

Отчёт по лабораторной работе №2:

Первоначальна настройка git

Коротков Андрей Романович

Оглавление

1	Цель работы	5
2	Задания	6
3	Выполнение лабораторной работы	7
3.0.1	Задание 1	7
3.0.2	Задание 2	8
3.0.3	Задание 3	11
3.0.4	Задание 4	14
4	Выводы	17
5	Ответы на контрольные вопросы	18

Список иллюстраций

3.1	Установка программного обеспечения	7
3.2	Настройка имени и почты владельца	8
3.3	Настройка конфига	8
3.4	Создание ключа ssh(rsa)	9
3.5	Создание ключа ssh(ed25519)	10
3.6	Создание ключа pgr	11
3.7	Копирование ключа pgr	12
3.8	Добавление ключа pgr	12
3.9	Настройка подписей	13
3.10	Настройка подписей	14
3.11	Создание репозитория	15
3.12	Создание структуры каталога	15
3.13	Отправка файлов на сервер	16

List of Tables

1 Цель работы

Изучить идеологию и применение средств контроля версий. Освоить умения по работе с git.

2 Задания

1. Создать базовую конфигурацию для работы с git.
2. Создать ключи SSH и PGP.
3. Настроить подписи git.
4. Создать локальный каталог для выполнения заданий по предмету.

3 Выполнение лабораторной работы

3.0.1 Задание 1

Для начала установим программное обеспечение командами `dnf install git` и `dnf install gh`

```
foot
[root@arkorotkov ~]# dnf install git
Fedora 40 - x86_64 - Updates                                     46 kB/s | 28 kB   00:00
Последняя проверка окончания срока действия метаданных: 0:00:01 назад, Сб 15 июн 2024 17:42:39.
Пакет git-2.45.2-2.fc40.x86_64 уже установлен.
Зависимости разрешены.
Нет действий для выполнения.
Выполнено!
[root@arkorotkov ~]# dnf install gh
Последняя проверка окончания срока действия метаданных: 0:01:13 назад, Вс 16 июн 2024 12:34:50.
Зависимости разрешены.
=====
Пакет                Архитектура          Версия              Резепозиторий        Размер
=====
Установка:
gh                   x86_64               2.45.0-1.fc40      fedora                8.7 М
=====
Результат транзакции
=====
Установка 1 Пакет

Объем загрузки: 8.7 М
Объем изменений: 43 М
Продолжить? [д/Н]: y
Загрузка пакетов:
gh-2.45.0-1.fc40.x86_64.rpm                                9.5 MB/s | 8.7 MB   00:00
=====
Общий размер
Проверка транзакции                                         5.8 MB/s | 8.7 MB   00:01
Проверка транзакции успешно завершена.
Идет проверка транзакции
Тест транзакции проведен успешно.
Выполнение транзакции
Подготовка :
Установка : gh-2.45.0-1.fc40.x86_64                      1/1
Запуск скрипглета: gh-2.45.0-1.fc40.x86_64                1/1
Установлен:
gh-2.45.0-1.fc40.x86_64
Выполнено!
[root@arkorotkov ~]#
```

Рис. 3.1: Установка программного обеспечения

Проведём базовую настройку:

- Зададим имя и email владельца репозитория командами `git config --global`

user.name "Name Surname" и git config --global user.email "work@mail"

```
foot
[root@arkorotkov ~]# git config --global user.name "Andrey Korotkov"
[root@arkorotkov ~]# git config --global user.email "korotandro@gmail.com"
[root@arkorotkov ~]#
```

Рис. 3.2: Настройка имени и почты владельца

- Настроим utf-8 в выводе сообщений git командой git config --global core.quotepath false
- Зададим имя начальной ветки командой git config --global init.defaultBranch master
- Настроим параметр autocrlf командой git config --global core.autocrlf input
- Настроим параметр safecrlf командой git config --global core.safecrlf warn

```
foot
[root@arkorotkov ~]# git config --global core.quotepath false
[root@arkorotkov ~]# git config --global init.defaultBranch master
[root@arkorotkov ~]# git config --global core.autocrlf input
[root@arkorotkov ~]# git config --global core.safecrlf warn
[root@arkorotkov ~]#
```

Рис. 3.3: Настройка конфига

3.0.2 Задание 2

Создадим ключ ssh о алгоритму rsa с размером 4096 бит командой ssh-keygen -t rsa -b 4096.


```
foot
[root@arkorotkov ~]# 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 (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:jKm8fQ1sj1x6CC/SvZeXophGSvHwihVKV4cFSwIX110 root@arkorotkov
The key's randomart image is:
+---[RSA 4096]-----+
|  ..+.*..E      |
|  . ++.0        |
|  ...          |
|  . =  +        |
|  . o *o.S      |
|  ..0.= + .     |
|  +0= * 0 . .   |
|  . +0+o0 B o   |
|  .0+00= o      |
+----[SHA256]-----+
[root@arkorotkov ~]#
```

Рис. 3.4: Создание ключа ssh(rsa)

Создадим ключ ssh с алгоритму ed25519 командой `ssh-keygen -t ed25519`

```
foot
[root@arkorotkov ~]# ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/root/.ssh/id_ed25519):
Enter passphrase (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:f2ZTZVe5pWCKUnGHk8HCCx1HXp1Bya5md+0GTzL/aKg root@arkorotkov
The key's randomart image is:
+--[ED25519 256]--+
|      .+=+B++  o|
|    oo+o=.B   .o|
|      o.+ = .  *|
|      . o . . . +.|
|      .S . . . .|
|      .+ . .= o|
|      o..=o o |
|      +....=|
|      E. ...o|
+-----[SHA256]-----+
[root@arkorotkov ~]#
```

Рис. 3.5: Создание ключа ssh(ed25519)

Создадим ключ pgp командой `gpg --full-generate-key`. Из предложенных опций выбираем:

- тип RSA and RSA;
- размер 4096;
- выбираем срок действия;

Вводим личную информацию и комментарий.

```
foot
(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: Andrey Korotkov
Адрес электронной почты: korotandro@gmail.com
Примечание:
Вы выбрали следующий идентификатор пользователя:
    "Andrey Korotkov <korotandro@gmail.com>"

Сменить (N)Имя, (C)Примечание, (E)Адрес; (O)Принять/(Q)Выход? o
Необходимо получить много случайных чисел. Желательно, чтобы Вы
в процессе генерации выполняли какие-то другие действия (печать
на клавиатуре, движения мыши, обращения к дискам); это даст генератору
случайных чисел больше возможностей получить достаточное количество энтропии.
Необходимо получить много случайных чисел. Желательно, чтобы Вы
в процессе генерации выполняли какие-то другие действия (печать
на клавиатуре, движения мыши, обращения к дискам); это даст генератору
случайных чисел больше возможностей получить достаточное количество энтропии.
gpg: /root/.gnupg/trustdb.gpg: создана таблица доверия
gpg: создан каталог '/root/.gnupg/openpgp-revocs.d'
gpg: сертификат отзыва записан в '/root/.gnupg/openpgp-revocs.d/A38FE3B5D9B1D354D86BB4D8E681C3069A1E279B.rev'.
открытый и секретный ключи созданы и подписаны.

pub   rsa4096 2024-06-16 [SC]
       A38FE3B5D9B1D354D86BB4D8E681C3069A1E279B
uid           Andrey Korotkov <korotandro@gmail.com>
sub   rsa4096 2024-06-16 [E]

[root@aikorotkov ~]#
```

Рис. 3.6: Создание ключа gpg

3.0.3 Задание 3

3.0.3.1 Добавление PGP ключа в GitHub

Копируем приватный ключ в файл командой `gpg --armor --export почта_владельца`
> /файл.txt

```
foot
[root@arkorotkov ~]# gpg --armor --export korotandro@gmail.com > /media/sf_work/gpg.txt
[root@arkorotkov ~]#
```

Рис. 3.7: Копирование ключа pgr

Переходим на <https://github.com/settings/keys> и добавляем полученный ключ.

Add new GPG key

Title

Key

```
-----BEGIN PGP PUBLIC KEY BLOCK-----

mQINBGZutc8BEACb/mleyZLVjl+4Of2uko/QecBJHE4f2YWh1GwWSgJURgvHbsg0
92zMAhBGTkoqxRFvWHyDMuPdu5Q7xP5DNNSGTLpUoqYLM6enTi03GOyDpHolf
QNE
kbDO/CaRo+IMIYYva/Um+tCOWpLwcrxt6KPsUwxCr4xZ85yWaw/rEvNNdkWDUNn
I
WzQ5266KmEkLIINiLp6V0+VO8Zc6pZhTtrblQE0zr3KeOPRqvONTD4bDR/Sh+J4t
aGaAjVMYzg/3rZ62TPuxvPsSBnhy/tcvpO6ybFRLanCkj8G8DN4G9fuDV73q+sLy
mDaxOcngSwveTUrwNjVjJ1zpnwsr1Q+M+bcFZZ+cu/K3CIAgwFzmNO0v5HRef2b
5
7aq84/iG1qcl73Utez/juPyncdvTXdEx7xejZR9+TACuyzh+zBXp2nP0Sm632NZ
F0eCuqw5GZjQM0PHRhLLnigPKM8imtzyWID6cY8ZJ0hm3kKtfCxTfVBC7aWBdSxr
-----END PGP PUBLIC KEY BLOCK-----
```

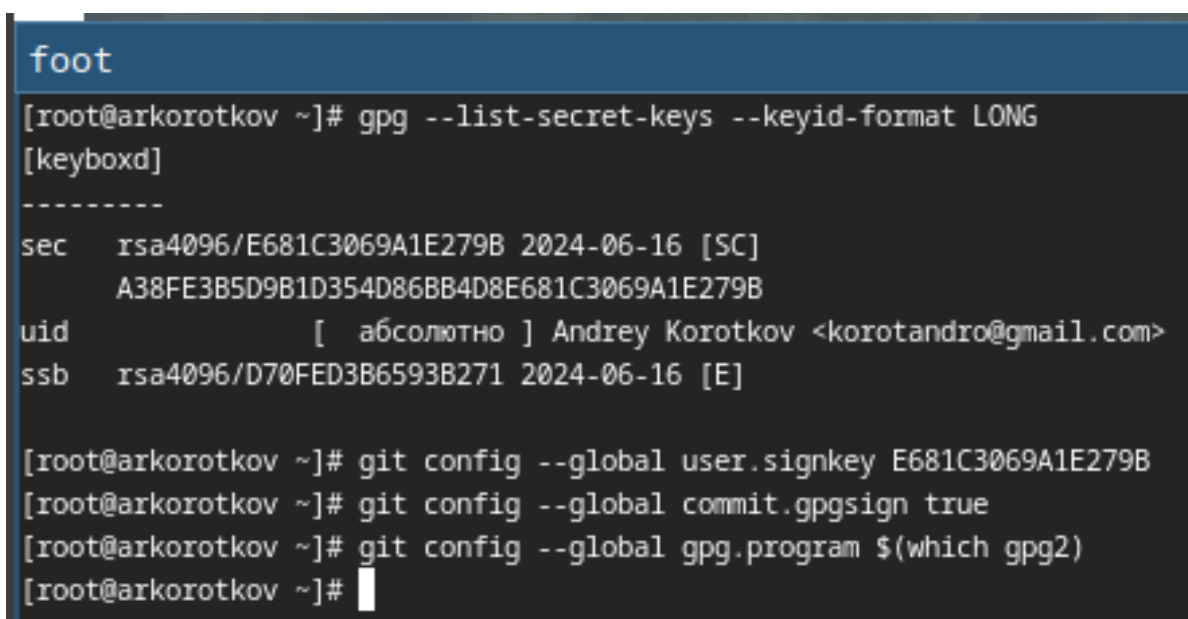
Add GPG key

Рис. 3.8: Добавление ключа pgr

3.0.3.2 Настройка автоматических подписей коммитов git

Используя введённый email, укажем Git применять его при подписи коммитов используя команды:

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



```
foot
[root@arkorotkov ~]# gpg --list-secret-keys --keyid-format LONG
[keyboxd]
-----
sec   rsa4096/E681C3069A1E279B 2024-06-16 [SC]
      A38FE3B5D9B1D354D86BB4D8E681C3069A1E279B
uid           [ абсолютно ] Andrey Korotkov <korotandro@gmail.com>
ssb   rsa4096/D70FED3B6593B271 2024-06-16 [E]

[root@arkorotkov ~]# git config --global user.signkey E681C3069A1E279B
[root@arkorotkov ~]# git config --global commit.gpgsign true
[root@arkorotkov ~]# git config --global gpg.program $(which gpg2)
[root@arkorotkov ~]#
```

Рис. 3.9: Настройка подписей

3.0.3.3 Настройка gh

Авторизация командой `gh auth login`

```
foot
[root@arkorotkov ~]# gh auth login
? What account do you want to log into? 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: OS_labs
? How would you like to authenticate GitHub CLI? Paste an authentication token
Tip: you can generate a Personal Access Token here https://github.com/settings/tokens
The minimum required scopes are 'repo', 'read:org', 'admin:public_key'.
? Paste your authentication token: *****
- 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 korotandr
[root@arkorotkov ~]#
```

Рис. 3.10: Настройка подписей

3.0.4 Задание 4

- Создадим каталог для репозитория командой `mkdir -p ~/study/2024/OS`
- Перейдём в созданный каталог командой `cd ~/study/2024/OS`
- Создадим репозиторий на основе шаблона командой `gh repo create study_2024_os-intro --template=yamadharm/course-directory-student-template --public`
- Клонировать созданный репозиторий в каталог командой `git clone --recursive git@github.com:korotandr/study_2024_os-intro.git os-intro`

```
foot
[root@arkorotkov ~]# mkdir -p /media/sf_work/study/2024/OS/
[root@arkorotkov ~]# cd /media/sf_work/study/2024/OS/
[root@arkorotkov OS]# gh repo create study_2024_os-intro --template=yamadharma/course-directory-student-template --public
✓ Created repository korotandr/study_2024_os-intro on GitHub
https://github.com/korotandr/study_2024_os-intro
[root@arkorotkov OS]# git clone --recursive git@github.com:korotandr/study_2024_os-intro.git os-intro
Клонирование в «os-intro»...
The authenticity of host 'github.com (140.82.121.4)' can't be established.
ED25519 key fingerprint is SHA256:+D1Y3wvvV6TuJJhbpZisF/zLDA0zPMSvHdkr4UvCOqU.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added 'github.com' (ED25519) to the list of known hosts.
remote: Enumerating objects: 32, done.
remote: Counting objects: 100% (32/32), done.
remote: Compressing objects: 100% (31/31), done.
remote: Total 32 (delta 1), reused 18 (delta 0), pack-reused 0
Получение объектов: 100% (32/32), 18.60 КиБ | 9.30 МБ/с, готово.
Определение изменений: 100% (1/1), готово.
Подмодуль «template/presentation» (https://github.com/yamadharma/academic-presentation-markdown-template.git) зарегистрирован по пути «template/presentation»
Подмодуль «template/report» (https://github.com/yamadharma/academic-laboratory-report-template.git) зарегистрирован по пути «template/report»
Клонирование в «/media/sf_work/study/2024/OS/os-intro/template/presentation»...
remote: Enumerating objects: 95, done.
remote: Counting objects: 100% (95/95), done.
remote: Compressing objects: 100% (67/67), done.
remote: Total 95 (delta 34), reused 87 (delta 26), pack-reused 0
Получение объектов: 100% (95/95), 96.99 КиБ | 1.11 МБ/с, готово.
Определение изменений: 100% (34/34), готово.
Клонирование в «/media/sf_work/study/2024/OS/os-intro/template/report»...
remote: Enumerating objects: 126, done.
remote: Counting objects: 100% (126/126), done.
remote: Compressing objects: 100% (87/87), done.
remote: Total 126 (delta 52), reused 108 (delta 34), pack-reused 0
Получение объектов: 100% (126/126), 335.80 КиБ | 2.32 МБ/с, готово.
Определение изменений: 100% (52/52), готово.
Submodule path 'template/presentation': checked out '40a1761813e197d00e8443ff1ca72c60a304f24c'
Submodule path 'template/report': checked out '7c31ab8e5dfa8cdb2d67caeb8a19ef8028ced88e'
[root@arkorotkov OS]#
```

Рис. 3.11: Создание репозитория

Настройка каталога курса:

- Перейти в каталог курса командой `cd os-intro`
- Удалить лишние файлы командой `rm package.json`
- Создать необходимые каталоги командой `make prepare`

```
foot
[root@arkorotkov OS]# cd os-intro/
[root@arkorotkov os-intro]# rm package.json
rm: удалить обычный файл 'package.json'? y
[root@arkorotkov os-intro]# make prepare
[root@arkorotkov os-intro]#
```

Рис. 3.12: Создание структуры каталога

Отправка файлов на сервер последовательностью команд:

- `git add .`
- `git commit -am 'feat(main): make course structure'`
- `git push`

```
foot
create mode 100644 project-personal/stage4/report/report.md
create mode 100644 project-personal/stage5/presentation/Makefile
create mode 100644 project-personal/stage5/presentation/image/kulyabov.jpg
create mode 100644 project-personal/stage5/presentation/presentation.md
create mode 100644 project-personal/stage5/report/Makefile
create mode 100644 project-personal/stage5/report/bib/cite.bib
create mode 100644 project-personal/stage5/report/image/placeimg_800_600_tech.jpg
create mode 100644 project-personal/stage5/report/pandoc/csl/gost-r-7-0-5-2008-numeric.csl
create mode 100644 project-personal/stage5/report/pandoc/filters/pandoc_eqnos.py
create mode 100644 project-personal/stage5/report/pandoc/filters/pandoc_fignos.py
create mode 100644 project-personal/stage5/report/pandoc/filters/pandoc_secnos.py
create mode 100644 project-personal/stage5/report/pandoc/filters/pandoc_tablenos.py
create mode 100644 project-personal/stage5/report/pandoc/filters/pandocxnos/__init__.py
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/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-numeric.csl
create mode 100644 project-personal/stage6/report/pandoc/filters/pandoc_eqnos.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandoc_fignos.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandoc_secnos.py
create mode 100644 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@arkorotkov os-intro]# git push
Перечисление объектов: 38, готово.
Подсчет объектов: 100% (38/38), готово.
При сжатии изменений используется до 2 потоков
Сжатие объектов: 100% (30/30), готово.
Запись объектов: 100% (37/37), 342.19 КиБ | 2.74 МБ/с, готово.
Total 37 (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:korotandr/study_2024_os-intro.git
 82b6022..bacf784 master -> master
[root@arkorotkov os-intro]#
```

Рис. 3.13: Отправка файлов на сервер

4 Выводы

В ходе данной лабораторной работы я смог изучить идеологию и применение средств контроля версий, а так же освоил основные аспекты работы с git.

5 Ответы на контрольные вопросы

1. Системы контроля версий (Version Control Systems, VCS) — это программные инструменты, предназначенные для управления изменениями в исходном коде, документации и других данных в процессе разработки. Основные задачи:

- Отслеживание изменений
- Совместная работа
- Управление версиями
- Восстановление версий
- Резервное копирование и восстановление

2. Хранилище: Это центральное место, где сохраняется вся история изменений проекта, включая все версии файлов, метаданные изменений (например, авторы и временные метки), и другие данные, необходимые для управления версиями.

Commit: Это операция, при которой изменения, внесённые в файлы рабочей копии, сохраняются в хранилище как отдельная версия (коммит).

История: Это последовательность всех коммитов, сохранённых в хранилище.

Рабочая копия: Это текущая версия файлов проекта на локальной машине разработчика, с которой он непосредственно работает.

Взаимосвязь этих понятий:

- Разработчик клонирует или извлекает репозиторий, создавая рабочую копию на своём компьютере.
- В рабочей копии он вносит изменения в файлы проекта.

- После завершения работы он фиксирует (commit) изменения, добавляя новый коммит в историю репозитория.
- Хранилище обновляется, и новая версия проекта становится доступной для других участников команды.
- История изменений позволяет отслеживать все внесённые изменения и при необходимости откатываться к предыдущим версиям.

3. Централизованные системы контроля версий (CVCS)

- Единое хранилище
- Обмен данными через сервер
- Простота администрирования
- Примеры:
 - CVS (Concurrent Versions System)
 - Subversion (SVN)

Децентрализованные системы контроля версий (DVCS)

- Полные копии репозитория
- Работа офлайн
- Масштабируемость и гибкость
- Примеры:
 - Git
 - Mercurial

4. Примерный процесс работы:

- Инициализация или клонирование репозитория
- Работа с файлами
- Отслеживание изменений
- Коммит изменений

- Синхронизация
- Просмотр истории

5.Примерный процесс работы в команде:

- Клонирование репозитория
- Обновление перед началом работы
- Работа с файлами и коммиты
- Регулярная синхронизация
- Разрешение конфликтов
- Push изменений
- Просмотр истории и анализ

6.Основные задачи, решаемые инструментальным средством git:

- Отслеживание изменений
- Управление версиями
- Совместная работа
- Создание и управление ветками
- Объединение изменений (Merge)
- Разрешение конфликтов
- История изменений
- Обеспечение целостности данных
- Резервное копирование и восстановление
- Работа офлайн
- Интеграция с CI/CD

7.git init — создание нового репозитория в текущей директории.

git clone URL — клонирование удалённого репозитория на локальную машину.

git add file — добавление файла в коммит.

git commit -m “Сообщение” — создание коммита с описанием изменений.

git branch branch-name — создание новой ветки.

`git checkout branch-name` — переключение на другую ветку.

`git merge branch-name` — объединение изменений из указанной ветки в текущую.

`git push origin branch-name` — отправка локальных изменений в удалённый репозиторий.

`git pull origin branch-name` — извлечение изменений из удалённого репозитория и объединение их с локальными.

`git log` — просмотр истории коммитов.

`git fetch` — синхронизация с удалённым репозиторием

8. Работа с локальным репозиторием:

- `git init`
- `git add file1.txt file2.txt`
- `git commit -m "Initial commit"`
- `git status`
- `git log`
- `git branch new-feature`
- `git checkout new-feature`
- `git checkout master`
- `git merge new-feature`
- Разрешение конфликтов

Работа с удалённым репозиторием:

- `git clone https://github.com/user/repository.git`
- `git remote add origin https://github.com/user/repository.git`
- `git push origin master`
- `git pull origin master`
- `git remote -v`
- `git checkout -b new-feature`
- `git push origin new-feature`
- `git push origin -delete new-feature`

- `git fetch`

9. Ветви (branches) в системах контроля версий, являются важным инструментом для управления разработкой проекта. Они позволяют разработчикам работать над различными задачами, функциями или исправлениями независимо друг от друга.

10. Игнорирование некоторых файлов при commit в системах контроля версий позволяет исключить из репозитория временные, сгенерированные или конфиденциальные файлы, которые не должны быть частью версионного контроля. Для этого в Git используется специальный файл `.gitignore`.