

Лабораторная работа №2

Отчёт

Ермишина Мария Кирилловна

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	10
5	Выводы	21
6	Контрольные вопросы	22
	Список литературы	24

Список иллюстраций

4.1	Установка git	10
4.2	Установка gh	10
4.3	Базовая настройка git	11
4.4	Создание SSH	12
4.5	Создание PGP	13
4.6	Вывод ключей	14
4.7	Копирование PGP	14
4.8	Ключ на Git	15
4.9	Наличие ключа	16
4.10	Настройка авто-подписей	16
4.11	Терминал	17
4.12	Подтверждение на сайте	17
4.13	Создание каталогов	18
4.14	git clone	18
4.15	Наличие шаблона на сайте	19
4.16	Удаление файлов	19
4.17	Создание каталогов	20
4.18	Отправка файлов на сервер	20

Список таблиц

1 Цель работы

Целью данной лабораторной работы являются: изучение идеологии и применение средств контроля версий; освоение умения по работе с git. [1]

2 Задание

- Создать базовую конфигурацию для работы с git.
- Создать ключ SSH.
- Создать ключ PGP.
- Настроить подписи git.
- Зарегистрироваться на Github.
- Создать локальный каталог для выполнения заданий по предмету.

3 Теоретическое введение

Основ-	
ные	
коман-	
ды git	Описание каталога

git	Создание основного дерева репозитория
init	
git	Получение обновлений (изменений) текущего дерева из центрального
pull	репозитория
git	Отправка всех произведённых изменений локального дерева в
push	центральный репозиторий
git	Просмотр списка изменённых файлов в текущей директории
status	
git	Просмотр текущих изменений
diff	
git add	добавить все изменённые и/или созданные файлы и/или каталоги
.	
git add	добавить конкретные изменённые и/или созданные файлы и/или
имена_файлов	каталоги
git rm	удалить файл и/или каталог из индекса репозитория (при этом файл
имена_файлов	и/или каталог остаётся в локальной директории)

Основ-	
ные	
коман-	
ды git	Описание каталога
git	сохранить все добавленные изменения и все изменённые файлы
commit	
-am	
'Описание	
коммита '	
git	сохранить добавленные изменения с внесением комментария через
commit	встроенный редактор
git	создание новой ветки, базирующейся на текущей
checkout	
-b	
имя_ветки	
git	переключение на некоторую ветку
checkout	переключение на некоторую ветку
имя_ветки	(при переключении на ветку, которой ещё нет в локальном репозитории, она будет создана и связана с удалённой)
git	отправка изменений конкретной ветки в центральный репозиторий
push	
origin	
имя_ветки	
git	слияние ветки с текущим деревом
merge	
--no-ff	
имя_ветки	

Основ-	
ные	
коман-	
ды git	Описание каталога

git	удаление локальной уже слитой с основным деревом ветки
branch	
-d	
имя_ветки	
git	принудительное удаление локальной ветки
branch	
-D	
имя_ветки	
git	удаление ветки с центрального репозитория
push	
origin	
:имя_веткиgit	
push	
origin	
:имя_ветки	

4 Выполнение лабораторной работы

1. Установка git и gh. Их мы устанавливаем с помощью команд:

- `dnf install git` (рис. 4.1)
- `dnf install gh` (рис. 4.2)

```
[root@ermimash ~]# dnf install git
Обновление и загрузка репозитория:
Репозитории загружены.
Пакет "git-2.48.1-1.fc41.x86_64" уже установлен.

Нечего делать.
[root@ermimash ~]#
```

Рис. 4.1: Установка git

```
[root@ermimash ~]# dnf install gh
Обновление и загрузка репозитория:
Репозитории загружены.
Пакет
Установка:
  gh
  Арх.   Версия
  x86_64 2.65.0-1.fc41
  Репозиторий
  updates
  Размер
  42.6 MiB

Сводка транзакции:
  Установка:      1 пакета

Общий размер входящих пакетов составляет 10 MiB. Необходимо загрузить 10 MiB.
После этой операции будут использоваться дополнительные 43 MiB (установка 43 MiB, удаление 0 B).
Is this ok [y/N]: y
[1/1] gh-0:2.65.0-1.fc41.x86_64 100% | 9.3 MiB/s | 10.3 MiB | 00m01s
-----
[1/1] Total 100% | 7.2 MiB/s | 10.3 MiB | 00m01s
Выполнение транзакции
[1/3] Проверить файлы пак100% | 29.0 B/s | 1.0 B | 00m00s
[2/3] Подготовить транзак100% | 2.0 B/s | 1.0 B | 00m00s
[3/3] Установка gh-0:2.65.0-1.fc41 100% | 26.6 MiB/s | 42.7 MiB | 00m02s
Завершено!
[root@ermimash ~]#
```

Рис. 4.2: Установка gh

2. Базовая настройка git. (рис. 4.3)
3. Зададим имя и email владельца репозитория с помощью команд: `git config --global user.name "Name Surname"` `git config --global user.email "work@mail"`
4. Настроим utf-8 в выводе сообщений git: `git config --global core.quotepath false`
5. Зададим имя начальной ветки (будем называть её master): `git config --global init.defaultBranch master`
6. Параметры `autocrlf` и `safecrlf`: `git config --global core.safecrlf warn`

```
[root@ermimash ~]# git config --global user.name "ErmiMash"
[root@ermimash ~]# git config --global user.email "1132230166@pfur.ru"
[root@ermimash ~]# git config --global core.quotepath false
[root@ermimash ~]# git config --global init.defaultBranch master
[root@ermimash ~]# git config --global core.autocrlf input
[root@ermimash ~]# git config --global core.safecrlf warn
```

Рис. 4.3: Базовая настройка git

3. Создаём ключи SSH. (рис. 4.4) Их мы создаём с помощью команд:
 - `ssh-keygen -t rsa -b 4096` (создание по алгоритму *rsa* с размером 4096 бит)
 - `ssh-keygen -t ed25519` (создание по алгоритму *ed25519*)

```

[root@ermimash ~]# ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase for "/root/.ssh/id_rsa" (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa
Your public key has been saved in /root/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:3o1jJMXjkyUR/DgtKUyle0jnhQ2ZLBN4wkRCtTYan0U root@ermimash
The key's randomart image is:
+---[RSA 4096]-----+
|  .o=+.E==+ |
|  .+o=O==+ |
|  . +=+oB=+ |
|  = oB=*o |
|  . o S.Bo |
|  . = + |
|  . = . |
|  . . |
|  |
+-----[SHA256]-----+
[root@ermimash ~]# ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/root/.ssh/id_ed25519):
Enter passphrase for "/root/.ssh/id_ed25519" (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_ed25519
Your public key has been saved in /root/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:PRYhaMX9ZE1lofL9UR6QoKMY9Jgw0naiSpzsJNzBMXs root@ermimash
The key's randomart image is:
+--[ED25519 256]--+
| .O=.. +O..OO+=. |
| .**.* ..O.++. |
| + =.+E . OO+. .. |
| .0 .. o ...+. .o |
| * ..S + . ... |
| .. . . .. |
|  |
|  |
+-----[SHA256]-----+
[root@ermimash ~]#

```

Рис. 4.4: Создание SSH

4. Создаём ключ PGP. (рис. 4.5) Используем команду для генерации ключа:

- `gpg --full-generate-key`

Данный ключ мы создаём со следующими опциями: 1. тип RSA and RSA 2. размер 4096 3. 0 (срок действия не истекает никогда)

Также указываем свои данные (Имя, Почту (которая соответствует почте на GitHub), комментарий), которые будут храниться в ключе.

```

[root@ermimash ~]# gpg --full-generate-key
gpg (GnuPG) 2.4.5; Copyright (C) 2024 g10 Code GmbH
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

gpg: создан каталог '/root/.gnupg'
Выберите тип ключа:
  (1) RSA and RSA
  (2) DSA and Elgamal
  (3) DSA (sign only)
  (4) RSA (sign only)
  (9) ECC (sign and encrypt) *default*
 (10) ECC (только для подписи)
 (14) Existing key from card
Ваш выбор? 1
длина ключей RSA может быть от 1024 до 4096.
Какой размер ключа Вам необходим? (3072) 4096
Запрошенный размер ключа - 4096 бит
Выберите срок действия ключа.
  0 = не ограничен
  <n> = срок действия ключа - n дней
  <n>w = срок действия ключа - n недель
  <n>m = срок действия ключа - n месяцев
  <n>y = срок действия ключа - n лет
Срок действия ключа? (0) 0
Срок действия ключа не ограничен
Все верно? (y/N) y

GnuPG должен составить идентификатор пользователя для идентификации ключа.

Ваше полное имя: Mariya Ermishina
Адрес электронной почты: 1132230166@pfur.ru
Примечание:
Вы выбрали следующий идентификатор пользователя:
  "Mariya Ermishina <1132230166@pfur.ru>"

Сменить (N)Имя, (C)Примечание, (E)Адрес; (O)Принять/(Q)Выход?

```

Рис. 4.5: Создание PGP

5. Настройка GitHub Данный пункт я пропустила, так как уже имею аккаунт на GitHub, который я создавала в 1 семестре учёбы.
6. Добавление PGP ключа в GitHub.
7. Выводим список ключей и копируем отпечаток приватного ключа с помощью следующей команды: `gpg --list-secret-keys --keyid-format LONG` (рис. 4.6)

```
[root@ermimash ~]# gpg --list-secret-keys --keyid-format LONG
gpg: проверка таблицы доверия
gpg: marginals needed: 3 completes needed: 1 trust model: pgp
gpg: глубина: 0 достоверных: 1 подписанных: 0 доверие: 0-, 0q, 0n, 0m, 0f, 1u
[keyboxd]
-----
sec   rsa4096/6EBEE46B3F690BCB 2025-03-04 [SC]
      42DAB1B9E0D8B8C6525BDD4C6EBEE46B3F690BCB
uid       [ абсолютно ] Maria Ermishina <1132230166@pfur.ru>
ssb   rsa4096/8675D038B5392509 2025-03-04 [E]

[root@ermimash ~]#
```

Рис. 4.6: Вывод ключей

2. Копируем сгенерированный ключ PGP в буфер обмена (с помощью команды получаем полный код, а далее вручную его копируем): `gpg --armor --export хclip -sel clip` (рис. 4.7)

```
[root@ermimash ~]# gpg --armor --export 1132230166@pfur.ru хclip -sel clip
-----BEGIN PGP PUBLIC KEY BLOCK-----

mQINBGfHw1cBEAC4GU0ChLfDRYGU0NhoWRa6p76eFoQdbNGZcrDGxw45d+xAAdvG
ncnUxnyXZ84CAUyGJRS+Uz9hfKyGsrBCJVA7AOZnse3KbrdiEH35CqvNBjyadDVP
razsygpcIRqFBkMP317bWahTr8ckOKGdX1+CwMJJJo97wtech24ryVyK4liW6bg
wfe2gvStG5sONHXVzNh57zY6uvNNpRHFcVOnPfzAzeVwjG+8wJz7m9F70fA8kgV
hgJcwfcA/w5cp/EXMHqoHQvHFpZoRJDIFzp0aSCuTspVKWly2eswLpLkzqz851
uVALLpIc/E2vCP5AH9yk5FFvMCULGDk1xf12jI/VXzyhuikhmxe15Qzv0eYL/dJJ
8F3oR0b3cKpINXI96gfdI3dam8MI9EMDy6XB0SSR5kSB2ZcgR+XDyL0nafSunbb4
1oyMHYFKiCf++zMA92RB6GpYMA/RaDUhw1uIWiSuejFR1bxx3hMFeyWvEt9BD8Yk
Xxz72HTqtM4vPH71/KWmEbX52Mqo1QX4cbimecst0v5c2WMyC3b1521CqK5305k
2be0zkQhJZTWLtlM9a7Qt29B2pAjHiSdkiSzw313cHiGGrwDTByyMR9HZSWVW95F
tjH8j1SiUQWtS7kBSbqABFEcJM51nnXF21fjERb6UmpariGcqy9oRNLOwARAQAB
tCRNYXJpYSBFCm1pc2hpbmEgPDEzMzIyMzAxNjZAcGZ1c15ydT6JAIEEEwEIAQsw
IQR2rG54Ni4x1Jb3UxuvuRrP2kLywUCZ8daJwIbAwULCQgHAgIiAgYVCgkICwIE
FgIDAQIEBwIXgAAKCRBuuvuRrP2kLy3ucD/90r3cgotI/YNwhWY2FV184vSDHdpD/
VAUiDy8W0urGsOUFnIva/J8xK8QRckmt9BW1jvQmmvTyG6NwsBRQHbkrfKu/+C
zRURSjCAtoiinugp0RIEfRjjhhyWfj1kp3B5Puatt7hVKKRNIFBzUu7cGN1t/qid
YeQPLabCfpuxnZjD13nTW+9rvJsdANXYc/JmJW7h7xRuvMAG+KQGOW/foHW7UpLo
XoyAAE83b/BsSdsC7QFVIQhUcYAp4UQ0qy7VaRyx14ankP5p5f4d1Wg0bCeu3USq
ZunJ5H0erEHC6vdM2/iC6n3kFEATSWJQzph9WCZLNbVyokQeHuDgFYILwWEl1syZ
hG+1ZG4JpxB212IXrQmdOUYh0XcBkkFwckUkSGVnpehoko2zLjNXMghSoj8CQ7bR
1hH8zC6K51c0nuQzdZ+NbDe9x8b34fMdmUkEBwaHuR8sVp6CqezP/ViWH+HJQP3M
NjSV187UICyzkWnLzzEoyaZPvJ1H8Ny/a5nTo/a9Wk6974qR0z7/FHFfJ3f06Kxm
5XdoN6iRTGxa4ounqGKP2X0jMfr05wH6e5GKMw8ssUxRu52JIBUGfC4kbHfAcoB
```

Рис. 4.7: Копирование PGP

Вставляем наш ключ на сайте GitHub и даём ему название. (рис. 4.8) Проверяем наличие ключа, обновив страницу. (рис. 4.9)

<> Developer settings

Add new GPG key

Title

Key

```
uvR10Q0nTCS0T4  
BAoQD0Z8Xq44rFYFjULkOlCui4DEZsg/DV9xyQJhjy/AOF9/  
TydQHOYddbGVf/xG  
Z5vr6ZtHAv14NL1qvdRTrxMd/  
LAYdifb+hk6FCUnNlj0SLXO0nKvFnKN9YYNMxX2  
Bsqu73gM+Qho1icuebG94a9mCE2XfnARawPdfS3pbRRIXYdRkzuepV  
k2QCtrJsEH  
kag0  
=CwmL  
-----END PGP PUBLIC KEY BLOCK-----
```

Add GPG key

Рис. 4.8: Ключ на Git

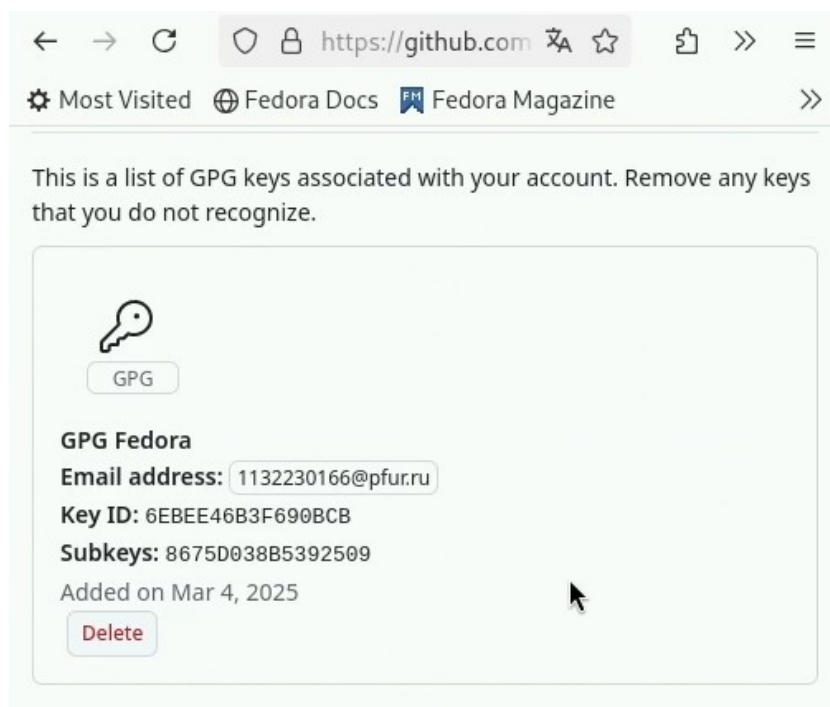


Рис. 4.9: Наличие ключа

7. Настройка автоматических подписей коммитов git. (рис. 4.10) Используя введённый email, укажите Git применять его при подписи коммитов с помощью следующих команд:

- `git config --global user.signingkey`
- `git config --global commit.gpgsign true`
- `git config --global gpg.program $(which gpg2)`

```
[root@ermimash ~]# git config --global user.signingkey 1132230166@pfur.ru
[root@ermimash ~]# git config --global commit.gpgsign true
[root@ermimash ~]# git config --global gpg.program $(which gpg2)
[root@ermimash ~]#
```

Рис. 4.10: Настройка авто-подписей

8. Настройка gh. (рис. 4.11) Мы авторизуемся с помощью команды:

- `gh auth login`

После ввода команды мы отвечаем на несколько вопросов и авторизуемся через браузер. При успешной авторизации получаем данную информацию в терминале (рис. 4.11) и на сайте (рис. 4.12).

```
[root@ermimash ~]# gh auth login
? Where do you use GitHub? GitHub.com
? What is your preferred protocol for Git operations on this host? SSH
? Upload your SSH public key to your GitHub account? /root/.ssh/id_rsa.pub
? Title for your SSH key: Fedora sway
? How would you like to authenticate GitHub CLI? Login with a web browser

! First copy your one-time code: 112A-D5E9
Press Enter to open https://github.com/login/device in your browser...
Authorization required, but no authorization protocol specified

Error: cannot open display: :0
✓ Authentication complete.
- gh config set -h github.com git_protocol ssh
✓ Configured git protocol
! Authentication credentials saved in plain text
✓ Uploaded the SSH key to your GitHub account: /root/.ssh/id_rsa.pub
✓ Logged in as Ermimash
[root@ermimash ~]#
```

Рис. 4.11: Терминал

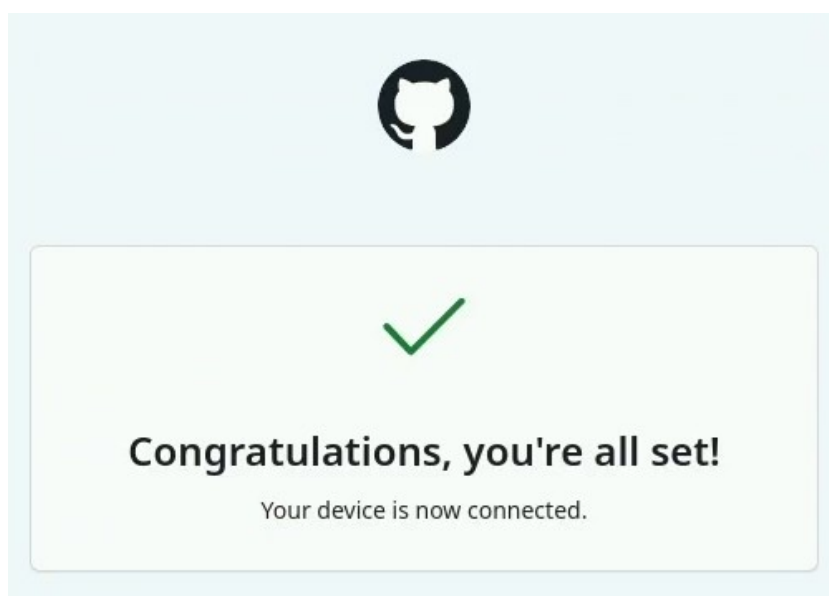


Рис. 4.12: Подтверждение на сайте

9. Создание репозитория курса на основе шаблона. Для начала мы создаём каталог для создания шаблона с помощью следующей команды:
- `mkdir -p ~/work/study/2024-2025/“Операционные системы”` (рис. 4.13)

Далее мы переходим в созданный каталог и создаём шаблон рабочего пространства, указывая актуальный учебный год и название предмета, с помощью следующих команд: - `cd ~/work/study/2024-2025/“Операционные системы”` (рис. 4.13) - `gh repo create study_2024-2025_os-intro --template=yamadharm/course-directory-student-template --public` (рис. 4.13) - `git clone --recursive git@github.com:/study_2022-2023_os-intro.git os-intro` (рис. 4.14)

```
[root@ermimash ~]# mkdir -p ~/work/study/2024-2025/"Операционные системы"
[root@ermimash ~]# cd ~/work/study/2024-2025/"Операционные системы"
[root@ermimash Операционные системы]# gh repo create study_2024-2025_os-intro --template=yamadharm/course-directory-student-template --public
✓ Created repository ErmiMash/study_2024-2025_os-intro on GitHub
https://github.com/ErmiMash/study_2024-2025_os-intro
[root@ermimash Операционные системы]#
```

Рис. 4.13: Создание каталогов

```
[root@ermimash Операционные системы]# git clone --recursive git@github.com:ErmiMash/study_2024-2025_os-intro.git os-intro
Клонирование в «os-intro»...
Enter passphrase for key '/root/.ssh/id_rsa':
Enter passphrase for key '/root/.ssh/id_rsa':
remote: Enumerating objects: 36, done.
remote: Counting objects: 100% (36/36), done.
remote: Compressing objects: 100% (35/35), done.
remote: Total 36 (delta 1), reused 21 (delta 0), pack-reused 0 (from 0)
Получение объектов: 100% (36/36), 19.37 КиБ | 9.69 МиБ/с, готово.
Определение изменений: 100% (1/1), готово.
Подмодуль «template/presentation» (https://github.com/yamadharm/academic-presentation-markdown-template.git) зарегистрирован по пути «template/presentation»
Подмодуль «template/report» (https://github.com/yamadharm/academic-laboratory-report-template.git) зарегистрирован по пути «template/report»
Клонирование в «/root/work/study/2024-2025/Операционные системы/os-intro/template/presentation»...
remote: Enumerating objects: 111, done.
remote: Counting objects: 100% (111/111), done.
remote: Compressing objects: 100% (77/77), done.
remote: Total 111 (delta 42), reused 100 (delta 31), pack-reused 0 (from 0)
Получение объектов: 100% (111/111), 102.17 КиБ | 1.10 МиБ/с, готово.
Определение изменений: 100% (42/42), готово.
Клонирование в «/root/work/study/2024-2025/Операционные системы/os-intro/template/report»...
remote: Enumerating objects: 142, done.
remote: Counting objects: 100% (142/142), done.
remote: Compressing objects: 100% (97/97), done.
remote: Total 142 (delta 60), reused 121 (delta 39), pack-reused 0 (from 0)
Получение объектов: 100% (142/142), 341.09 КиБ | 2.37 МиБ/с, готово.
Определение изменений: 100% (60/60), готово.
Submodule path 'template/presentation': checked out 'c9b2712b4b2d431ad5086c9c72a02bd2fcald4a6'
Submodule path 'template/report': checked out 'c26e22effe7b3e0495707d82ef561ab185f5c748'
[root@ermimash Операционные системы]#
```

Рис. 4.14: git clone

Проверяем наличие созданного шаблона на сайте. (рис. 4.15)



Рис. 4.15: Наличие шаблона на сайте

10. Настройка каталога курса. Мы переходим в каталог курса и удаляем лишние файлы с помощью следующих команд: (рис. 4.16)

- `cd ~/work/study/2022-2023/“Операционные системы”/os-intro`
- `rm package.json`

```
[root@ermimash Операционные системы]# cd ~/work/study/2024-2025/“Операционные системы”/os-intro
[root@ermimash os-intro]# ls
CHANGELOG.md  COURSE  Makefile  README.en.md  README.md
config        LICENSE  package.json  README.git-flow.md  template
[root@ermimash os-intro]# rm package.json
rm: удалить обычный файл 'package.json'? y
[root@ermimash os-intro]# ls
CHANGELOG.md  COURSE  Makefile  README.git-flow.md  template
config        LICENSE  README.en.md  README.md
[root@ermimash os-intro]#
```

Рис. 4.16: Удаление файлов

Далее создаём необходимфе каталоги: (рис. 4.17) - `echo os-intro > COURSE` - `make prepare`

```
[root@ermimash os-intro]# echo os-intro > COURSE
[root@ermimash os-intro]# make
Usage:
  make <target>

Targets:
  list           List of courses
  prepare       Generate directories structure
  submodule     Update submules

[root@ermimash os-intro]# make prepare
[root@ermimash os-intro]#
```

Рис. 4.17: Создание каталогов

После создания каталогов отправляем файлы на сервер: (рис. 4.18) - `git add . - git commit -am 'feat(main): make course structure' - git push`

```
create mode 100644 project-personal/stage5/report/pandoc/filters/pandocxnos/core.py
create mode 100644 project-personal/stage5/report/pandoc/filters/pandocxnos/main.py
create mode 100644 project-personal/stage5/report/pandoc/filters/pandocxnos/pandocattributes.py
create mode 100644 project-personal/stage5/report/report.md
create mode 100644 project-personal/stage6/presentation/.projectile
create mode 100644 project-personal/stage6/presentation/.texlabroot
create mode 100644 project-personal/stage6/presentation/Makefile
create mode 100644 project-personal/stage6/presentation/image/kulyabov.jpg
create mode 100644 project-personal/stage6/presentation/presentation.md
create mode 100644 project-personal/stage6/report/Makefile
create mode 100644 project-personal/stage6/report/bib/cite.bib
create mode 100644 project-personal/stage6/report/image/placeimg_800_600_tech.jpg
create mode 100644 project-personal/stage6/report/pandoc/csl/gost-r-7-0-5-2008-numerical.csl
create mode 100755 project-personal/stage6/report/pandoc/filters/pandoc_eqnos.py
create mode 100755 project-personal/stage6/report/pandoc/filters/pandoc_fignos.py
create mode 100755 project-personal/stage6/report/pandoc/filters/pandoc_secnos.py
create mode 100755 project-personal/stage6/report/pandoc/filters/pandoc_tablenos.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandocxnos/__init__.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandocxnos/core.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandocxnos/main.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandocxnos/pandocattributes.py
create mode 100644 project-personal/stage6/report/report.md
[root@ermimash os-intro]# git push
Enter passphrase for key '/root/.ssh/id_rsa':
Перечисление объектов: 40, готово.
Подсчет объектов: 100% (40/40), готово.
При сжатии изменений используется до 3 потоков
Сжатие объектов: 100% (30/30), готово.
Запись объектов: 100% (38/38), 342.31 КиБ | 2.76 МиБ/с, готово.
Total 38 (delta 4), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (4/4), completed with 1 local object.
To github.com:ErmiMash/study_2024-2025_os-intro.git
   b6401b7..8126b75  master -> master
[root@ermimash os-intro]#
```

Рис. 4.18: Отправка файлов на сервер

5 Выводы

Изучила идеологии и применение средств контроля версий; освоила умения по работе с git.

6 Контрольные вопросы

1. Это программное обеспечение для облегчения работы с изменяющейся информацией. Позволяет хранить несколько версий одного документа, возвращаться к более ранним версиям, определять кто и когда сделал изменение, и многое другое.
2. Хранилище - место хранения всех версий и служеб. информации. Commit - версия; процесс создания новой версии. История - место, где хранятся все коммиты, по которым можно посмотреть данные по коммитам. Рабочая копия - тек. состояние файла проекта, осн. на версии из хранилища.
3. Централизованные VCS: одно основное хранилище всего проекта, каждый копирует себе файлы и изменяет, а потом добавляет изменения; Децентрализованные: у каждого пользователя свой вариант (возможно не один) репозитория.
4. Инициализация репозитория, добавление файлов, фиксация изменений, проверка статуса, просмотр истории коммитов, создание и слияние веток, удаление веток, откат изменений, резервное копирование.
5. Клонирование репозитория, создание новой ветки, создание новой ветки, внесение изменений, добавление изменений в индекс, создание коммита, обновление локальной ветки, слияние изменений, отправка изменений в удалённый репозиторий, открытие Pull Request (PR), код-ревью, слияние PR, регулярное обновление.
6. Две основные задачи: 1 - хранить информацию о всех изменениях в коде, начиная с самой первой строчки; 2 - обеспечивать удобство командной

работы с кодом.

7. `git init` (инициализировать тек. каталог как `git`-репозиторий), `git -version` (проверка версии `Git`), `git remote` (посмотреть список удалённых репозиторий), `git remote -v` (более подробный вывод), `git add .` (позвол. охватить все файлы в тек. каталоге (вместе с теми, которые начинаются на точку)), `git -http` (узнать больше о доступных параметрах и командах), `git push origin master` (передать лок. коммиты в ветку удал. репозитория)
8. Локальный: создание новой ветки, работа без интернета, сохранение изменений; удалённый: отправки кодов, получение изменений, создание новых веток.
9. Ветки нужны, чтобы несколько программистов могли работать над одним и тем же проектом (или даже файлом) одновременно, не мешая друг другу. Также используются для тестов: чтобы не мешать основному проекту создаётся отдельная ветка для экспериментов.
10. Игнорируемые файлы — это, как правило, артефакты сборки и файлы, генерируемые машиной из исходных файлов в вашем репозитории, либо файлы, которые по какой-либо иной причине не должны попадать в коммиты. Игнорируемые файлы отслеживаются в специальном файле `.gitignore`, который регистрируется в корневом каталоге репозитория.

Список литературы

1. Д. К. Лабораторная работа № 3. Markdown. RUDN.