

Департамент образования и науки города Москвы
Государственное автономное образовательное учреждение
высшего образования города Москвы
«Московский городской педагогический университет»
Институт цифрового образования
Департамент информатики управления и технологий

Ли Александр Андреевич БД-241м

**Практическая работа 2.2. Создание и управление репозиториями на
GitHub. Работа с ветками, слияниями, разрешение конфликтов**

Направление подготовки/специальность
38.04.05 - Бизнес-информатика
Бизнес-аналитика и большие данные
(очная форма обучения)
Вариант 12

Москва

2024

| | |
|--|-----------|
| Введение | 3 |
| Основная часть | 3 |
| 1. Создание тестового репозитория | 3 |
| Клонирование репозитория..... | 5 |
| Варианты заданий | 9 |
| Отправка изменений на GitHub: | 13 |
| Настройка SSH для GitHub. | 14 |
| 2. Задание | 17 |
| Индивидуальные задания..... | 20 |
| Заключение | 26 |

Введение

Практическая работа нацелена на знакомство студентов с основными концепциями и командами Git, такими как создание коммитов, инициализация и клонирование репозитория, использование Git для управления версиями проекта, подключение локального с удаленным репозиторием посредством обмена ssh ключами.

Основная часть

1. Создание тестового репозитория

Необходимо создать тестовый репозиторий и добавить в них два файла двумяразными коммитами.

Для начала создадим 2 файла mgpu.txt и index.html.

Файл mgpu.txt с текстом Hello, MGPU!

Файл index.html с текстом <h1> я коммичу</h1>

```
sasha@froge:~$ git init
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:   git config --global init.defaultBranch <name>
hint:
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this command:
hint:
hint:   git branch -m <name>
Initialized empty Git repository in /home/sasha/.git/
sasha@froge:~$
```

Рис.1

Сначала инициализируем репозиторий командой
git init

```
2. sasha@froge: ~
sasha@froge:~$ echo "Hello, MGPU" > mgpu.txt
sasha@froge:~$ ls
Desktop Documents Downloads mgpu.txt Music Pictures Public scripts snap Templates thinclient_drives Videos
sasha@froge:~$
```

Рис.2

Далее создадим файла `mgpu.txt` с текстом " Hello, MGPU!" командой

echo " Hello, MGPU!" > mgpu.txt

и проверим её существование в каталоге при помощи

ls

Добавим файла в индекс и сделаем первый коммит

```
sasha@froge:~$  
sasha@froge:~$  
sasha@froge:~$ git config --global user.email "frogedoge@mail.ru"  
sasha@froge:~$ git config --global user.name "sasha"  
sasha@froge:~$ git config --global --list  
user.email=frogedoge@mail.ru  
user.name=sasha
```

Рис.3

Для начала нужно указать почту и имя пользователя, чтоб Git смог определить личность. Для этого пропишем

git config --global user.email "frogedoge@mail.ru "

git config --global user.name "Sasha"

и **git config --global --list** , чтобы проверить записались ли данные.

```
sasha@froge:~$ git commit -m "Добавлен файл mgpu.txt с текстом 'Hello, MGPU\!'"  
[master (root-commit) 6b265eb] Добавлен файл mgpu.txt с текстом 'Hello, MGPU\!'  
1 file changed, 1 insertion(+)  
create mode 100644 mgpu.txt  
sasha@froge:~$ █
```

Рис.4

После чего пропишем команды для того, чтобы сделать commit

git add mgpu.txt

git commit -m "Добавлен файл mgpu.txt с текстом 'Hello, MGPU\!'"

Создадим второй файл `index.html` с текстом `"<h1> я коммичу</h1>"`

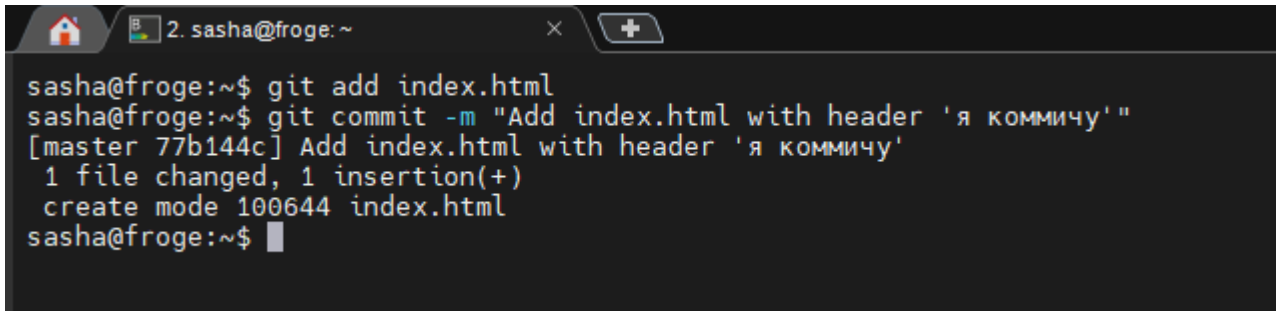
```
sasha@froge:~$ echo "<h1> я коммичу</h1>" > index.html  
sasha@froge:~$ █
```

Рис.5

Для создания второго файла используем команду

echo "<h1> я коммичу</h1>" > index.html

Добавим второй файл в индекс и создадим второй коммит:

A terminal window titled '2. sasha@froge: ~' showing the execution of git commands. The user runs 'git add index.html', then 'git commit -m "Add index.html with header 'я коммичу'"'. The output shows the commit is successful, with '1 file changed, 1 insertion(+)' and 'create mode 100644 index.html'.

```
sasha@froge:~$ git add index.html
sasha@froge:~$ git commit -m "Add index.html with header 'я коммичу'"
[master 77b144c] Add index.html with header 'я коммичу'
1 file changed, 1 insertion(+)
create mode 100644 index.html
sasha@froge:~$
```

Рис.6

git add index.html

git commit -m "Add index.html with header 'я коммичу'"

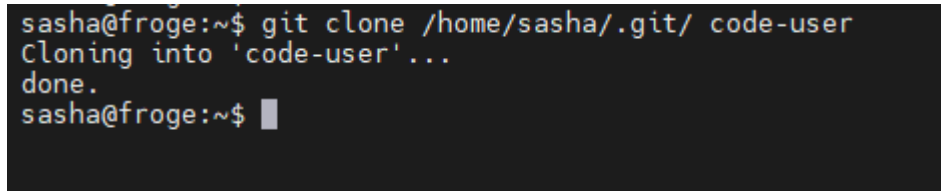
Теперь в репозитории есть два файла, добавленных двумя разными коммитами.

Клонирование репозитория

Для того, чтобы клонировать репозиторий, нужно перейти в необходимый каталог, после чего в нем прописать

git clone /home/sasha/.git code-user

Первый параметр «откуда», второй — «куда»

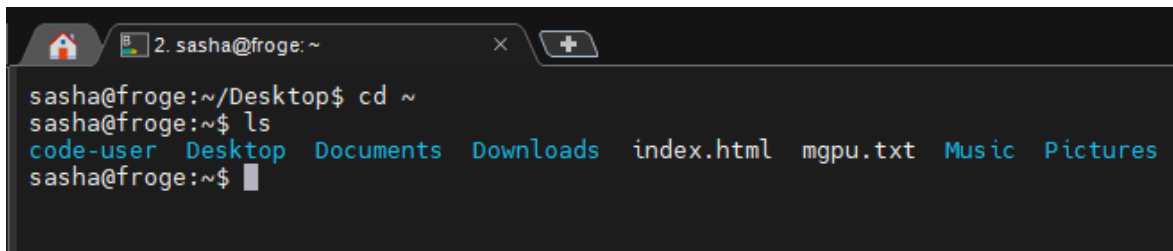
A terminal window showing the execution of 'git clone /home/sasha/.git/ code-user'. The output is 'Cloning into 'code-user'...' followed by 'done.'.

```
sasha@froge:~$ git clone /home/sasha/.git/ code-user
Cloning into 'code-user'...
done.
sasha@froge:~$
```

Рис.7

git clone /home/sasha/.git code-user

После успешного клонирования терминал выдаст **done**.

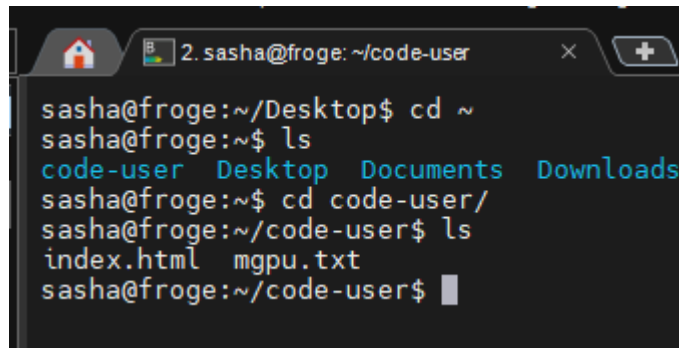
A terminal window showing the user navigating to the home directory and listing files. The command 'cd ~' is followed by 'ls', which lists 'code-user', 'Desktop', 'Documents', 'Downloads', 'index.html', 'mgpu.txt', 'Music', and 'Pictures'.

```
sasha@froge:~/Desktop$ cd ~
sasha@froge:~$ ls
code-user Desktop Documents Downloads index.html mgpu.txt Music Pictures
sasha@froge:~$
```

Рис.8

Проверим создалась ли директория code-user командой:

ls



```
sasha@froge:~/Desktop$ cd ~
sasha@froge:~$ ls
code-user  Desktop  Documents  Downloads
sasha@froge:~$ cd code-user/
sasha@froge:~/code-user$ ls
index.html  mgpu.txt
sasha@froge:~/code-user$
```

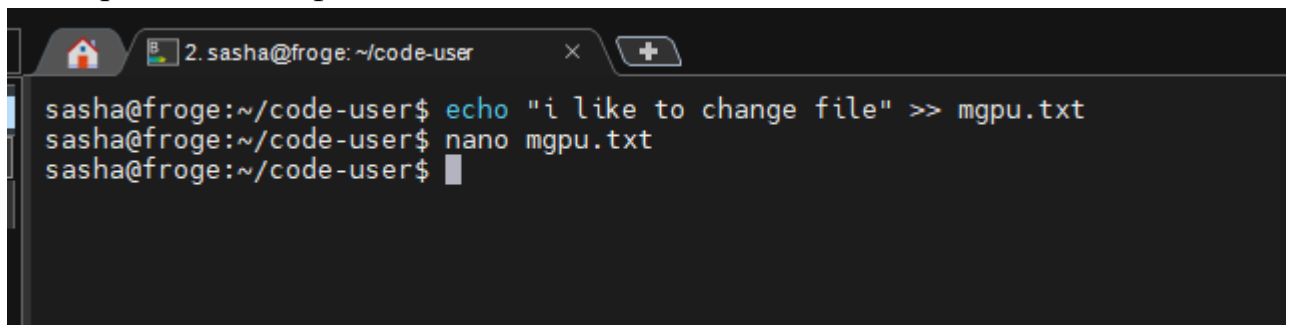
Рис.9

Проверим клонировались ли файлы в code-user перейдя в директорию:

Cd code-user

ls

В репозитории есть два файла. Внесем изменения в них:

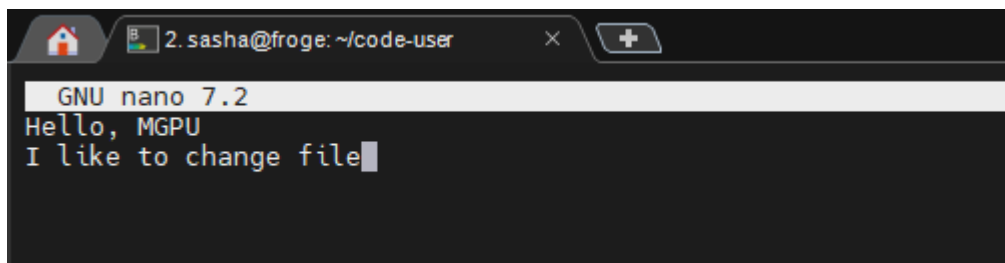


```
sasha@froge:~/code-user$ echo "i like to change file" >> mgpu.txt
sasha@froge:~/code-user$ nano mgpu.txt
sasha@froge:~/code-user$
```

Рис.10

Добавим в mgpu.txt вторую строчку I like to change files

echo "I like to change files" >> mgpu.txt

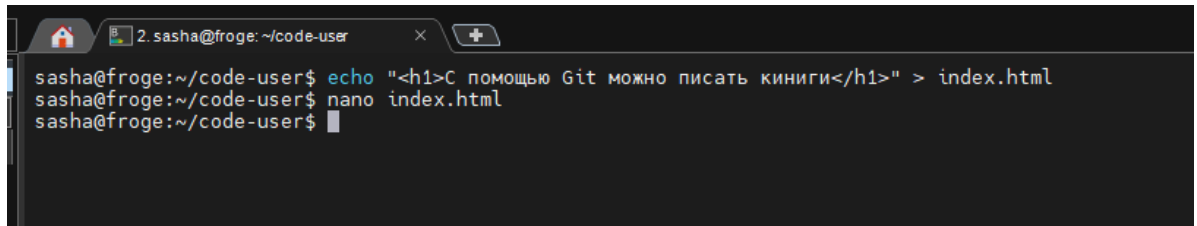


```
GNU nano 7.2
Hello, MGPU
I like to change file
```

Рис.11

Пропишем **nano mgpu.txt**, чтобы проверить изменения

Прделаем аналогичные действия со вторым файлом

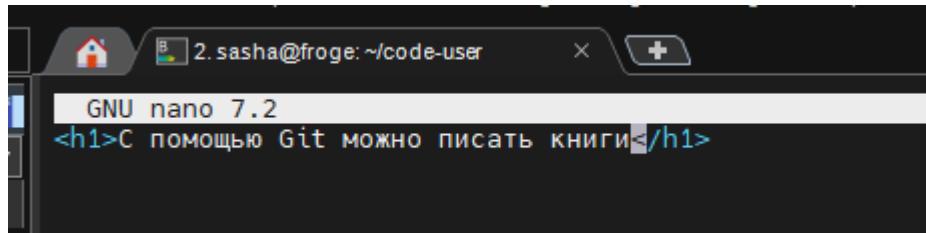


```
sasha@froge:~/code-user$ echo "<h1>С помощью Git можно писать книги</h1>" > index.html
sasha@froge:~/code-user$ nano index.html
sasha@froge:~/code-user$
```

Рис.12

Изменим изначальную строчку командой

echo "<h1>С помощью Git можно писать книги</h1>" > index.html

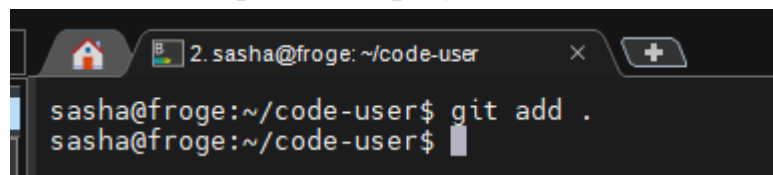


```
GNU nano 7.2
<h1>С помощью Git можно писать книги</h1>
```

Рис.13

Пропишем **nano index.html** чтобы проверить изменения

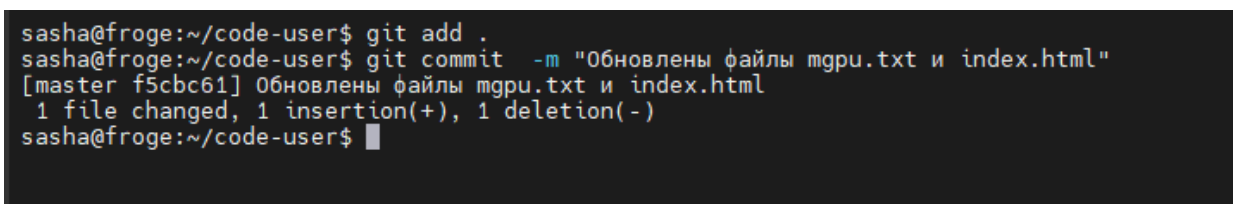
Сделаем один коммит, содержащий сразу два эти изменения.



```
sasha@froge:~/code-user$ git add .
sasha@froge:~/code-user$
```

Рис.14

git add .

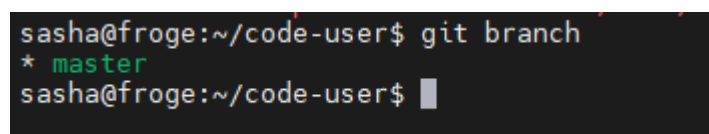


```
sasha@froge:~/code-user$ git add .
sasha@froge:~/code-user$ git commit -m "Обновлены файлы mgpu.txt и index.html"
[master f5cbc61] Обновлены файлы mgpu.txt и index.html
1 file changed, 1 insertion(+), 1 deletion(-)
sasha@froge:~/code-user$
```

Рис.15

git commit -m "Обновлены файлы mgpu.txt и index.html"

Добавим изменения в основной репозиторий с помощью **git push**.



```
sasha@froge:~/code-user$ git branch
* master
sasha@froge:~/code-user$
```

Рис.16

Пропишем **git branch** чтобы узнать название доступных веток.

```
sasha@froge:~/code-user$ git push origin master
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (4/4), 428 bytes | 428.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
To /home/sasha/.git/
   f5cbc61..4bb43a3  master -> master
sasha@froge:~/code-user$
```

oraxterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

Рис.17

В конце добавим изменения в основной репозиторий

git push origin master

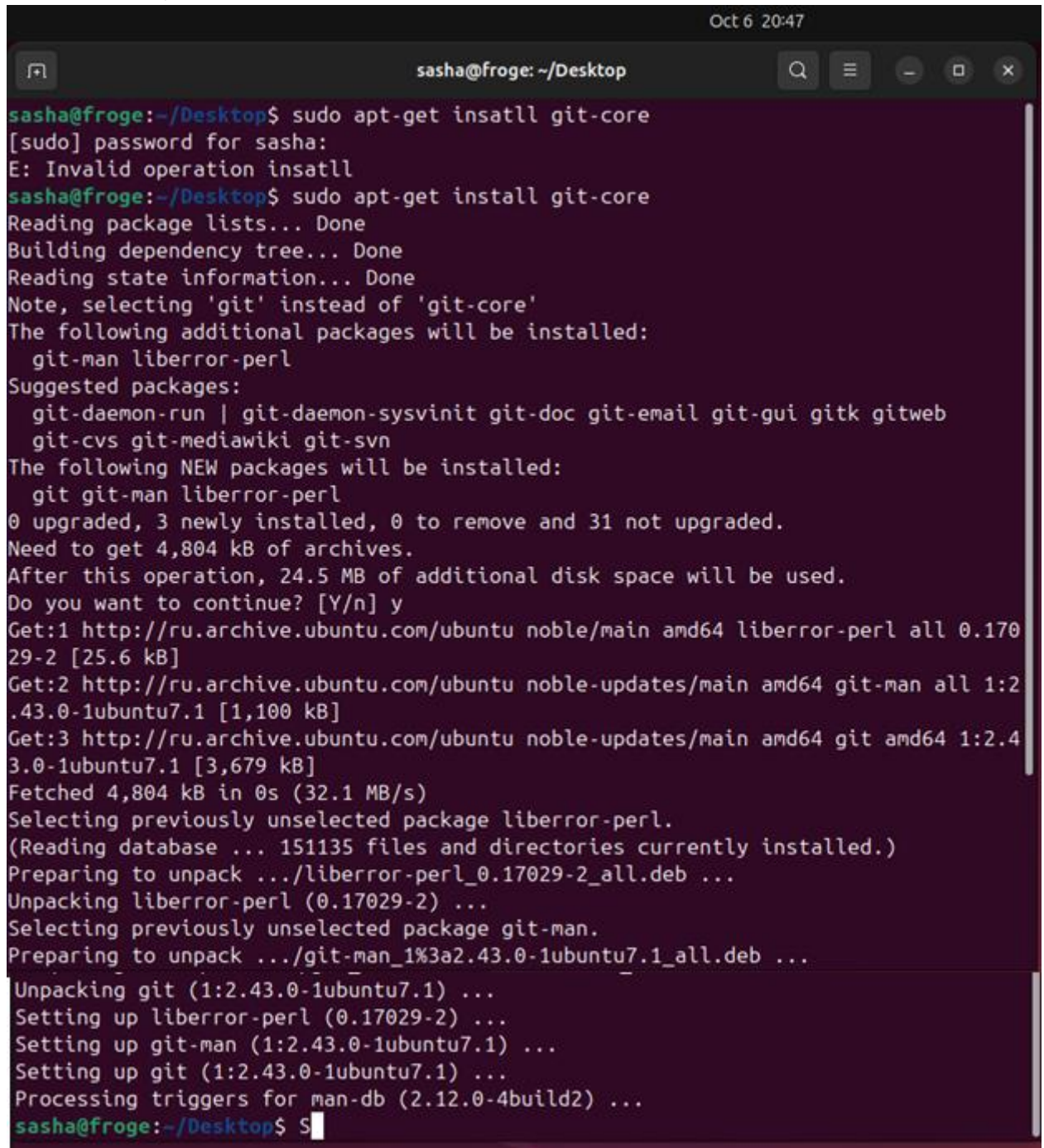
где

origin - последний опубликованный коммит на сервере

master - имя ветки

Варианты заданий

Для начала установим Git.



```
sasha@froge: ~/Desktop
sasha@froge:~/Desktop$ sudo apt-get install git-core
[sudo] password for sasha:
E: Invalid operation insatll
sasha@froge:~/Desktop$ sudo apt-get install git-core
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Note, selecting 'git' instead of 'git-core'
The following additional packages will be installed:
  git-man liberror-perl
Suggested packages:
  git-daemon-run | git-daemon-sysvinit git-doc git-email git-gui gitk gitweb
  git-cvs git-mediawiki git-svn
The following NEW packages will be installed:
  git git-man liberror-perl
0 upgraded, 3 newly installed, 0 to remove and 31 not upgraded.
Need to get 4,804 kB of archives.
After this operation, 24.5 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://ru.archive.ubuntu.com/ubuntu noble/main amd64 liberror-perl all 0.170
29-2 [25.6 kB]
Get:2 http://ru.archive.ubuntu.com/ubuntu noble-updates/main amd64 git-man all 1:2
.43.0-1ubuntu7.1 [1,100 kB]
Get:3 http://ru.archive.ubuntu.com/ubuntu noble-updates/main amd64 git amd64 1:2.4
3.0-1ubuntu7.1 [3,679 kB]
Fetched 4,804 kB in 0s (32.1 MB/s)
Selecting previously unselected package liberror-perl.
(Reading database ... 151135 files and directories currently installed.)
Preparing to unpack .../liberror-perl_0.17029-2_all.deb ...
Unpacking liberror-perl (0.17029-2) ...
Selecting previously unselected package git-man.
Preparing to unpack .../git-man_1%3a2.43.0-1ubuntu7.1_all.deb ...
Unpacking git (1:2.43.0-1ubuntu7.1) ...
Setting up liberror-perl (0.17029-2) ...
Setting up git-man (1:2.43.0-1ubuntu7.1) ...
Setting up git (1:2.43.0-1ubuntu7.1) ...
Processing triggers for man-db (2.12.0-4build2) ...
sasha@froge:~/Desktop$ S
```

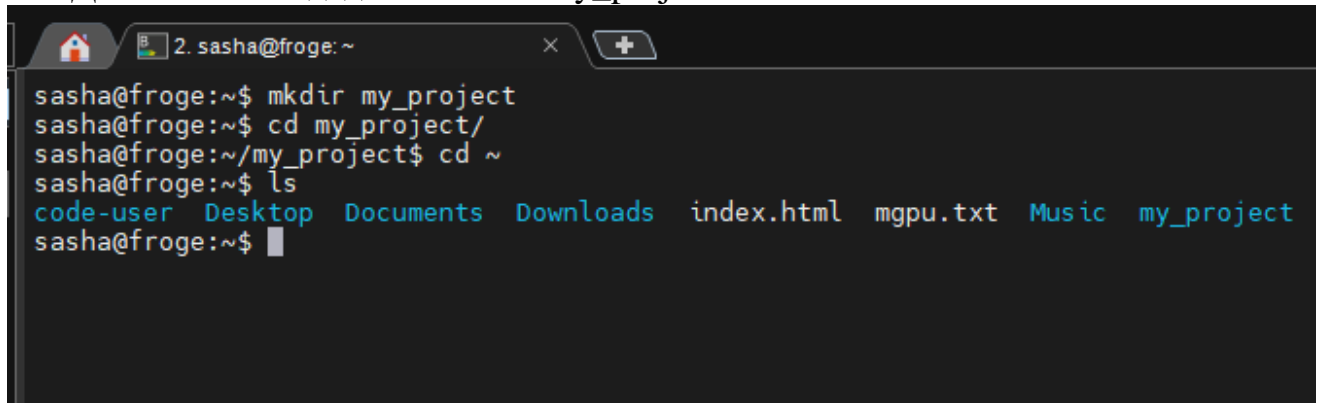
Рис.18

Для этого воспользуемся командой

Sudo apt-get install git-core

Задание 1. Создадим каталог, перенесем его в Git и создадим файлы настройки .gitignore и загрузим его в GitHub на Ubuntu 24:

Для начала создадим каталог my_project



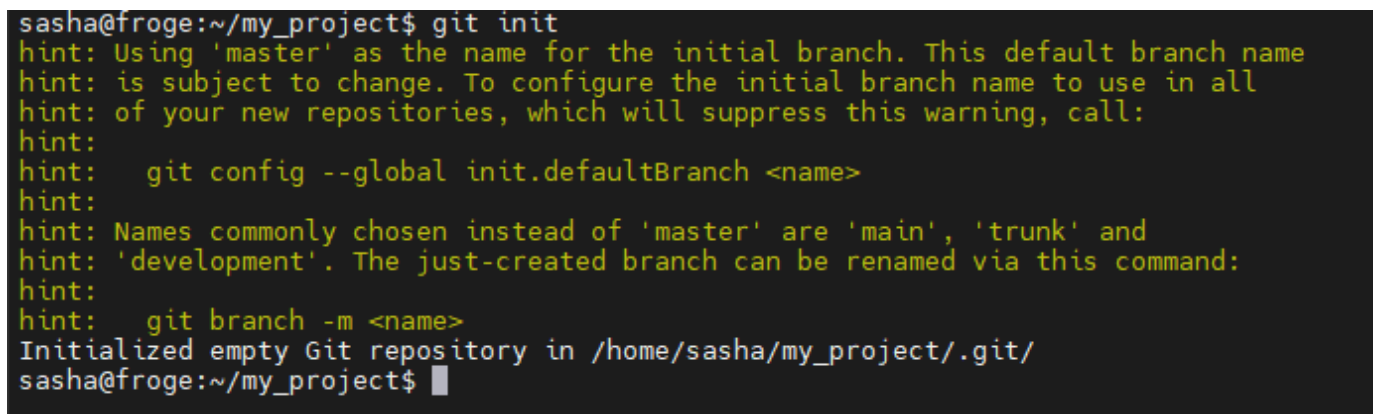
```
sasha@froge:~$ mkdir my_project
sasha@froge:~$ cd my_project/
sasha@froge:~/my_project$ cd ~
sasha@froge:~$ ls
code-user  Desktop  Documents  Downloads  index.html  mgpu.txt  Music  my_project
sasha@froge:~$
```

Рис.19

Воспользуемся командой

mkdir my_project, после чего перейдем в него и проверим его существование **cd my_project , ls**

Инициализируем Git репозиторий в директории my_project:



```
sasha@froge:~/my_project$ git init
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:   git config --global init.defaultBranch <name>
hint:
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this command:
hint:
hint:   git branch -m <name>
Initialized empty Git repository in /home/sasha/my_project/.git/
sasha@froge:~/my_project$
```

Рис.20

git init

Далее создадим файлы README.md и main.py:

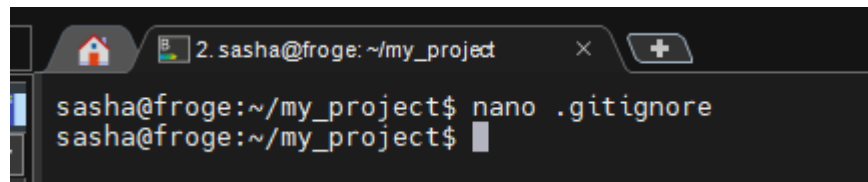
```
sasha@froge:~/my_project$ touch README.md
sasha@froge:~/my_project$ ls
README.md
sasha@froge:~/my_project$ touch main.py
sasha@froge:~/my_project$ ls
main.py README.md
sasha@froge:~/my_project$
```

Рис.21

touch README.md

touch main.py

Создадим .gitignore:

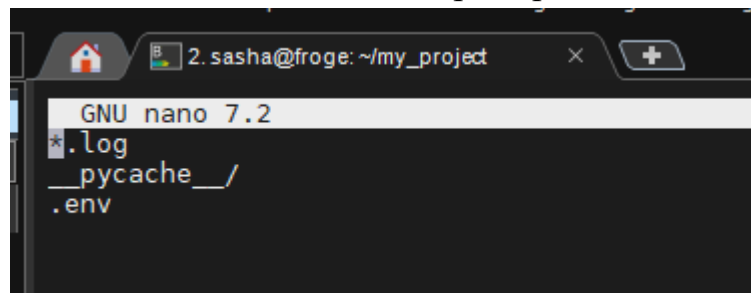


```
2. sasha@froge: ~/my_project
sasha@froge:~/my_project$ nano .gitignore
sasha@froge:~/my_project$
```

Рис.22

Командой **nano .gitignore**

Добавим в файл типичные исключения, например:



```
GNU nano 7.2
*.log
__pycache__/.env
```

Рис.23

***.log**

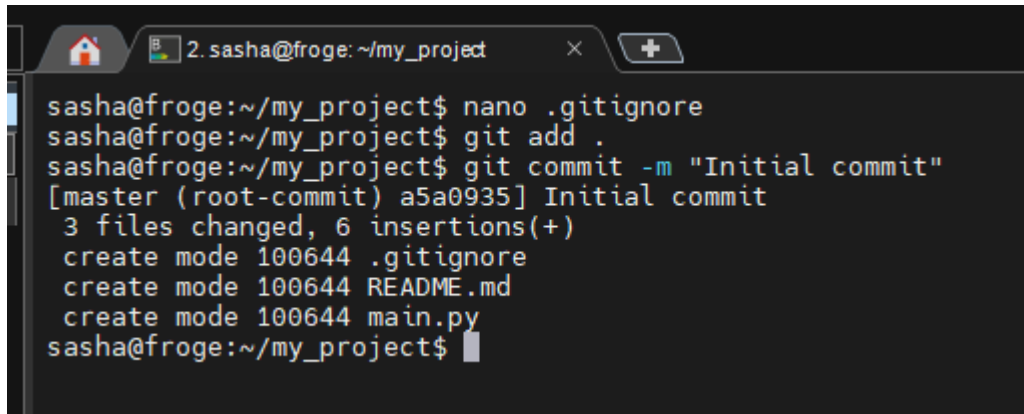
__pycache__/.env

.env

Добавим файлы в Git командой:

git add .

После чего сделаем первого коммита:



```
sasha@froge:~/my_project$ nano .gitignore
sasha@froge:~/my_project$ git add .
sasha@froge:~/my_project$ git commit -m "Initial commit"
[master (root-commit) a5a0935] Initial commit
3 files changed, 6 insertions(+)
create mode 100644 .gitignore
create mode 100644 README.md
create mode 100644 main.py
sasha@froge:~/my_project$
```

Рис.24

git commit -m "Initial commit"

После чего создадим репозиторий на GitHub. Для этого откроем GitHub в браузере, нажмем "+" в правом верхнем углу и выберем "New repository", назовем его "praktika-2" и настроим его.

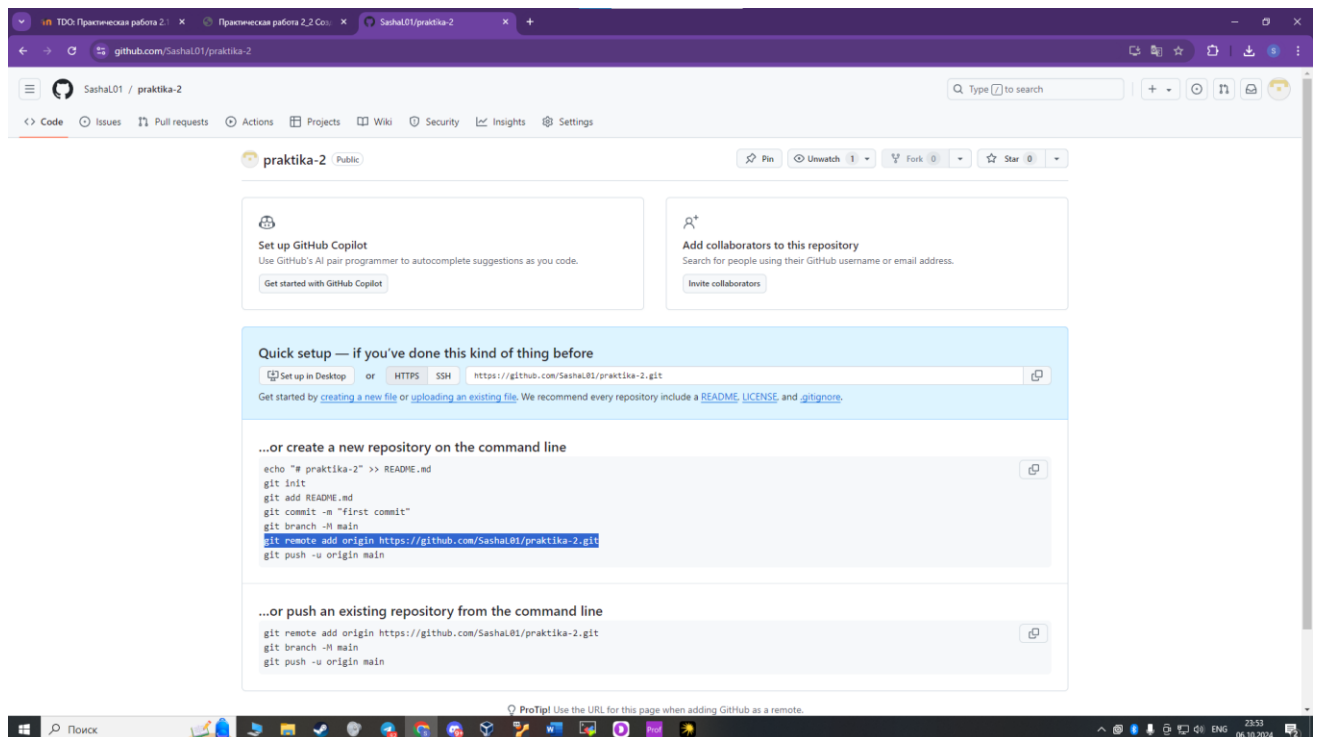


Рис.25

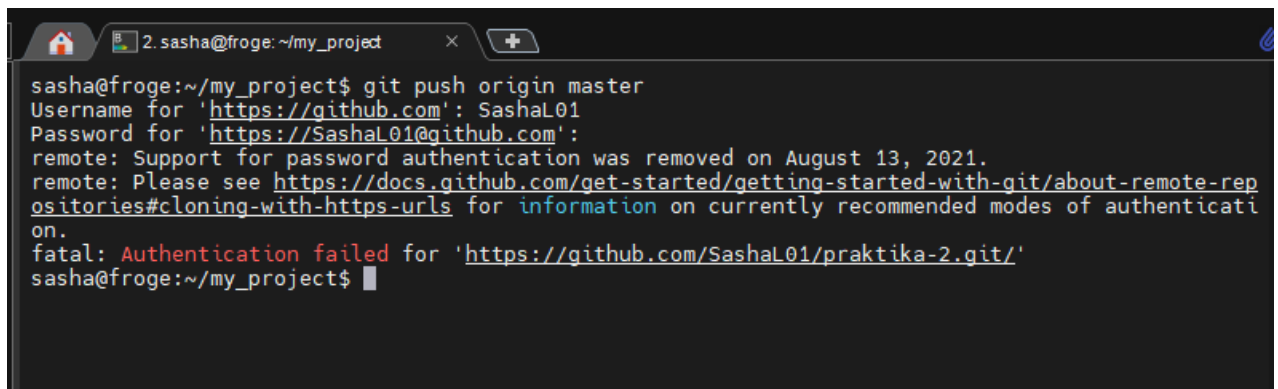
Свяжем локальный репозиторий с GitHub командой:

```
sasha@froge:~/my_project$ git remote add origin https://github.com/SashaL01/praktika-2.git  
sasha@froge:~/my_project$
```

Рис.26

git remote add origin https://github.com/SashaL01/praktika-2.git

Отправка изменений на GitHub:



```
2. sasha@froge: ~/my_project  
sasha@froge:~/my_project$ git push origin master  
Username for 'https://github.com': SashaL01  
Password for 'https://SashaL01@github.com':  
remote: Support for password authentication was removed on August 13, 2021.  
remote: Please see https://docs.github.com/get-started/getting-started-with-git/about-remote-repositories#cloning-with-https-urls for information on currently recommended modes of authentication.  
fatal: Authentication failed for 'https://github.com/SashaL01/praktika-2.git/'  
sasha@froge:~/my_project$
```

Рис.27

Отправим изменения на github командой

git push -u origin main

Но GitHub больше не поддерживает аутентификацию с помощью пароля для операций с Git.

Настройка SSH для GitHub.

1. Для начала проверим наличие существующих SSH-ключей:

ls -al ~/.ssh

```
sasha@froge:~/my_project$ ls -al ~/.ssh
total 24
drwx----- 2 sasha sasha 4096 Oct  7 00:29 .
drwxr-x--- 23 sasha sasha 4096 Oct  7 00:05 ..
-rw----- 1 sasha sasha   0 Oct  2 11:22 authorized_keys
-rw----- 1 sasha sasha 411 Oct  7 00:29 id_ed25519
-rw-r--r-- 1 sasha sasha  99 Oct  7 00:29 id_ed25519.pub
-rw----- 1 sasha sasha 978 Oct  2 12:18 known_hosts
-rw-r--r-- 1 sasha sasha 142 Oct  2 12:17 known_hosts.old
```

Рис.28

Создадим новый SSH-ключ:

```
2.sasha@froge: ~/my_project
-rw-r--r-- 1 sasha sasha 142 Oct  2 12:17 known_hosts.old
sasha@froge:~/my_project$ ssh-keygen -t ed25519 -C "frogedoge@mail.ru"
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/sasha/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/sasha/.ssh/id_ed25519
Your public key has been saved in /home/sasha/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:BFtxVm7rQsUp2CN6ZAToxAKaPkZ5MKPfttiXEL3uwG4 frogedoge@mail.ru
The key's randomart image is:
+--[ED25519 256]--+
|.= . .o+.o..|
|o.*+. = = o .|
|+o =. o * + *|
|o..... * . = .|
|+. + o S . .|
|. = + o . .|
|. = + . .|
|.E+ .|
|..|
+----[SHA256]-----+
sasha@froge:~/my_project$ ls -al ~/.ssh
```

Рис.29

ssh-keygen -t ed25519 -C “frogedoge@mail.ru” (укажем почту связанную с учетной записью github)

Запустим SSH-агент командой:

```
sasha@froge:~/my_project$ eval "$(ssh-agent -s)"
Agent pid 6997
sasha@froge:~/my_project$
```

Рис.30

```
eval "$(ssh-agent -s)"
```

Добавим SSH-ключ в ssh-agent:

```
sasha@froge:~/my_project$ ssh-add ~/.ssh/id_ed25519
Identity added: /home/sasha/.ssh/id_ed25519 (frogedoge@mail.ru)
sasha@froge:~/my_project$
```

Рис.31

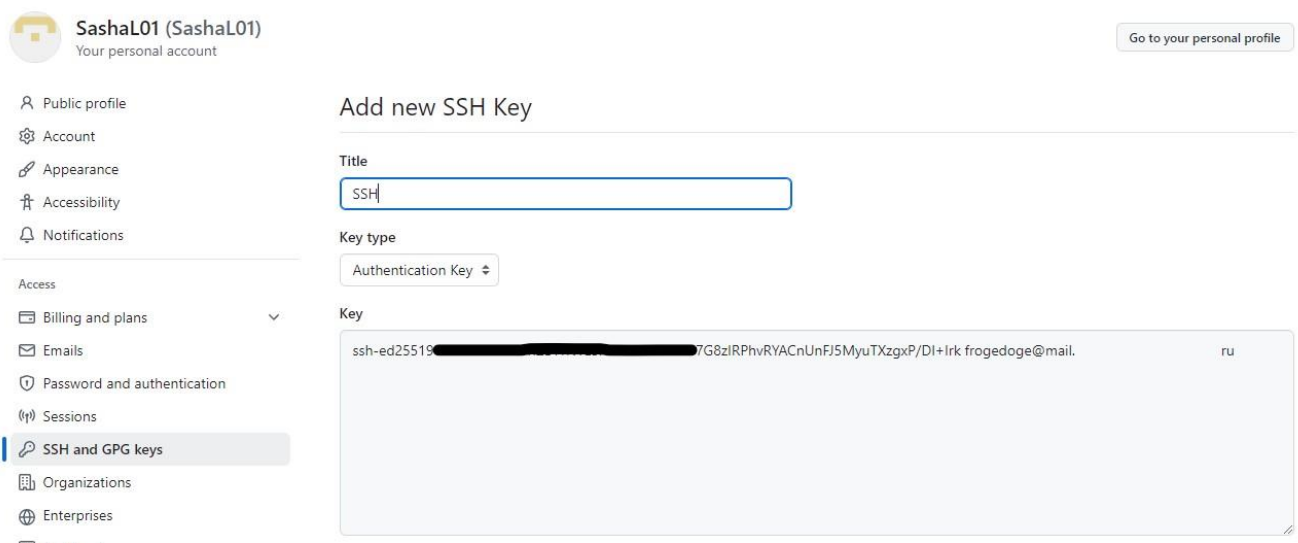
```
ssh-add ~/.ssh/id_ed25519
```

Скопируем публичный SSH-ключ в буфер обмена:

```
cat ~/.ssh/id_ed25519.pub
```

Перейдем на GitHub в профиле настройках, во вкладке "SSH and GPG keys", создадим новый SSH key

В поле "Key" вставим ключ.



SashaL01 (SashaL01)
Your personal account

Go to your personal profile

Public profile
Account
Appearance
Accessibility
Notifications

Access

Billing and plans
Emails
Password and authentication
Sessions
SSH and GPG keys
Organizations
Enterprises

Add new SSH Key

Title
SSH

Key type
Authentication Key

Key
ssh-ed25519 7G8zIRPhvRYACnUnFJ5MyuTXzgxP/DI+lrk frogedoge@mail.ru

Рис.32

Нажмем "Add SSH key" после чего, появится информация о новом добавленном ключе.



Рис.33

Проверим подключение с github при помощи:
ssh -T git@github.com

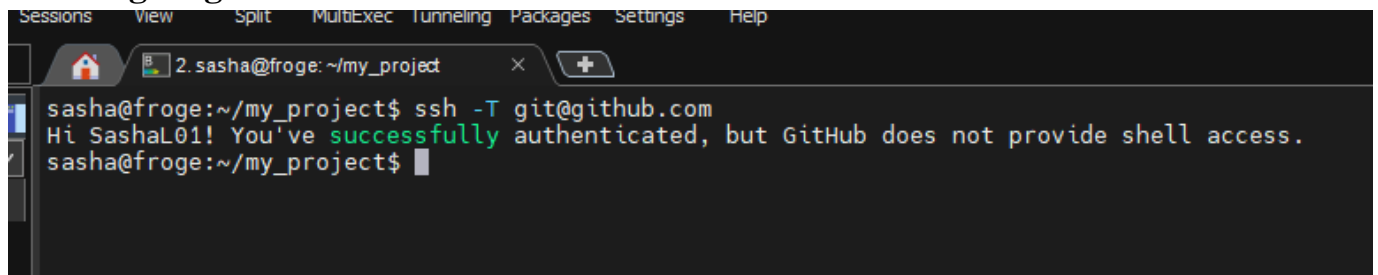


Рис.34

Необходимо убедиться, что удаленный репозиторий настроен с использованием SSH.

Git remote -v

Показывает, что подключение настроено по https

```
sasha@froge:~/my_project$ git remote -v
origin https://github.com/SashaL01/praktika-2.git (fetch)
origin https://github.com/SashaL01/praktika-2.git (push)
sasha@froge:~/my_project$ git remote rm origin
sasha@froge:~/my_project$
```

Рис.35

Нужно удалить его командой

Git remote rm origin

После чего прописать

```
git remote add origin git@github.com:SashaL01/praktika-2.git
```

Отправим изменения на GitHub:

```
git push origin master
```

```
sasha@froge:~/my_project$ git push origin master
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (4/4), 290 bytes | 290.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:SashaL01/praktika-2.git
 * [new branch]      master -> master
sasha@froge:~/my_project$
```

Рис.36

Если связь настроено правильно, то не потребует ввод пароля. GitHub будет использовать ваш SSH-ключ для аутентификации.

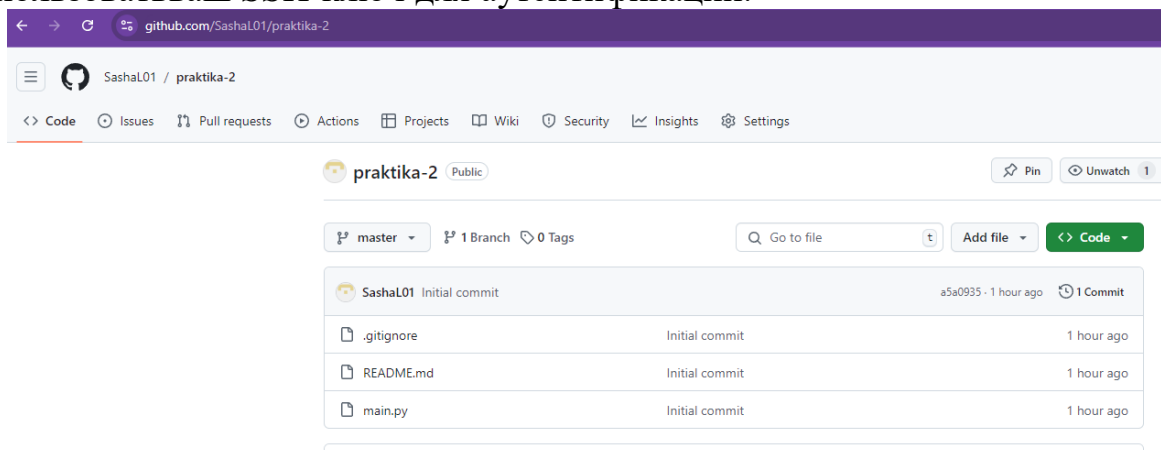


Рис.37

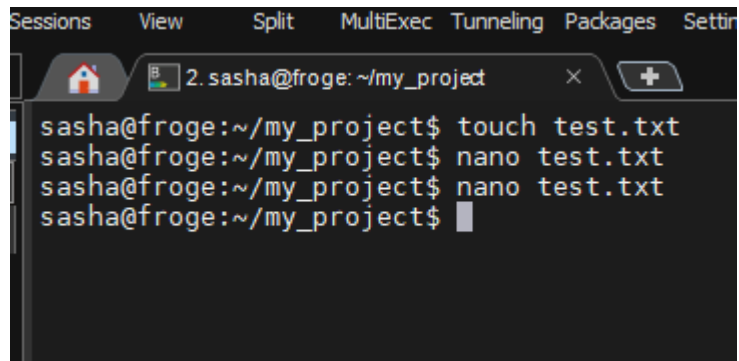
Задание 2.

Создадим новый файл и отправим изменения на удаленный репозиторий в GitHub.

Перейдите в директорию вашего локального Git-репозитория:

```
Cd my_project
```

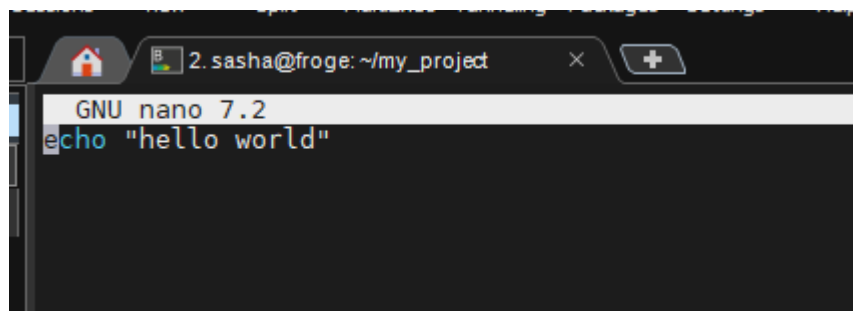
Создадим новый файл `touch test.txt` и добавим в файл `nano test.txt` текст “hello world”

A terminal window titled '2. sasha@froge: ~/my_project' showing a sequence of commands: 'touch test.txt', 'nano test.txt', 'nano test.txt', and a prompt 'sasha@froge:~/my_project\$' with a cursor.

```
sasha@froge:~/my_project$ touch test.txt
sasha@froge:~/my_project$ nano test.txt
sasha@froge:~/my_project$ nano test.txt
sasha@froge:~/my_project$
```

Рис.38

Создаем текстовый файл

A terminal window titled '2. sasha@froge: ~/my_project' showing the nano editor interface. The title bar says 'GNU nano 7.2'. The first line of the file contains 'echo "hello world"'.

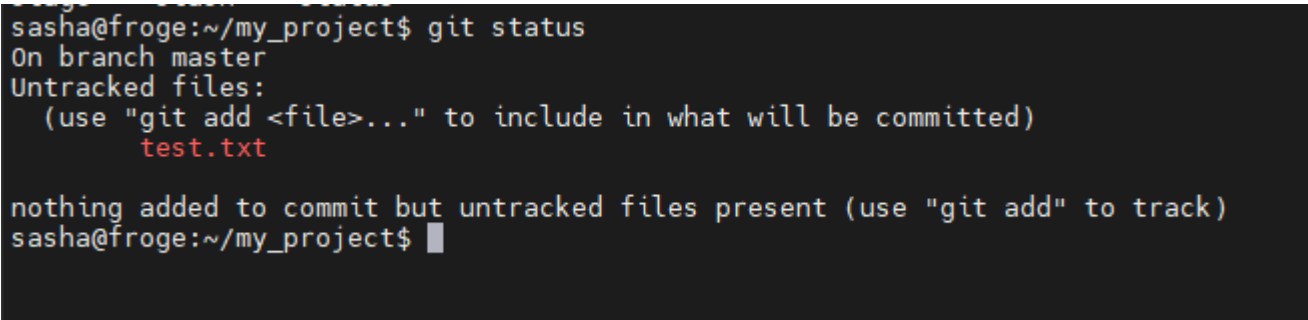
```
GNU nano 7.2
echo "hello world"
```

Рис.39

Добавляем текст

Проверим статус репозитория командой:

git status

A terminal window showing the output of the 'git status' command. It indicates that the file 'test.txt' is untracked and not yet added to the commit.

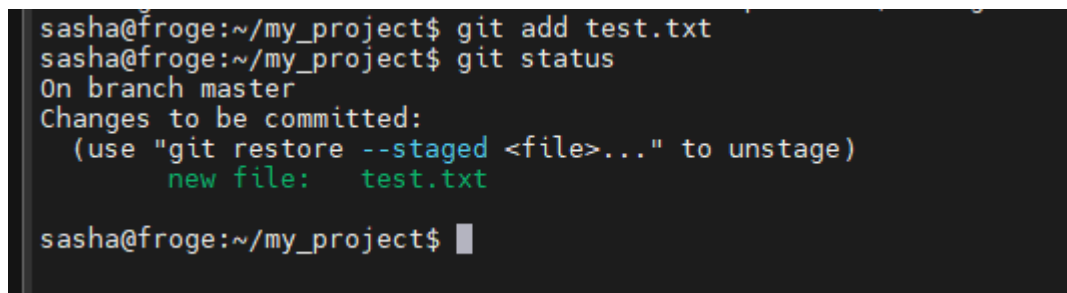
```
sasha@froge:~/my_project$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    test.txt

nothing added to commit but untracked files present (use "git add" to track)
sasha@froge:~/my_project$
```

Рис.40

Добавим файл в индекс Git:

git add test.txt

A terminal window showing the execution of 'git add test.txt' followed by 'git status'. The output shows that 'test.txt' is now staged for commit.

```
sasha@froge:~/my_project$ git add test.txt
sasha@froge:~/my_project$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   test.txt

sasha@froge:~/my_project$
```

Рис.41

Сделаем коммит:

`git commit -m "Добавлен новый файл test.txt"`

```
>
sasha@froge:~/my_project$ git commit -m "добавлен новый файл test.txt"
[master 1dde661] добавлен новый файл test.txt
1 file changed, 1 insertion(+)
create mode 100644 test.txt
sasha@froge:~/my_project$
```

Рис.42

Отправьте изменения на GitHub:

```
sasha@froge:~/my_project$ git push origin master
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 323 bytes | 323.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:SashaL01/praktika-2.git
a5a0935..1dde661 master -> master
sasha@froge:~/my_project$
```

Рис.43

`git push origin master`

The screenshot shows the GitHub interface for a repository named 'praktika-2' (Public). At the top, there are buttons for 'Pin' and 'Unwatch'. Below the repository name, there are tabs for 'master', '1 Branch', and '0 Tags'. A search bar 'Go to file' and buttons 'Add file' and 'Code' are also visible. The main content area displays a list of commits. The most recent commit is by 'SashaL01' with the message 'добавлен новый файл test.txt', commit hash '1dde661', and 'now' time. It shows 2 commits in total. Below this, a table lists the files included in the commit:

| File | Commit Message | Time |
|------------|------------------------------|------------|
| .gitignore | Initial commit | 1 hour ago |
| README.md | Initial commit | 1 hour ago |
| main.py | Initial commit | 1 hour ago |
| test.txt | добавлен новый файл test.txt | now |

Рис.44

Проверим добавление файлов на сайте github.

Индивидуальные задания

Вариант 12. Продвинутое использование индекса:

Создайте репозиторий с несколькими файлами. Внесите изменения в разные части каждого файла. Используйте `git add -p` для интерактивного добавления только определенных изменений в индекс. Создайте коммит из этих выборочных изменений.

Для начала создадим директорию для нового проекта:

```
sasha@froge:~$ mkdir my_new_project  
sasha@froge:~$ cd my_new_project/
```

Рис.45

Mkdir my_new_project

И перейдем в неё

Cd my_new_project

Инициализируем новый Git-репозиторий:

```
sasha@froge:~/my_new_project$ git init  
hint: Using 'master' as the name for the initial branch. This default branch name  
hint: is subject to change. To configure the initial branch name to use in all  
hint: of your new repositories, which will suppress this warning, call:  
hint:  
hint:   git config --global init.defaultBranch <name>  
hint:  
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and  
hint: 'development'. The just-created branch can be renamed via this command:  
hint:  
hint:   git branch -m <name>  
Initialized empty Git repository in /home/sasha/my_new_project/.git/  
sasha@froge:~/my_new_project$
```

mobaxterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

Рис.46

Git init

Создадим несколько файлов

```
sasha@froge:~/my_new_project$ touch test.text
sasha@froge:~/my_new_project$ ls
test.text
sasha@froge:~/my_new_project$ touch test.py
sasha@froge:~/my_new_project$ ls
test.py test.text
sasha@froge:~/my_new_project$ touch test.html
sasha@froge:~/my_new_project$ ls
test.html test.py test.text
sasha@froge:~/my_new_project$
```

Рис.47

Touch test.text

Touch test.py

Touch test.html

Добавим в каждый файл немного текста:

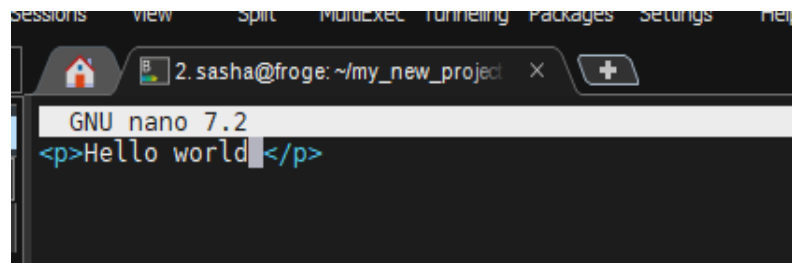


Рис.48

Перейдем в каждый и добавим текст

Nano test.html

<p>hello world</p>

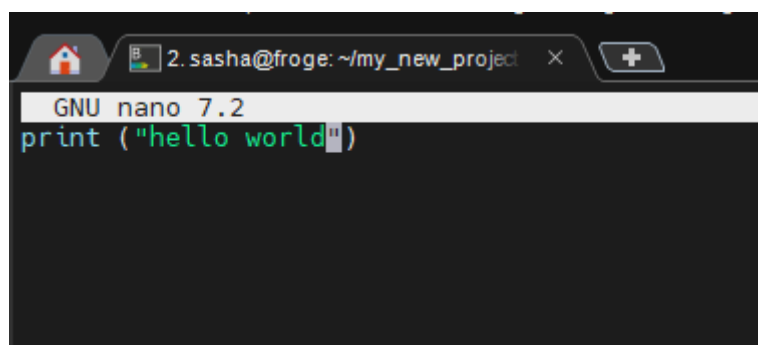


Рис.49

Nano test.py

print("hello world")

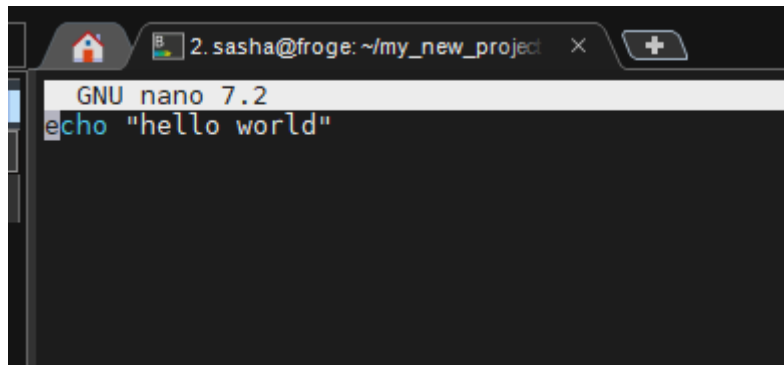


Рис.50

Nano test.text

Echo “hello world”

Добавим эти файлы в репозиторий и сделайте первый коммит:

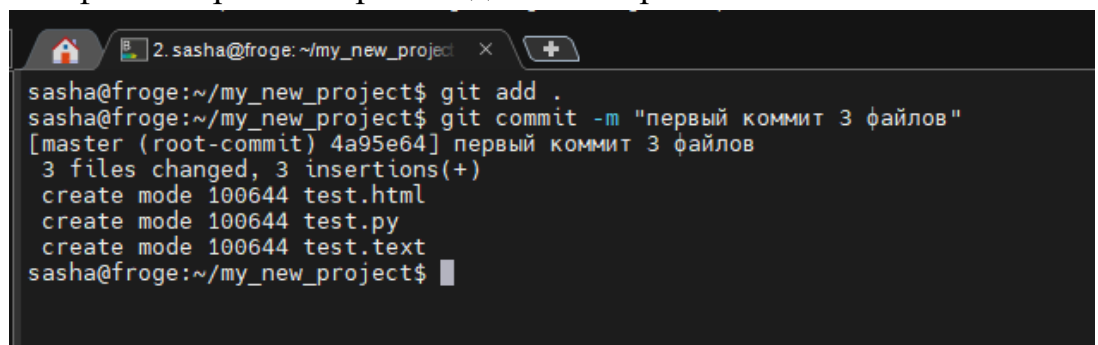


Рис.51

Git add .

Git commit -m “первый коммит 3 файлов”

Внесем небольшие изменения в файлы

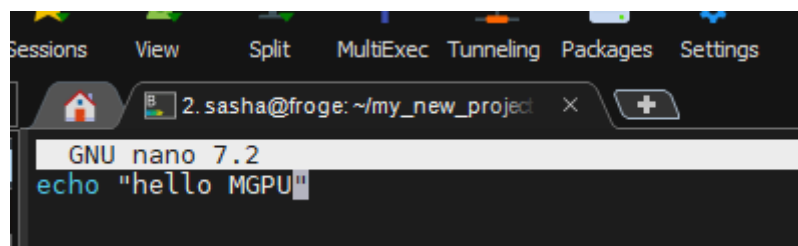


Рис.52

Nano test.text

Echo “hello MGPU”

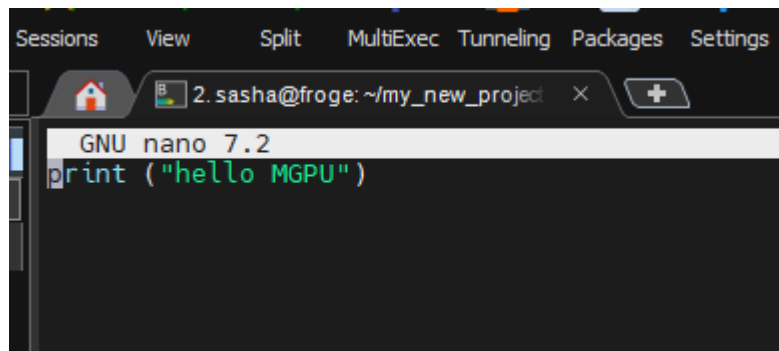


Рис.53

Nano test.py
print ("hello MGPU")

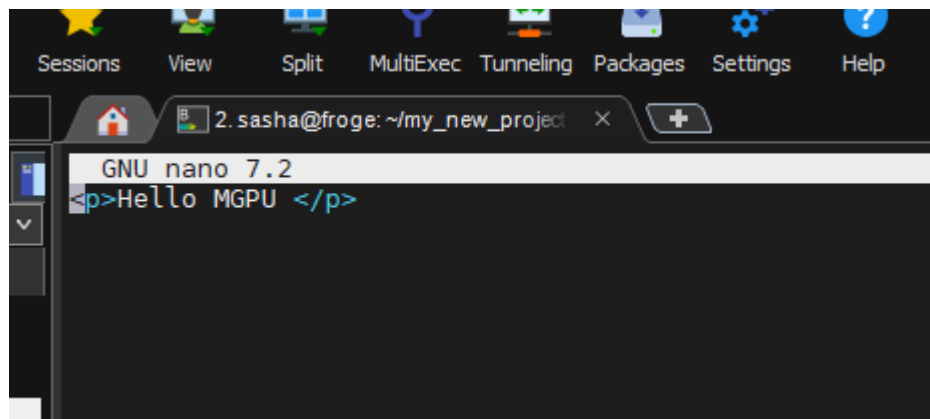


Рис.54

Nano test.html
<p>hello MGPU</p>

Проверим статус изменений:

```
sasha@froge:~/my_new_project$ 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:   test.html
        modified:   test.py
        modified:   test.text

no changes added to commit (use "git add" and/or "git commit -a")
sasha@froge:~/my_new_project$
```

Рис.55

Git status

Выберем изменения для добавления в индекс:

После запуска **git add -p** Git начнёт показывать изменения по частям и предложит несколько опций:

- **y**: добавить эту часть изменений в индекс.
- **n**: пропустить эту часть изменений.
- **s**: разбить эту часть на ещё более мелкие части.
- **e**: вручную отредактировать выбранные изменения.
- **q**: выйти без добавления изменений.

```
sasha@froge:~/my_new_project$ git add -p
diff --git a/test.html b/test.html
index 36761d0..990eadc 100644
--- a/test.html
+++ b/test.html
@@ -1,1 @@
-<p>Hello world </p>
+<p>Hello MGPU </p>
(1/1) Stage this hunk [y,n,q,a,d,e,?]? █
```

git add -p

Добавим изменения для test.py и test.text , а test.html оставим без изменений

Рис.56

```
sasha@froge:~/my_new_project$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   test.py
        modified:   test.text

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:   test.html

sasha@froge:~/my_new_project$ █
```

Рис.57

Проверим статус изменений

Git status

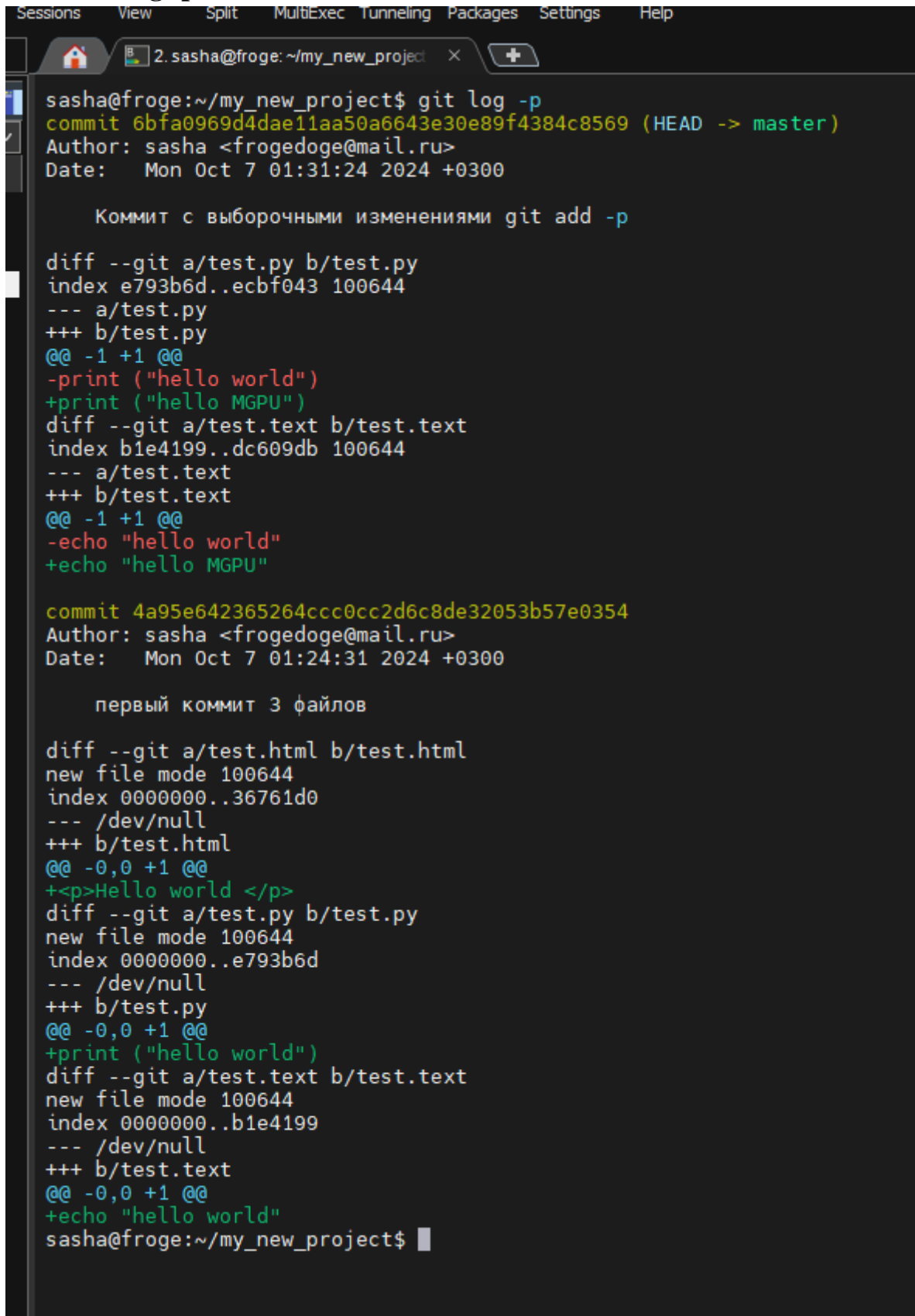
Создадим коммит из выбранных изменений

```
sasha@froge:~/my_new_project$ git commit -m "Коммит с выборочными изменениями git add -p"
[master 6bfa096] Коммит с выборочными изменениями git add -p
2 files changed, 2 insertions(+), 2 deletions(-)
sasha@froge:~/my_new_project$ █
```

Рис.58

Проверим результат командой

Git log -p



```
Sessions View Split MultiExec Tunneling Packages Settings Help
2. sasha@froge: ~/my_new_project X +

sasha@froge:~/my_new_project$ git log -p
commit 6bfa0969d4dae11aa50a6643e30e89f4384c8569 (HEAD -> master)
Author: sasha <frogedoge@mail.ru>
Date: Mon Oct 7 01:31:24 2024 +0300

    Коммит с выборочными изменениями git add -p

diff --git a/test.py b/test.py
index e793b6d..ecbf043 100644
--- a/test.py
+++ b/test.py
@@ -1,1 @@
-print ("hello world")
+print ("hello MGPU")
diff --git a/test.text b/test.text
index b1e4199..dc609db 100644
--- a/test.text
+++ b/test.text
@@ -1,1 @@
-echo "hello world"
+echo "hello MGPU"

commit 4a95e642365264ccc0cc2d6c8de32053b57e0354
Author: sasha <frogedoge@mail.ru>
Date: Mon Oct 7 01:24:31 2024 +0300

    первый коммит 3 файлов

diff --git a/test.html b/test.html
new file mode 100644
index 0000000..36761d0
--- /dev/null
+++ b/test.html
@@ -0,0 +1 @@
+<p>Hello world </p>
diff --git a/test.py b/test.py
new file mode 100644
index 0000000..e793b6d
--- /dev/null
+++ b/test.py
@@ -0,0 +1 @@
+print ("hello world")
diff --git a/test.text b/test.text
new file mode 100644
index 0000000..b1e4199
--- /dev/null
+++ b/test.text
@@ -0,0 +1 @@
+echo "hello world"
sasha@froge:~/my_new_project$
```

Рис.59

Заключение

В ходе выполнения практической работы, были получены знания о том как делать коммиты, инициализировать и клонировать репозитории, использование Git для управления версиями проекта, подключение локального с удаленным репозиторием посредством обмена ssh ключами.