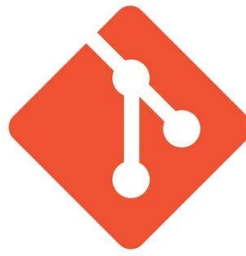


## *Использование системы контроля версий Git*



В рамках данной работы, было использовано приложение эмулирующее работу командой строки Git – Git Bash.

В начале, мы выбрали каталог, в котором мы создадим репозиторий. Чтобы перейти в этот каталог мы использовали команду `cd C:/Practice`:

```
Honor@LapTop MINGW64 ~ (master)
$ cd C:\Practice
```

Чтобы создать новый репозиторий, используем команду `git init`, также чтобы удостовериться, что создание репозитория прошло успешно, воспользуемся командой `ls -la`:

```
Honor@LapTop MINGW64 /c/Practice
$ git init
Initialized empty Git repository in C:/Practice/.git/

Honor@LapTop MINGW64 /c/Practice (master)
$ ls -la
total 12
drwxr-xr-x 1 Honor 197609 0 Sep 26 22:31 ./
drwxr-xr-x 1 Honor 197609 0 Sep 26 22:29 ../
drwxr-xr-x 1 Honor 197609 0 Sep 26 22:31 .git/
```

Команда `git status` позволяет просмотреть изменения и создание коммитов, то есть статус репозитория:

```
Honor@LapTop MINGW64 /c/Practice (master)
$ git status
On branch master

No commits yet

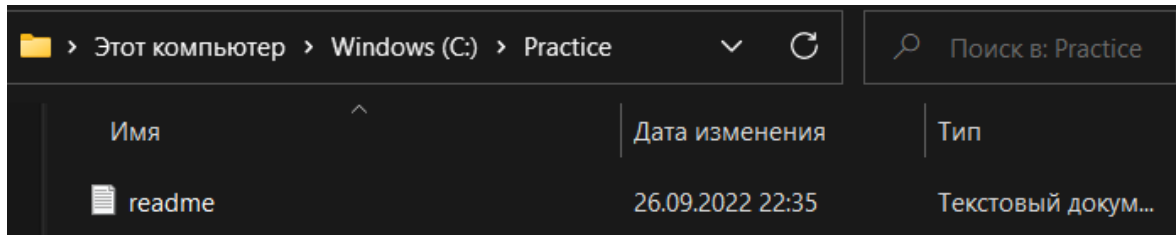
nothing to commit (create/copy files and use "git add" to track)
```

Коммитов нет, а также написано, что мы находимся на ветке `master`.

С помощью команды `touch` создадим текстовый файл `touch readme.txt`:

```
Honor@LapTop MINGW64 /c/Practice (master)
$ touch readme.txt
```

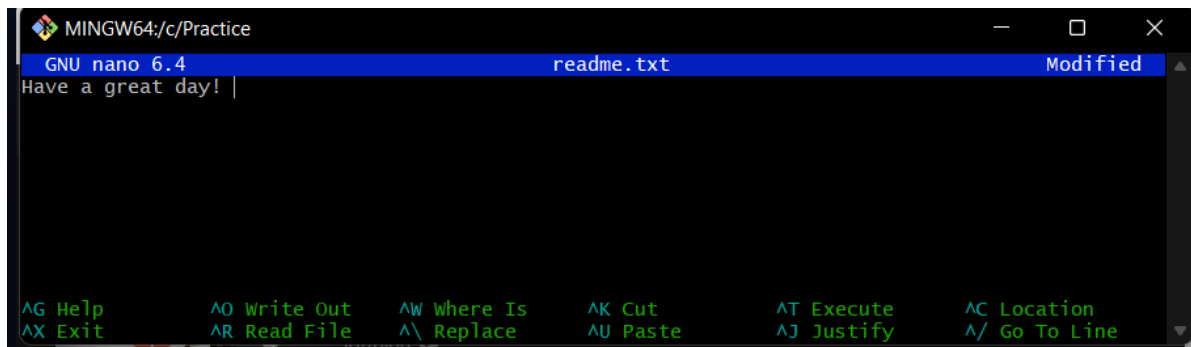
Данный текстовый файл появился в проводнике:



Команда nano поможет нам добавить/изменить текст в readme, для этого введем nano readme.txt

```
Honor@LapTop MINGW64 /c/Practice (master)
$ nano readme.txt
```

Вводим любой текст и сохраняем с помощью ctrl+s, чтобы выйти с редактора нажимаем ctrl+x



Проверяя статус git, обнаружим новый не отслеживаемый файл readme.txt, который мы только что создали

```
Honor@LapTop MINGW64 /c/Practice (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
  readme.txt

nothing added to commit but untracked files present (use "git add" to track)
```

Так как этот файл не отслеживаемый, то его нужно как-то начать отслеживать и в этом нам поможет команда add, для этого вводим git add readme.txt

```
Honor@LapTop MINGW64 /c/Practice (master)
$ git add readme.txt
warning: in the working copy of 'readme.txt', LF will be replaced by CRLF the next time Git touches it
```

Проверим статус git еще раз

```
Honor@LapTop MINGW64 /c/Practice (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   readme.txt
```

Теперь, файл readme отслеживается и мы можем это закоммитить, что мы сейчас и сделаем с помощью команды `commit -m` “комментарий о нашем действии”. Введем `git commit -m “First file”`

```
Honor@LapTop MINGW64 /c/Practice (master)
$ git commit -m "First file"
[master (root-commit) 8172236] First file
 1 file changed, 1 insertion(+)
 create mode 100644 readme.txt
```

Проверив статус, мы увидим, что больше нечего коммитить.

Изменим текст readme, используем знакомую команду `nano`

```
GNU nano 6.4                               readme.txt                               Modified
Have a wonderful day!
```

^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




Проверив статус, увидим изменения красным цветом, значит нужно снова добавить его с помощью команды `add`.

```
Honor@LapTop MINGW64 /c/Practice (master)
$ git status
On branch 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:   readme.txt

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

Honor@LapTop MINGW64 /c/Practice (master)
$ git add readme.txt
warning: in the working copy of 'readme.txt', LF will be replaced by CRLF the next time Git touch
es it
```

Создадим 2 копии текстового файла readme: `readmefirst` и `readmesecond`. Добавим их с помощью команды `git add “*.txt”`

Имя	Дата изменения	Тип
 readme	26.09.2022 22:40	Текстовый докум...
 readmefirst	26.09.2022 22:40	Текстовый докум...
 readmesecond	26.09.2022 22:40	Текстовый докум...

```
Honor@LapTop MINGW64 /c/Practice (master)
$ git add "*.txt"
warning: in the working copy of 'readmefirst.txt', LF will be replaced by CRLF the next time Git
touches it
warning: in the working copy of 'readmesecond.txt', LF will be replaced by CRLF the next time Git
touches it
```

Снова проверяем статус

```
Honor@LapTop MINGW64 /c/Practice (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   readme.txt
        new file:   readmefirst.txt
        new file:   readmesecond.txt
```

Видим, что readme был изменен, а наши 2 копии появились. Теперь можно их закоммитить.

```
Honor@LapTop MINGW64 /c/Practice (master)
$ git commit -m "All txt files were added"
[master 99b2231] All txt files were added
 3 files changed, 3 insertions(+), 1 deletion(-)
 create mode 100644 readmefirst.txt
 create mode 100644 readmesecond.txt
```

Чтобы просмотреть историю коммитов, используется команда git log. Будут показаны комментарии к действиям, кем и когда они были сделаны, первым будет показано последнее изменение.

```
Honor@LapTop MINGW64 /c/Practice (master)
$ git log
commit 99b223117ea03586e23147b745de305dc0c9c90e (HEAD -> master)
Author: Amira Zuhir <zuhiramira@gmail.com>
Date:   Mon Sep 26 22:49:54 2022 +0300

    All txt files were added

commit 8172236f89a533a55ecfe08e11e5dfa6c6387aa7
Author: Amira Zuhir <zuhiramira@gmail.com>
Date:   Mon Sep 26 22:39:17 2022 +0300

    First file
```

Для более детальной информации используется команда git log --summary.

```
Honor@LapTop MINGW64 /c/Practice (master)
$ git log --summary
commit 99b223117ea03586e23147b745de305dc0c9c90e (HEAD -> master)
Author: Amira Zuhir <zuhiramira@gmail.com>
Date:   Mon Sep 26 22:49:54 2022 +0300

    All txt files were added

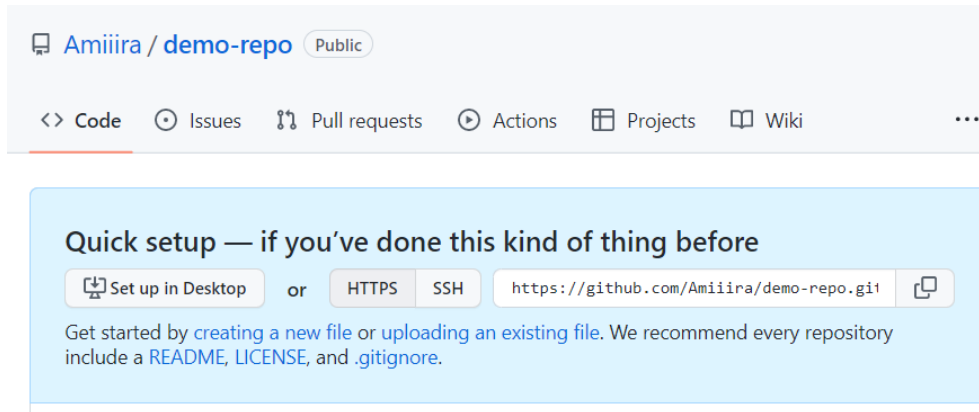
    create mode 100644 readmefirst.txt
    create mode 100644 readmesecond.txt

commit 8172236f89a533a55ecfe08e11e5dfa6c6387aa7
Author: Amira Zuhir <zuhiramira@gmail.com>
Date:   Mon Sep 26 22:39:17 2022 +0300

    First file

    create mode 100644 readme.txt
```

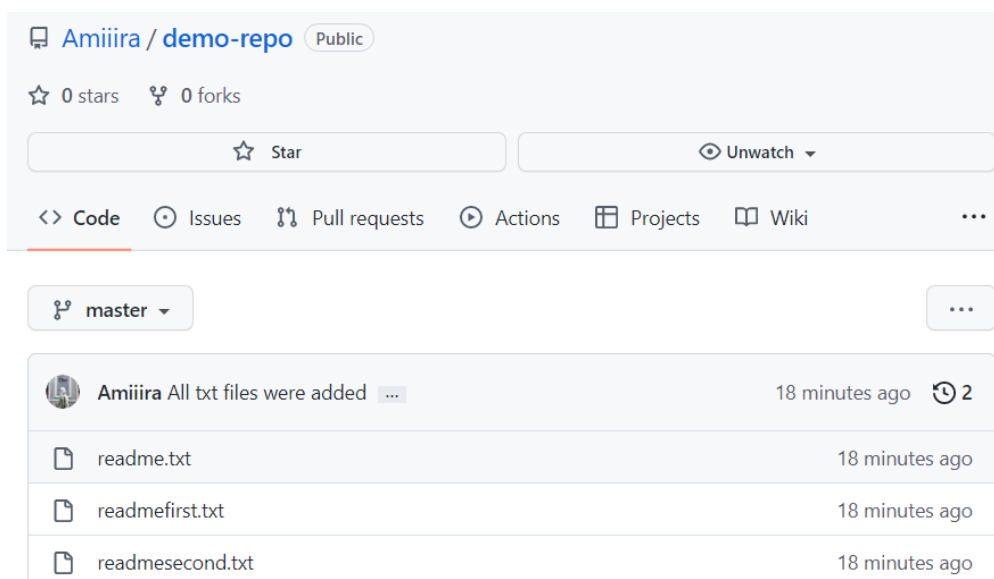
Чтобы выгрузить репозиторий на удаленный репозиторий, переходим в GitHub и создаем пустой репозиторий и копируем URL адрес.



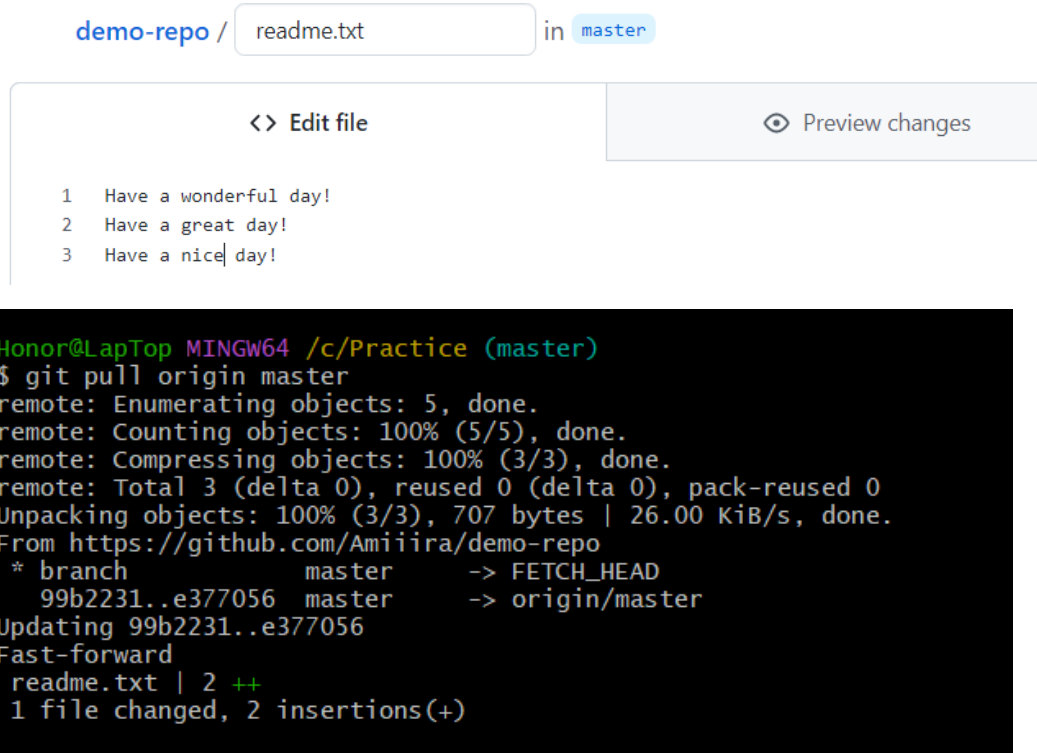
Команда `git remote add origin URL` адрес поможет добавить нам удаленный репозиторий, а команда `git push -u origin master` поможет нам выгрузить файлы туда, `master` это ветка с которой мы работаем. Далее нужно авторизоваться через GitHub. После этого мы получим информацию о выгрузке. Также, зайдя на GitHub репозиторий, мы увидим наши файлы, причем указывается время создания файла, а не его появления на Гитхаб.

```
Honor@LapTop MINGW64 /c/Practice (master)
$ git remote add origin https://github.com/Amiiira/demo-repo.git

Honor@LapTop MINGW64 /c/Practice (master)
$ git push -u origin master
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (6/6), 499 bytes | 249.00 KiB/s, done.
Total 6 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/Amiiira/demo-repo.git
* [new branch]      master -> master
branch 'master' set up to track 'origin/master'.
```



Допустим при работе в команде, кто-то выгрузил файлы с изменениями и вам нужно получить эти файлы обратно. Для этого используем команду `git pull origin master`. Изменим файл `readme` в удаленном репозитории и вернем файлы обратно.



The image shows a GitHub repository interface for 'demo-repo' with the file 'readme.txt' selected in the 'master' branch. The file content is:

```
1 Have a wonderful day!  
2 Have a great day!  
3 Have a nice day!
```

Below the file view is a terminal window showing the execution of a `git pull` command:

```
Honor@LapTop MINGW64 /c/Practice (master)  
$ git pull origin master  
remote: Enumerating objects: 5, done.  
remote: Counting objects: 100% (5/5), done.  
remote: Compressing objects: 100% (3/3), done.  
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0  
Unpacking objects: 100% (3/3), 707 bytes | 26.00 KiB/s, done.  
From https://github.com/Amiiira/demo-repo  
* branch      master      -> FETCH_HEAD  
   99b2231..e377056  master    -> origin/master  
Updating 99b2231..e377056  
Fast-forward  
  readme.txt | 2 ++  
  1 file changed, 2 insertions(+)
```

Видим подробную информацию об изменениях, а точнее что прибавилось 2 строчки.

Также, мы можем проверить насколько отличны закоммиченные файлы и файлы которые мы сами изменили. Для этой цели используется команда `git diff HEAD`. Head это указатель на закоммиченную версию файла.

Для начала изменим `readme` убрав восклицательные знаки.



The image shows a terminal window with the GNU nano 6.4 text editor open, editing the file 'readme.txt'. The content is:

```
Have a wonderful day  
Have a great day  
Have a nice day|
```

```
Honor@LapTop MINGW64 /c/Practice (master)
$ nano readme.txt

Honor@LapTop MINGW64 /c/Practice (master)
$ git diff HEAD
diff --git a/readme.txt b/readme.txt
index a145a24..f82913f 100644
--- a/readme.txt
+++ b/readme.txt
@@ -1,3 +1,3 @@
-Have a wonderful day!
-Have a great day!
-Have a nice day!
+Have a wonderful day
+Have a great day
+Have a nice day
```

Видны все изменения.

Также, можно добавить каталог в наш репозиторий с помощью команды `mkdir folder`, `folder` это название каталога.

```
Honor@LapTop MINGW64 /c/Practice (master)
$ mkdir folder
```

Имя	Дата изменения	Тип
folder	26.09.2022 23:24	Папка с файлами
readme	26.09.2022 23:22	Текстовый докум...
readmefirst	26.09.2022 22:40	Текстовый докум...
readmesecond	26.09.2022 22:40	Текстовый докум...


Создадим текстовый файл в каталоге `folder` с помощью команды `touch folder/bye.txt`

```
Honor@LapTop MINGW64 /c/Practice (master)
$ touch folder/bye.txt
```

« Windows (C:) » Practice » folder

▼ ↺

🔍 Поиск в: folder

Имя	Дата изменения	Тип
 bye	26.09.2022 23:24	Текстовый докум...

Снова проверим статус репозитория.

```
Honor@LapTop MINGW64 /c/Practice (master)
$ 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:   readme.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        folder/

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

Видим, что каталог появился, но он не отслеживается, поэтому добавим его и его файл с помощью команды git add.

```
Honor@LapTop MINGW64 /c/Practice (master)
$ git add folder/bye.txt

Honor@LapTop MINGW64 /c/Practice (master)
$ 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)
        new file:   folder/bye.txt

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:   readme.txt
```

После проверки статуса, видим, что теперь мы можем закоммитить каталог и его файл.

Команда git diff --staged позволит нам просмотреть изменения на стадии staged.

```
Honor@LapTop MINGW64 /c/Practice (master)
$ git diff --staged
diff --git a/folder/bye.txt b/folder/bye.txt
new file mode 100644
index 0000000..e69de29
```

Команда git reset позволяет отменить изменения.

```
Honor@LapTop MINGW64 /c/Practice (master)
$ git reset folder/bye.txt
Unstaged changes after reset:
M       readme.txt
```

После данной команды видим, что изменений нет

```
Honor@LapTop MINGW64 /c/Practice (master)
$ git diff --staged

Honor@LapTop MINGW64 /c/Practice (master)
$
```



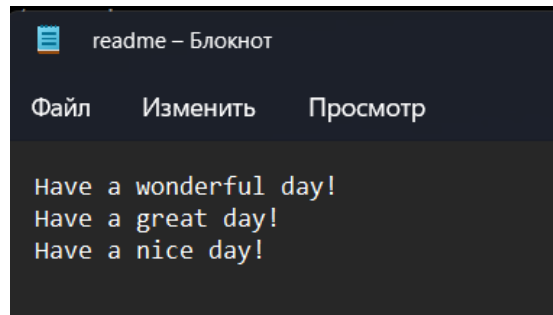
Файл можно вернуть в исходное состояние с помощью команды `git checkout - readme.txt`. Посмотрим также статус `git` и проверить `readme.txt` с помощью команды `cat readme.txt`, которая выведет текст прямо на консоль.

```
Honor@LapTop MINGW64 /c/Practice (master)
$ git checkout -- readme.txt

Honor@LapTop MINGW64 /c/Practice (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
  folder/

nothing added to commit but untracked files present (use "git add" to track)
```



```
Honor@LapTop MINGW64 /c/Practice (master)
$ cat readme.txt
Have a wonderful day!
Have a great day!
Have a nice day!
```

Создадим еще одну ветку, которая будет называться `clean_up`. Воспользуемся командой `git branch clean_up`.

```
Honor@LapTop MINGW64 /c/Practice (master)
$ git branch clean_up

Honor@LapTop MINGW64 /c/Practice (master)
$ git branch
  clean_up
* master
```

Команда `git branch` показывает на какой ветке мы сейчас находимся. Чтобы оказаться на другой ветке, будет использовать команду `git checkout`.

```
Honor@LapTop MINGW64 /c/Practice (master)
$ git checkout clean_up
Switched to branch 'clean_up'

Honor@LapTop MINGW64 /c/Practice (clean_up)
$ git branch
* clean_up
  master
```

Команда `rm -r folder` поможет нам удалить каталог и файлы на этой ветке.

```
Honor@LapTop MINGW64 /c/Practice (clean_up)
$ rm -r folder
```

```
Honor@LapTop MINGW64 /c/Practice (clean_up)
$ rm -r folder

Honor@LapTop MINGW64 /c/Practice (clean_up)
$ git commit -m "DDeleted folder and files"
On branch clean_up
nothing to commit, working tree clean




Honor@LapTop MINGW64 /c/Practice (clean_up)
$ git status
On branch clean_up
nothing to commit, working tree clean

Honor@LapTop MINGW64 /c/Practice (clean_up)
$ git rm readmefirst.txt
rm 'readmefirst.txt'


Honor@LapTop MINGW64 /c/Practice (clean_up)
$ git rm readmeseccond.txt
rm 'readmeseccond.txt'

Honor@LapTop MINGW64 /c/Practice (clean_up)
$ git status
On branch clean_up
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        deleted:   readmefirst.txt
        deleted:   readmeseccond.txt
```

ДО:

 readme	26.09.2022 23:29	Текстовый докум...
 readmefirst	26.09.2022 22:40	Текстовый докум...
 readmeseccond	26.09.2022 22:40	Текстовый докум...

После:

Имя	Дата изменения	Тип
 readme	26.09.2022 23:29	Текстовый докум...

Зафиксируем изменения, чтобы мы могли выполнить слияние ветвей.

```
Honor@LapTop MINGW64 /c/Practice (clean_up)
$ git commit -m "DDeleted folder and files"
[clean_up 6ff76ae] DDeleted folder and files
2 files changed, 2 deletions(-)
delete mode 100644 readmefirst.txt
delete mode 100644 readmeseccond.txt
```

Т.к мы удалили файлы только с папки clean\_up, а с ветки master, то выполнив их слияние они удалятся автоматически.

Для слияния используем команду git merge clean\_up.

```
Honor@LapTop MINGW64 /c/Practice (clean_up)
$ git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.

Honor@LapTop MINGW64 /c/Practice (master)
$ git merge clean_up
Updating e377056..6ff76ae
Fast-forward
 readmefirst.txt | 1 -
 readmesecound.txt | 1 -
 2 files changed, 2 deletions(-)
 delete mode 100644 readmefirst.txt
 delete mode 100644 readmesecound.txt
```

Этот компьютер > Windows (C:) > Practice			Поиск в: Practice
Имя	Дата изменения	Тип	
readme	26.09.2022 23:29	Текстовый докум...	

Ветвь для очистки clean\_up больше не нужна, избавимся от нее с помощью команды `git branch -d clean_up`.

```
Honor@LapTop MINGW64 /c/Practice (master)
$ git branch -d clean_up
Deleted branch clean_up (was 6ff76ae).
```

Выгрузим все изменения на GitHub.

```
Honor@LapTop MINGW64 /c/Practice (master)
$ git push
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 4 threads
Compressing objects: 100% (1/1), done.
Writing objects: 100% (2/2), 246 bytes | 246.00 KiB/s, done.
Total 2 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/Amiira/demo-repo.git
 e377056..6ff76ae master -> master
```

В итоге, в репозитории на GitHub остался только один файл readme, остальные мы удалили.

master

Amiira DDeleted folder and files

2 minutes ago 4

readme.txt

18 minutes ago

readme.txt

Have a wonderful day!  
Have a great day!  
Have a nice day!