

Отчёт по лабораторной работе № 2

НБИбд-01-23

Анастасия Романовна Зинченко

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
4	Выводы	17
	Список литературы	18

Список иллюстраций

3.1	Установка git	7
3.2	Установка gh	7
3.3	Установка gh	7
3.4	Настройка utf-8	8
3.5	Ветка master	8
3.6	Параметр autocrlf	8
3.7	Параметр safecrlf	8
3.8	Ключ ssh по алгоритму rsa	9
3.9	Ключ ssh по алгоритму ed25519	9
3.10	Генерация ключа	10
3.11	Учётная запись Github	11
3.12	Список ключей	11
3.13	Список ключей	12
3.14	Подписи коммиттов	12
3.15	Авторизация	12
3.16	Создание репозитория	13
3.17	Создание репозитория	13
3.18	Удаление лишних файлов	13
3.19	Создание каталогов	13
3.20	Создание каталогов	14
3.21	Проверка	14

Список таблиц

1 Цель работы

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

2 Задание

1. Установка программного обеспечения а. Установка git б. Установка gh
2. Базовая настройка git
3. Создайте ключи ssh
4. Создайте ключи pgp
5. Настройка github
6. Добавление PGP ключа в GitHub
7. Настройка автоматических подписей коммитов git
8. Настройка gh
9. Шаблон для рабочего пространства а. Создание репозитория курса на основе шаблона б. Настройка каталога курса
10. Контрольные вопросы

3 Выполнение лабораторной работы

Установила git с помощью команды `dnf install git` (рис. 3.1).

```
[arzinchenko@arzinchenko ~]$ sudo -i
[sudo] пароль для arzinchenko:
[root@arzinchenko ~]# dnf install git
Fedora 39 - x86_64 - Updates                                12 kB/s | 20 kB  00:01
Fedora 39 - x86_64 - Updates                                2.3 MB/s | 5.4 MB 00:02
```

Рис. 3.1: Установка git

Установила gh с помощью команды `dnf install gh` (рис. 3.2).

```
[root@arzinchenko ~]# dnf install gh
Последняя проверка окончания срока действия метаданных: 0:00:27 назад, Ср 21 фев 2024 21:08:29.
Зависимости разрешены.
=====
Пакет      Архитектура      Версия      Репозиторий      Размер
-----
Установка: gh-2.43.1-1.fc39 x86_64      2.43.1-1.fc39 updates          9.1 М
=====
Результат транзакции
=====
Установка 1 Пакет
=====
Объем загрузки: 9.1 М
Объем изменений: 46 М
Продолжить? [д/н]: у
Загрузка пакетов:
gh-2.43.1-1.fc39.x86_64.rpm                                3.7 MB/s | 9.1 MB  00:02
-----
Общий размер                                              3.4 MB/s | 9.1 MB  00:02
=====
Проверка транзакции
Проверка транзакции успешно завершена.
Идет проверка транзакции
Тест транзакции проведен успешно.
Выполнение транзакции
Подготовка : gh-2.43.1-1.fc39.x86_64                      1/1
Установка  : gh-2.43.1-1.fc39.x86_64                      1/1
Запуск скриптов: gh-2.43.1-1.fc39.x86_64                  1/1
Проверка   : gh-2.43.1-1.fc39.x86_64                      1/1
```

Рис. 3.2: Установка gh

Задала имя и email своего репозитория с помощью команд `git config --global user.name "Anastasiia Zinchenko"` `git config --global user.email "zinchenkoa06zinchenko@yandex.ru"` (рис. 3.3).

```
[root@arzinchenko ~]# git config --global user.name "Anastasiia Zinchenko"
[root@arzinchenko ~]# git config --global user.email "1132231832@pfur.ru"
```

Рис. 3.3: Установка gh

Настроила utf-8 в выводе сообщений git с помощью команды `git config --global core.quotePath false` (рис. 3.4).

```
[root@arzinchenko ~]# git config --global core.quotePath false
[root@arzinchenko ~]#
```

Рис. 3.4: Настройка utf-8

Задала имя начальной ветки с помощью команды `git config --global init.defaultBranch master` (рис. 3.5).

```
[root@arzinchenko ~]# git config --global init.defaultBranch master
[root@arzinchenko ~]#
```

Рис. 3.5: Ветка master

Задала параметр `autocrlf` с помощью команды `git config --global core.autocrlf input` (рис. 3.6).

```
[root@arzinchenko ~]# git config --global core.autocrlf input
[root@arzinchenko ~]#
```

Рис. 3.6: Параметр autocrlf

Задала параметр `safecrlf` с помощью команды `git config --global core.safecrlf warn` (рис. 3.7).

```
[root@arzinchenko ~]# git config --global core.safecrlf warn
[root@arzinchenko ~]#
```

Рис. 3.7: Параметр safecrlf

Создала ключ ssh по алгоритму rsa с ключём размером 4096 бит с помощью команды `ssh-keygen -t rsa -b 4096` (рис. 3.8).


```
[root@arzinchenko ~]# 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:W//b42RPQ80cZabQLpAXCl+Ri0bXJH0sMh/P4S44Nzw root@arzinchenko
The key's randomart image is:
+---[RSA 4096]-----+
|      .  .0..+ |
|      o+o*o*=+ |
|      .o= *oB. |
|      o o o+=+ |
|      S.. o...+ |
|      o + E.. |
|      . + ++. |
|      .o+o |
|      +o+ |
+----[SHA256]-----+
[root@arzinchenko ~]#
```

Рис. 3.8: Ключ ssh по алгоритму rsa

Создала ключ ssh по алгоритму ed25519 с помощью команды `ssh-keygen -t ed25519` (рис. 3.9).

```
[root@arzinchenko ~]# 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:q2RMCYXiUWZ/5h9H1by1yX/MgJz0uj0C3XMXIf7T0Tc root@arzinchenko
The key's randomart image is:
+--[ED25519 256]--+
|  .+..      .+.. |
|  oo.o      .o. |
|  .o. .o o+.oo |
|  . . = .+.+= |
|      o S . o..++ |
|      o o o.. EB |
|      + .o.. . B |
|      o .. o * .oo |
|      . . = .. |
+----[SHA256]-----+
[root@arzinchenko ~]#
```

Рис. 3.9: Ключ ssh по алгоритму ed25519

Сгенерировала ключ gpg с помощью команды `gpg --full-generate-key` (рис. 3.10).

```

[root@arzinchenko ~]# gpg --full-generate-key
gpg (GnuPG) 2.4.3; Copyright (C) 2023 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: Anastasiia
Адрес электронной почты: 1132231832@pfur.ru
Примечание:
Вы выбрали следующий идентификатор пользователя:
  "Anastasiia <1132231832@pfur.ru>"

Сменить (N)Имя, (C)Примечание, (E)Адрес; (O)Принять/(Q)Выход? 0
Сменить (N)Имя, (C)Примечание, (E)Адрес; (O)Принять/(Q)Выход? o

```

Рис. 3.10: Генерация ключа

Из предложенных опций выбирала тип RSA and RSA; размер 4096; срок действия; имя; адрес электронной почты

У меня уже была создана учётная запись на Github и были заполнены основные данные (рис. 3.11).

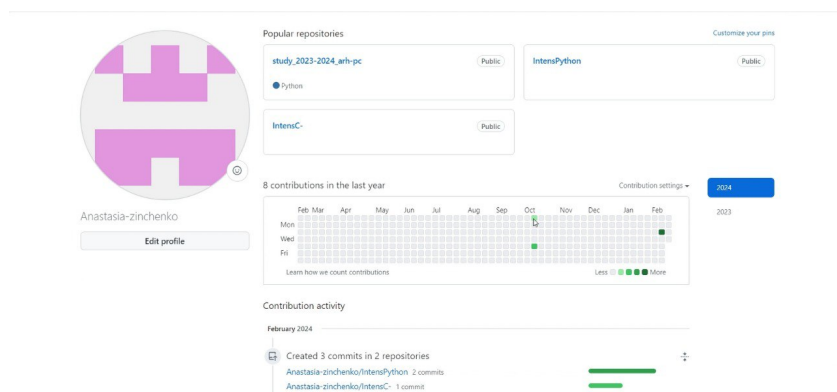


Рис. 3.11: Учётная запись Github

Вывела список ключей и скопировала отпечаток приватного ключа с помощью команды `gpg --list-secret-keys --keyid-format LONG` (рис. 3.12).

```
[root@arzinchenko ~]# 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: 2 подписанных: 0 доверие: 0-, 0q, 0n, 0m, 0f, 2u
[keyboard]
-----
sec rsa4096/7E03AC140CE4B50C 2024-02-21 [SC]
uid [ абсолютно ] Anastasiia <1132231832@pfur.ru>
ssb rsa4096/8F16CC36EA5358DA 2024-02-21 [E]

sec rsa4096/97D995650A79FBF3 2024-02-21 [SC]
uid [ абсолютно ] Anastasiia <zinchenkoa06zinchenko@yandex.ru>
ssb rsa4096/E89D0022D83C1E33 2024-02-21 [E]

[root@arzinchenko ~]#
```

Рис. 3.12: Список ключей

Скопировала сгенерированный PGP ключ в буфер обмена с помощью команды `gpg --armor --export | xclip -sel clip`. Перешла в настройки GitHub, нажала на кнопку New GPG key и вставила полученный ключ в поле ввода (рис. 3.13).

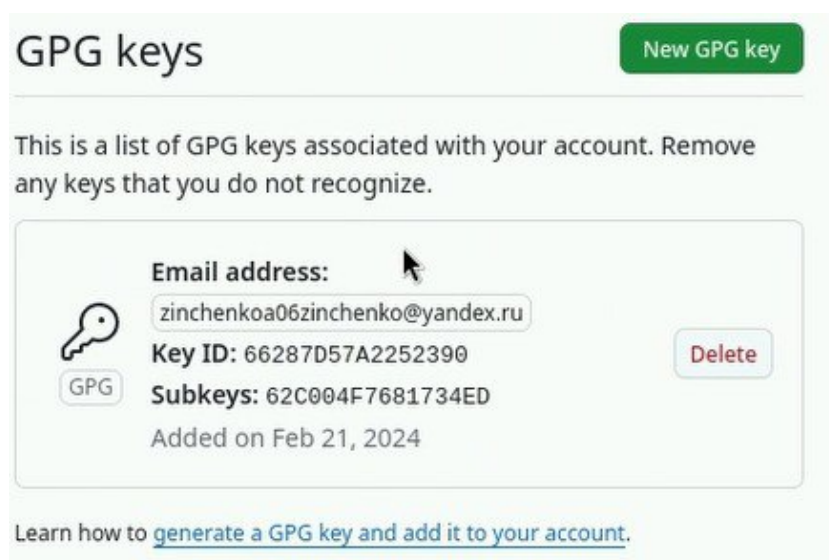


Рис. 3.13: Список ключей

Используя `zinchenkoa06zinchenko@yandex.ru`, указала Git применять его при подписи коммитов с помощью команд `git config --global user.signingkey git config --global commit.gpgsign true git config --global gpg.program $(which gpg2)` (рис. 3.14).

```
[arzinchenko@arzinchenko ~]$ git config --global user.signingkey 66287D57A2252390
[arzinchenko@arzinchenko ~]$ git config --global commit.gpgsign true
[arzinchenko@arzinchenko ~]$ git config --global gpg.program $(which gpg2)
[arzinchenko@arzinchenko ~]$
```

Рис. 3.14: Подписи коммитов

Я авторизовалась с помощью команды `gh auth login` (рис. 3.15).

```
[arzinchenko@arzinchenko ~]$ gh auth login
? What account do you want to log into? GitHub.com
? What is your preferred protocol for Git operations on this host? HTTPS
? Authenticate Git with your GitHub credentials? Yes
? How would you like to authenticate GitHub CLI? Login with a web browser

! First copy your one-time code: 5ED3-9851
Press Enter to open github.com in your browser...
✓ Authentication complete.
- gh config set -h github.com git_protocol https
```

Рис. 3.15: Авторизация

Я создала репозиторий с помощью команд `mkdir -p ~/work/study/2022-2023/“Операционные системы”` `cd ~/work/study/2022-2023/“Операционные`

системы” gh repo create study_2022-2023_os-intro --template=yamadharm/course-directory-student-template --public git clone --recursive git@github.com:/study_2022-2023_os-intro.git os-intro (рис. 3.16).

```
[arzinchenko@arzinchenko ~]$ mkdir -p ~/work/study/2023-2024/"Операционные системы"
[arzinchenko@arzinchenko ~]$ cd ~/work/study/2023-2024/"Операционные системы"
bash: cd: /home/arzinchenko/work/study/2023-2024/Операционные системы: Нет такого файла или каталога
[arzinchenko@arzinchenko ~]$ mkdir -p ~/work/study/2023-2024/"Операционные системы"
[arzinchenko@arzinchenko ~]$ cd ~/work/study/2023-2024/"Операционные системы"
[arzinchenko@arzinchenko Операционные системы]$ gh repo create study_2023-2024_os-intro --template=yamadharm/course-directory-student-template --public
[arzinchenko@arzinchenko Операционные системы]$ gh repo create study_2023-2024_os-intro --template=yamadharm/course-directory-student-template --public
[arzinchenko@arzinchenko Операционные системы]$ git clone --recursive git@github.com:Anastasia-zinchenko/study_2023-2024_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:Di3wvV6TujJhdpZisF/zLDABzPMsvHdk4UvCOqU.
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.
git@github.com: Permission denied (publickey).
fatal: Не удалось прочитать из внешнего репозитория.

Удостоверьтесь, что у вас есть необходимые права доступа
и репозиторий существует.
[arzinchenko@arzinchenko Операционные системы]$
```

Рис. 3.16: Создание репозитория

Для того чтобы настроить каталог курса я перешла в каталог курса с помощью команды `cd ~/work/study/2022-2023/"Операционные системы"/os-intro` (рис. 3.17).

```
[arzinchenko@arzinchenko Операционные системы]$ cd ~/work/study/2023-2024/"Операционные системы"/os-intro
```

Рис. 3.17: Создание репозитория

Удалила лишние файлы с помощью команды `rm package.json` (рис. 3.18).

```
[arzinchenko@arzinchenko os-intro]$ rm package.json
```

Рис. 3.18: Удаление лишних файлов

Создала необходимые каталоги с помощью команд `echo os-intro > COURSE` `make` (рис. 3.19).

```
[arzinchenko@arzinchenko os-intro]$ echo os-intro > COURSE
[arzinchenko@arzinchenko os-intro]$ ls
CHANGELOG.md  config  COURSE  LICENSE  Makefile  README.en.md  README.git-flow.md  README.md  template
[arzinchenko@arzinchenko os-intro]$ make
Usage:
  make <target>

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

Рис. 3.19: Создание каталогов

Отправила файлы на сервер с помощью команд `git add .` `git commit -am 'feat(main): make course structure'` `git push` (рис. 3.20).

```
*** Пожалуйста, скажите мне кто вы есть.

Запустите

git config --global user.email "you@example.com"
git config --global user.name "Ваше Имя"

для указания идентификационных данных аккаунта по умолчанию.
Пропустите параметр --global для указания данных только для этого репозитория.

fatal: не удалось выполнить автоопределение адреса электронной почты (получено «arzinchenko@arzinchenko.(none)»)
[arzinchenko@arzinchenko os-intro]$ git config --global user.email emd06zinchenko@yandex.ru
[arzinchenko@arzinchenko os-intro]$ git config --global user.name "zinkenkoa06zinchenko@yandex.ru"
[arzinchenko@arzinchenko os-intro]$ git commit -am 'feat(main): make course structure'
Author identity unknown

*** Пожалуйста, скажите мне кто вы есть.

Запустите

git config --global user.email "you@example.com"
git config --global user.name "Ваше Имя"

для указания идентификационных данных аккаунта по умолчанию.
Пропустите параметр --global для указания данных только для этого репозитория.

fatal: не удалось выполнить автоопределение адреса электронной почты (получено «arzinchenko@arzinchenko.(none)»)
[arzinchenko@arzinchenko os-intro]$ git config --global user.email emd06zinchenko@yandex.ru
[arzinchenko@arzinchenko os-intro]$ git config --global user.name "zinkenkoa06zinchenko@yandex.ru"
[arzinchenko@arzinchenko os-intro]$ git commit -am 'feat(main): make course structure'
(master 9cdf8e8) feat(main): make course structure
2 files changed, 1 insertion(+), 14 deletions(-)
delete mode 100644 package.json
[arzinchenko@arzinchenko os-intro]$ git push
Перечисление объектов: 5, готово.
Подсчет объектов: 100% (5/5), готово.
Сжатие объектов: 100% (2/2), готово.
Запись объектов: 100% (3/3), 957 байтов | 957.00 КиБ/с, готово.
Всего 3 (изменений 1), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:Anastasia-zinchenko/study_2023-2024_os-intro.git
beae54e..9cdf8e8 master -> master
[arzinchenko@arzinchenko os-intro]$
```

Рис. 3.20: Создание каталогов

Проверка файлов на сервере (рис. 3.21).

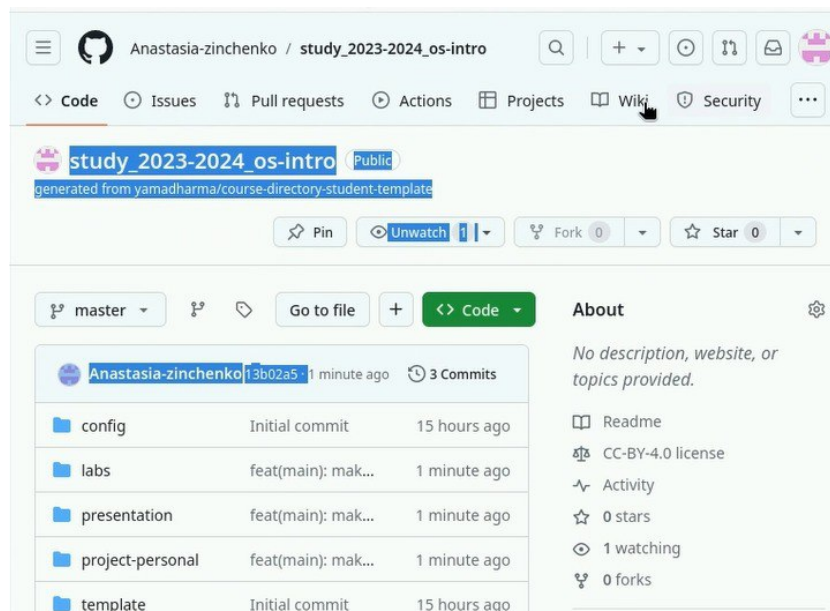


Рис. 3.21: Проверка

Контрольные вопросы

1. Что такое системы контроля версий (VCS) и для решения каких задач они предназначаются? Это программное обеспечение для облегчения работы с информацией, которая изменяется. Они позволяют хранить несколько версий одного и того же документа, при необходимости возвращаться к более ранним версиям, а также определять кто и когда сделал какие - либо изменения.
2. Объясните следующие понятия VCS и их отношения: хранилище, commit, история, рабочая копия. Хранилище - место хранения всех версий и служебной информации. Commit - процесс создания новой версии. История - место, где сохраняются все коммиты, по которым можно посмотреть данные о коммитах. Рабочая копия - текущее состояние файлов проекта, основанное на версии, загруженной из хранилища.
3. Что представляют собой и чем отличаются централизованные и децентрализованные VCS? Приведите примеры VCS каждого вида. Централизованные VCS: одно основное хранилище всего проекта и каждый пользователь копирует себе необходимые ему файлы из этого репозитория, изменяет, а затем добавляет свои изменения обратно. Децентрализованные VCS: у каждого пользователя свой вариант репозитория.
4. Опишите действия с VCS при единоличной работе с хранилищем.
5. Опишите порядок работы с общим хранилищем VCS.
6. Каковы основные задачи, решаемые инструментальным средством git? 1. Хранение информации о всех изменениях в коде. 2. Обеспечение удобства работы над проектом в команде.
7. Назовите и дайте краткую характеристику командам git. 1. git-version (проверка версии Git). 2. git init - инициализация текущего рабочего каталога как Git репозиторий. 3. git clone - копирование существующего удаленного Git - репозитория. 4. git remote - просмотр списка текущих удаленных репозиторий. 5. git commit -am "Commit message" - сжатие всех индексированных

- файлов и отправка коммитов. 6. `git branch` - просмотр списка текущих веток.
8. Приведите примеры использования при работе с локальным и удалённым репозиториями.
 9. Что такое и зачем могут быть нужны ветви (branches)? Ветви нужны для того, чтобы программисты могли одновременно работать над одним и тем же файлом, не мешая друг другу.
 10. Как и зачем можно игнорировать некоторые файлы при `commit`? Игнорируемые файлы - артефакты сборки и файлы, генерируемые машиной из исходных файлов в репозитории, либо файлы, которые не должны попадать в коммиты.

4 Выводы

Я изучила идеологию и применение средств контроля версий и освоила умения по работе с git.

Список литературы