

5. Using a local Git command, connect your local repository from the previous exercises to your remote repository on GitHub. Note: If you used a password, SSH key or access token, **do not include it in your answer** as it should be kept secret.

```
gadi@DESKTOP-IEPN1TQ:/mnt/d/Academy/Studying/Bootcamp/ITC Feb 23/Assignments/Git/Solutions/Ex1 - Gitting Started$ git remote add orig
in https://github.com/Gadi-G-Ezer/Git_HW.git
gadi@DESKTOP-IEPN1TQ:/mnt/d/Academy/Studying/Bootcamp/ITC Feb 23/Assignments/Git/Solutions/Ex1 - Gitting Started$ git remote -v
origin  https://github.com/Gadi-G-Ezer/Git_HW.git (fetch)
origin  https://github.com/Gadi-G-Ezer/Git_HW.git (push)
```

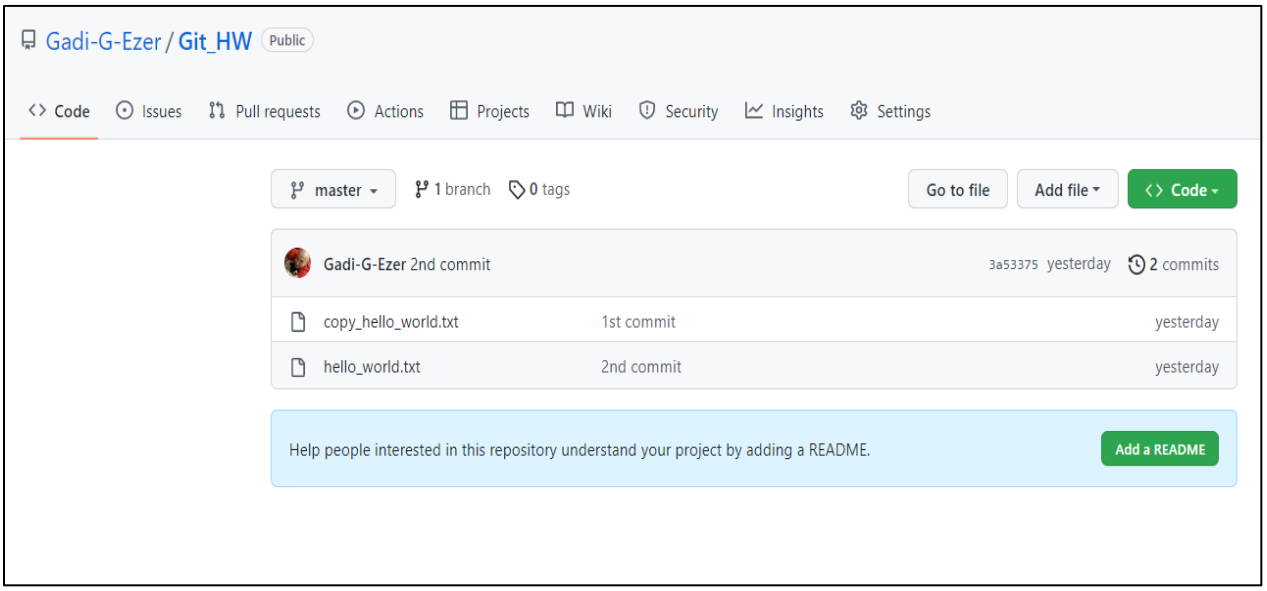
6. Execute another Git command to verify that your remote repository is set up properly.

```
gadi@DESKTOP-IEPN1TQ:/mnt/d/Academy/Studying/Bootcamp/ITC Feb 23/Assignments/Git/Solutions/Ex1 - Gitting Started$ git remote -v
origin  https://github.com/Gadi-G-Ezer/Git_HW.git (fetch)
origin  https://github.com/Gadi-G-Ezer/Git_HW.git (push)
```

7. Now you're finally ready to make your project available for other collaborators! Execute a Git command or commands on your local repository to push your code to the remote one. Document the command(s) you used in your answers file. You might need to insert your Github password to make the push, but **do not include it in your answer**. Note that once you push your code, all commits are being pushed at once (unless specified otherwise).

```
gadi@DESKTOP-IEPN1TQ:/mnt/d/Academy/Studying/Bootcamp/ITC Feb 23/Assignments/Git/Solutions/Ex1 - Gitting Started$ git push origin mas
ter
Username for 'https://github.com': Gadi-G-Ezer
Password for 'https://Gadi-G-Ezer@github.com':
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 8 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (6/6), 674 bytes | 51.00 KiB/s, done.
Total 6 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), done.
To https://github.com/Gadi-G-Ezer/Git_HW.git
 * [new branch]      master -> master
```

8. Go back to GitHub and use the UI to make sure your code is now available for collaborators.



Part 2: Collaborating with a Colleague

For the following sections, you are **required to work in pairs**. If you have trouble finding a partner, please let a mentor know.

Questions:

9. Write the name and GitHub username of your collaborator here.
   GGE, Gadiasia7

10. From now on, we will refer to your original repository as repo A, and to your colleague's repository as repo B. Provide your collaborator with read and write access to repo A on GitHub, and have them do the same for you on repo B. (No commands needed in your answer to this question, just confirm that you did this.)
    Done

11. Once there is a remote repository that you're interested to be in collaboration with, you shouldn't create a new local repository, but rather clone the existing one using the

remote repository's URL. Clone repo B, and document the command and its output. Note: Make sure that you clone it to a different folder than the one storing your local copy of repo A.

```
gadi@DESKTOP-IEPN1TQ:/mnt/d/Academy/Studying/Bootcamp/ITC Feb 23/Assignments/Git/Solutions$ git clone https://github.com/Gadiasia/HOM
E_WORK_ITC.git clone_gadiasia7
Cloning into 'clone_gadiasia7'...
remote: Enumerating objects: 12, done.
remote: Counting objects: 100% (12/12), done.
remote: Compressing objects: 100% (8/8), done.
remote: Total 12 (delta 2), reused 8 (delta 2), pack-reused 0
Unpacking objects: 100% (12/12), 2.03 KiB | 16.00 KiB/s, done.
```

12. Inspect your local copy of repo B both by showing the files and using the log command to make sure everything worked properly.

```
g_copy_hello_world.txt  g_hello_world.txt
gadi@DESKTOP-IEPN1TQ:/mnt/d/Academy/Studying/Bootcamp/ITC Feb 23/Assignments/Git/Solutions/clone_gadiasia7$ git log
commit 09abf011c0d139a4d2d5cea553aa6293fa799d4f (HEAD -> master, origin/master, origin/HEAD)
Author: Gadi Ezer <gadi.g.ezer@gmail.com>
Date:   Fri Mar 10 17:57:51 2023 +0200

    gadiasia 1st commit
```

13. Use a Git command that you haven't used before, show what revision and author last modified each line of the file `hello_world.txt` in repo B. Document the command and its output.

```
gadi@DESKTOP-IEPN1TQ:/mnt/d/Academy/Studying/Bootcamp/ITC Feb 23/Assignments/Git/Solutions/clone_gadiasia7$ git blame g_hello_world.t
xt
09abf011 (Gadi Ezer 2023-03-10 17:57:51 +0200 1) Lorem ipsum dolor sit amet, consectetuer adipiscing elit.
09abf011 (Gadi Ezer 2023-03-10 17:57:51 +0200 2) Aenean commodo ligula eget dolor.
09abf011 (Gadi Ezer 2023-03-10 17:57:51 +0200 3) Aenean massa.
09abf011 (Gadi Ezer 2023-03-10 17:57:51 +0200 4) Cum sociis natoque aaazzzaaazzzaaazzz penatibus et magnis dis parturient montes, nas
cetur ridiculus mus.
09abf011 (Gadi Ezer 2023-03-10 17:57:51 +0200 5) Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem.
09abf011 (Gadi Ezer 2023-03-10 17:57:51 +0200 6) Nulla consequat massa quis enim.
09abf011 (Gadi Ezer 2023-03-10 17:57:51 +0200 7) Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu.
09abf011 (Gadi Ezer 2023-03-10 17:57:51 +0200 8)
```

Part 3: Making Some Changes

Questions:

14. In repo B, open the file `hello_world.txt` and add a few random letters between the existing ones in all instances of the word "Aenean".

```
gadi@DESKTOP-IEPN1TQ:/mnt/d/Academy/Studying/Bootcamp/ITC Feb 23/Assignments/Git/Solutions/clone_gadiasia7$ vim g_hello_world.txt
```

15.Check the status of repo B before staging your changes.

```
gadi@DESKTOP-IEPN1TQ:/mnt/d/Academy/Studying/Bootcamp/ITC Feb 23/Assignments/Git/Solutions/clone_gadiasia7$ git status
On branch master
Your branch is up to date with 'origin/master'.

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:   g_hello_world.txt

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

16.Stage your changes and check the status of repo B again.

```
gadi@DESKTOP-IEPN1TQ:/mnt/d/Academy/Studying/Bootcamp/ITC Feb 23/Assignments/Git/Solutions/clone_gadiasia7$ git add -A
gadi@DESKTOP-IEPN1TQ:/mnt/d/Academy/Studying/Bootcamp/ITC Feb 23/Assignments/Git/Solutions/clone_gadiasia7$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   g_hello_world.txt
```

17.Commit your staged changes and check the status of repo B again.

```
gadi@DESKTOP-IEPN1TQ:/mnt/d/Academy/Studying/Bootcamp/ITC Feb 23/Assignments/Git/Solutions/clone_gadiasia7$ git commit -m "clone_gadi
asia7 1st commit"
[master 06c1286] clone_gadiasia7 1st commit
 1 file changed, 2 insertions(+), 2 deletions(-)
gadi@DESKTOP-IEPN1TQ:/mnt/d/Academy/Studying/Bootcamp/ITC Feb 23/Assignments/Git/Solutions/clone_gadiasia7$ git status
On branch master
Your branch is ahead of 'origin/master' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
```

18.Push your changes to the remote repo. Note that once you clone a repository, you don't need to add a remote URL as we did before because it's predefined in the Git repository settings.

```
gadi@DESKTOP-IEPN1TQ:/mnt/d/Academy/Studying/Bootcamp/ITC Feb 23/Assignments/Git/Solutions/clone_gadiasia7$ git remote -v
origin  https://github.com/Gadiasia/HOME_WORK_ITC.git (fetch)
origin  https://github.com/Gadiasia/HOME_WORK_ITC.git (push)
gadi@DESKTOP-IEPN1TQ:/mnt/d/Academy/Studying/Bootcamp/ITC Feb 23/Assignments/Git/Solutions/clone_gadiasia7$ git push
Username for 'https://github.com': gadiasia
Password for 'https://gadiasia@github.com':
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 314 bytes | 39.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/Gadiasia/HOME_WORK_ITC.git
   09abf01..06c1286  master -> master
```

19. Check the status of the local repo after pushing. Did the status change from what you saw the last time you checked?

```
gadi@DESKTOP-IEPN1TQ:/mnt/d/Academy/Studying/Bootcamp/ITC Feb 23/Assignments/Git/Solutions/clone_gadiasia7$ git status
On branch master
Your branch is up to date with 'origin/master'.

nothing to commit, working tree clean
```

20. Take a look at repo B's log to make sure that everything worked properly.

```
gadi@DESKTOP-IEPN1TQ:/mnt/d/Academy/Studying/Bootcamp/ITC Feb 23/Assignments/Git/Solutions/clone_gadiasia7$ git log
commit 06c1286ec9395c020add852591d97f102a8edd64 (HEAD -> master, origin/master, origin/HEAD)
Author: Gadi Ezer <gadi.g.ezer@gmail.com>
Date:   Fri Mar 10 19:21:19 2023 +0200

    clone_gadiasia7 1st commit

commit 09abf011c0d139a4d2d5cea553aa6293fa799d4f
Author: Gadi Ezer <gadi.g.ezer@gmail.com>
Date:   Fri Mar 10 17:57:51 2023 +0200

    gadiasia 1st commit
```

21. Verify that your change was pushed by going to the URL for the remote repo B on GitHub. Once your collaborator finishes the above, verify that repo A was updated on GitHub as well. (No details needed in your answer, just verify that you did this.)

    Done

Part 4: Houston, we have a problem

Questions:

22. **Once your colleague has finished the previous questions**, navigate to repo A. Open the file `hello_world.txt` and remove all instances of the word “Aenean”. Commit this change and make a push. What happens? Why?

```
gadi@DESKTOP-IEPN1TQ:/mnt/d/Academy/Studying/Bootcamp/ITC Feb 23/Assignments/Git/Solutions/Ex1 - Gitting Started$ vim hello_world.txt

gadi@DESKTOP-IEPN1TQ:/mnt/d/Academy/Studying/Bootcamp/ITC Feb 23/Assignments/Git/Solutions/Ex1 - Gitting Started$ git add hello_world
.txt
gadi@DESKTOP-IEPN1TQ:/mnt/d/Academy/Studying/Bootcamp/ITC Feb 23/Assignments/Git/Solutions/Ex1 - Gitting Started$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   hello_world.txt

gadi@DESKTOP-IEPN1TQ:/mnt/d/Academy/Studying/Bootcamp/ITC Feb 23/Assignments/Git/Solutions/Ex1 - Gitting Started$ git commit -m "2nd
commit by Gadi Ezer"
[master 840c755] 2nd commit by Gadi Ezer
 1 file changed, 2 insertions(+), 2 deletions(-)
```

```
gadi@DESKTOP-IEPN1TQ:/mnt/d/Academy/Studying/Bootcamp/ITC Feb 23/Assignments/Git/Solutions/Ex1 - Gitting Started$ git push origin mas
ter
Username for 'https://github.com': Gadi-G-Ezer
Password for 'https://Gadi-G-Ezer@github.com':
To https://github.com/Gadi-G-Ezer/Git_HW.git
 ! [rejected]        master -> master (fetch first)
error: failed to push some refs to 'https://github.com/Gadi-G-Ezer/Git_HW.git'
hint: Updates were rejected because the remote contains work that you do
hint: not have locally. This is usually caused by another repository pushing
hint: to the same ref. You may want to first integrate the remote changes
hint: (e.g., 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
```

Can’t push before because local repo A is one or more commits behind the remote repo A. Pull (or fetch and merge is required before pushing new changes to the remote)

23. Use the `git pull --no-rebase` command in order to pull the changes from the remote repository. What happens? Why?

```
gadi@DESKTOP-IEPN1TQ:/mnt/d/Academy/Studying/Bootcamp/ITC Feb 23/Assignments/Git/Solutions/Ex1 - Gitting Started$ git pull --no-rebas
e
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (1/1), done.
remote: Total 3 (delta 2), reused 3 (delta 2), pack-reused 0
Unpacking objects: 100% (3/3), 281 bytes | 10.00 KiB/s, done.
From https://github.com/Gadi-G-Ezer/Git_HW
   3a53375..036f66c  master     -> origin/master
There is no tracking information for the current branch.
Please specify which branch you want to merge with.
See git-pull(1) for details.

    git pull <remote> <branch>

If you wish to set tracking information for this branch you can do so with:

    git branch --set-upstream-to=<remote>/<branch> master
```

```
gadi@DESKTOP-IEPN1TQ:/mnt/d/Academy/Studying/Bootcamp/ITC Feb 23/Assignments/Git/Solutions/Ex1 - Gitting Started$ git log --reflog --
graph
* commit 840c755d3c3104562f2e0a79b63243dd86a495af (HEAD -> master)
| Author: Gadi Ezer <gadi.g.ezer@gmail.com>
| Date:   Fri Mar 10 19:37:35 2023 +0200
|
|     2nd commit by Gadi Ezer
|
| * commit 036f66c40217c9cbb60998cfacf1aa74994d3954 (origin/master)
|/  Author: Gadi Ezer <gadi.g.ezer@gmail.com>
|   Date:   Fri Mar 10 19:30:07 2023 +0200
|
|       clone_Gadi-Ezer 1st commit
|
* commit 3a533755f48edbcbb97f9c5f1c972f581f1f9cc9 (gadiasia7/master)
| Author: Gadi Ezer <gadi.g.ezer@gmail.com>
| Date:   Thu Mar 9 16:46:19 2023 +0200
|
|     2nd commit
|
* commit 55cad8c4fd148d087dc6a5f8a11ed5be2b5a76ff
  Author: Gadi Ezer <gadi.g.ezer@gmail.com>
  Date:   Thu Mar 9 15:39:59 2023 +0200

      1st commit
```

Since I didn't specify which branch locally to merge with, A new branch was made and need with the content of the pull.

```
gadi@DESKTOP-IEPN1TQ:/mnt/d/Academy/Studying/Bootcamp/ITC Feb 23/Assignments/Git/Solutions/Ex1 - Gitting Started$ git merge 036f66c40
217c9cbb60998cfacf1aa74994d3954
Auto-merging hello_world.txt
CONFLICT (content): Merge conflict in hello_world.txt
Automatic merge failed; fix conflicts and then commit the result.
```

Once I merger the new branch (i.e. commit 036f) with the local master branch (i.e. commit 840c) a conflict is raised as hello_world.txt content is not the same and need to determine what version to be adopted.

24.Run `git status` and explain its output.

```
gadi@DESKTOP-IEPN1TQ:/mnt/d/Academy/Studying/Bootcamp/ITC Feb 23/Assignments/Git/Solutions/Ex1 - Gitting Started$ git status
On branch master
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:   hello_world.txt

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

There is a conflict between the local version of hellow_world.txt and the version, of the same file, that was just pulled from the remote repo.

25. Use a Git command to inspect the difference between the fetched version of the file `hello_world.txt` and the local version of it. Does it make sense to you? Document the command and its output.

```
gadi@DESKTOP-IEPN1TQ:/mnt/d/Academy/Studying/Bootcamp/ITC Feb 23/Assignments/Git/Solutions/Ex1 - Gitting Started$ git diff hello_worl
d.txt
diff --cc hello_world.txt
index ae0bf91,80ec168..0000000
mode 100644,100644..100755
--- a/hello_world.txt
+++ b/hello_world.txt
@@@ -1,6 -1,6 +1,11 @@@
  Lorem ipsum dolor sit amet, consectetuer adipiscing elit.
++<<<<<<< HEAD
 + commodo ligula eget dolor.
 + massa.
++=======
+ A777777777enean commodo ligula eget dolor.
+ Aenea9999999999n massa.
++>>>>>>> 036f66c4021 7c9cbb60998cfacf1aa74994d3954
  Cum sociis natoque aaazzzaaazzzaaazzz penatibus et magnis dis parturient montes, nascetur ridiculus mus.
  Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem.
  Nulla consequat massa quis enim.
```

It make sense. On the local repo, all ‘Anmen’ words were deleted while on the remote repo additional characters were added to .

26. Edit `hello_world.txt` to resolve the conflict however you would like. Then commit and push your changes.

```
gadi@DESKTOP-IEPN1TQ:/mnt/d/Academy/Studying/Bootcamp/ITC Feb 23/Assignments/Git/Solutions/Ex1 - Gitting Started$ git add hello_world
.txt
gadi@DESKTOP-IEPN1TQ:/mnt/d/Academy/Studying/Bootcamp/ITC Feb 23/Assignments/Git/Solutions/Ex1 - Gitting Started$ git log --reflog --
graph
* commit 840c755d3c3104562f2e0a79b63243dd86a495af (HEAD -> master)
| Author: Gadi Ezer <gadi.g.ezer@gmail.com>
| Date:   Fri Mar 10 19:37:35 2023 +0200
|
|     2nd commit by Gadi Ezer
|
| * commit 036f66c40217c9cbb60998cfacf1aa74994d3954 (origin/master)
|/  Author: Gadi Ezer <gadi.g.ezer@gmail.com>
|   Date:   Fri Mar 10 19:30:07 2023 +0200
|
|       clone_Gadi-Ezer 1st commit
|
```

After staging the edited hello_word.txt branches are not yet merged.

```
gadi@DESKTOP-IEPN1TQ:/mnt/d/Academy/Studying/Bootcamp/ITC Feb 23/Assignments/Git/Solutions/Ex1 - Gitting Started$ git commit -m "fixe
d conflict by commit "
[master 84a3036] fixed conflict by commit
gadi@DESKTOP-IEPN1TQ:/mnt/d/Academy/Studying/Bootcamp/ITC Feb 23/Assignments/Git/Solutions/Ex1 - Gitting Started$ git log --reflog --
graph
*   commit 84a3036411212000be9745dbabe7edb86b3e2477 (HEAD -> master)
|\  Merge: 840c755 036f66c
| | Author: Gadi Ezer <gadi.g.ezer@gmail.com>
| | Date:   Fri Mar 10 21:21:06 2023 +0200
| |
| |     fixed conflict by commit
| |
| * commit 036f66c40217c9cbb60998cfacf1aa74994d3954 (origin/master)
| | Author: Gadi Ezer <gadi.g.ezer@gmail.com>
| | Date:   Fri Mar 10 19:30:07 2023 +0200
| |
| |     clone_Gadi-Ezer 1st commit
| |
* | commit 840c755d3c3104562f2e0a79b63243dd86a495af
|/  Author: Gadi Ezer <gadi.g.ezer@gmail.com>
|   Date:   Fri Mar 10 19:37:35 2023 +0200
|
|       2nd commit by Gadi Ezer
|
```

After committing the staged edited file hello_word.txt, branches were merged to a ew commit where HEAD is pointing to master branch.

27. Run `git log --all --decorate --oneline --graph` to view your repo's history as a graph. Explain what you see.

```
gadi@DESKTOP-IEPN1TQ:/mnt/d/Academy/Studying/Bootcamp/ITC Feb 23/Assignments/Git/Solutions/Ex1 - Gitting Started$ git log --all --dec
orate --oneline --graph
*   84a3036 (HEAD -> master) fixed conflict by commit
|\
| * 036f66c (origin/master) clone_Gadi-Ezer 1st commit
* | 840c755 2nd commit by Gadi Ezer
|/
* 3a53375 (gadiasia7/master) 2nd commit
* 55cad8c 1st commit
```

(The firs two commits (commit# 55ca and commit#3a53) are not related to this exercise).

Before the merge, Head pointed to commit#840c and the pull from the remote repo created a branch of commit#036f. when tried to merge between these two commits a conflict was raised as the same file hello_world.txt had different content in each branch. After solving the conflict, by editing the file staging it and commit it, the two branches were merged into a new commit# 84a3 where the HEAD point the master branch.

```
gadi@DESKTOP-IEPN1TQ:/mnt/d/Academy/Studying/Bootcamp/ITC Feb 23/Assignments/Git/Solutions/Ex1 - Gitting Started$ git push origin mas
ter
Username for 'https://github.com': Gadi-G-Ezer
Password for 'https://Gadi-G-Ezer@github.com':
Enumerating objects: 10, done.
Counting objects: 100% (10/10), done.
Delta compression using up to 8 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (6/6), 585 bytes | 65.00 KiB/s, done.
Total 6 (delta 4), reused 0 (delta 0)
remote: Resolving deltas: 100% (4/4), completed with 2 local objects.
To https://github.com/Gadi-G-Ezer/Git_HW.git
   036f66c..84a3036  master -> master
```

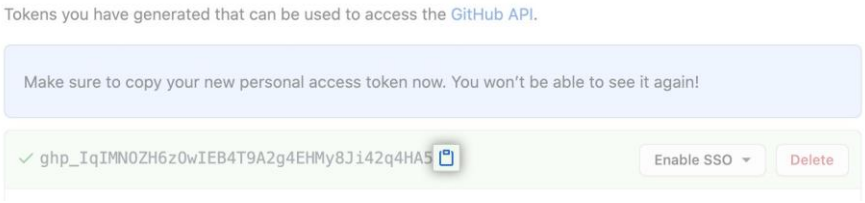
Finally, can push the changes to the remote repo,

Final note: The next Git exercise also uses repos A and B. Don’t delete them yet!

Appendix 1: Personal Access Token (PAT) Authentication

When you run commands like “git pull/push”, you are connecting to GitHub and must authenticate your identity. Since August 2021, GitHub no longer allows username-password authentication from the command line, and you are recommended to use a PAT in place of a password.

Follow the instructions from the GitHub documentation to create a PAT via the GitHub website. When you generate a PAT you will see something like the following:



You can then use it on the command line for Git commands that connect to GitHub (paste it in the “Password:” field):

```
$ git clone https://github.com/username/repo.git
Username: (your GitHub username)
Password: (your PAT)
```

Your PAT is like a password – keep it secret!

If you want Git to remember your PAT so you don’t have to enter it each time you run such commands, you may use credential caching. See: Caching your GitHub credentials in Git

For more information and a step-by-step tutorial, see: Authenticating with Github

Appendix 2: SSH Key Authentication

To use SSH keys for connecting to GitHub, you may follow the instructions in the GitHub documentation: Connecting to GitHub with SSH

The idea in **public-key cryptography** is that instead of a username and password, users hold two "keys" (usually long random-looking strings stored in files): a **private key** and a **public key**. On *nix systems (and in Git Bash) these are files stored in the hidden `~/.ssh` directory.

The **private key** should be treated like a password and kept private. Don't share your private key with anyone! The private key will have some filename like *id_rsa* (no file extension).

The **public key** may be shared with anyone. It can be used by any party to verify the identity of the user holding the corresponding private key. The public key will have some filename like *id_rsa.pub* (same as private key but ending in *.pub*).

If *~/.ssh* does not yet exist or does not contain any key pairs, you may generate a key pair with the instructions at: Generating a new SSH key and adding it to the ssh-agent

Once you generate them, try viewing their contents (e.g. with the *nix `less` command).

You may add your **public key** (NOT PRIVATE) to your GitHub account's security settings with the instructions at: Adding a new SSH key to your GitHub account

You now should be able to use Git commands like `git pull` from **the user account on the computer which has the corresponding private key file.**