

# Лабораторная работа №3 по предмету Операционные системы

Группа НПМбв-02-19

Нечаева Виктория Алексеевна

# Содержание

Цель работы	5
Задание	6
Выполнение лабораторной работы	7
Настройка github . . . . .	7
Установка программного обеспечения . . . . .	7
Базовая настройка git . . . . .	10
Создайте ключи ssh . . . . .	11
Создайте ключи pgr . . . . .	12
Добавление PGP ключа в GitHub . . . . .	14
Настройка автоматических подписей коммитов git . . . . .	16
Создание репозитория курса на основе шаблона . . . . .	16
Настройка каталога курса . . . . .	18
Выводы	21
Контрольные вопросы	22

## Список таблиц

## Список иллюстраций

1	Рисунок 1. Настроенный аккаунт github . . . . .	7
2	Рисунок 2. Установка git-flow . . . . .	8
3	Рисунок 3. Установка gh . . . . .	9
4	Рисунок 4. Установка gh . . . . .	10
5	Рисунок 5. Настройка git . . . . .	11
6	Рисунок 6. Создание ключей ssh . . . . .	12
7	Рисунок 7. Генерация и настройка ключа pgr . . . . .	13
8	Рисунок 8. Сформированный pgr ключ . . . . .	14
9	Рисунок 9. Вывод списка ключей . . . . .	14
10	Рисунок 10. Ввод pgr ключа в буфер . . . . .	15
11	Рисунок 11. GPG ключ в гитхабе . . . . .	15
12	Рисунок 12. GPG ключ в гитхабе . . . . .	16
13	Рисунок 13. Напоминание об авторизации . . . . .	17
14	Рисунок 14. Добавление ключа ssh . . . . .	17
15	Рисунок 15. Авторизация в github в консоли по токenu . . . . .	17
16	Рисунок 16. Клонировем репозиторий к себе на гитхаб . . . . .	18
17	Рисунок 18. git add, git commit . . . . .	20
18	Рисунок 19. git push . . . . .	20

## Цель работы

Целью данной работы является изучение идеологии и методов применения средств контроля версий и освоения умения по работе с git.

# 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.

Лабораторная работа выполняется в Ubuntu.

# Выполнение лабораторной работы

## Настройка github

1. Создайте учётную запись на <https://github.com>.
2. Заполните основные данные на <https://github.com>. На (рис. 1) учетная запись уже после выполнения этой лабораторной работы. Репозитория два, так как аккаунт создан давно и есть один приватный репозиторий.

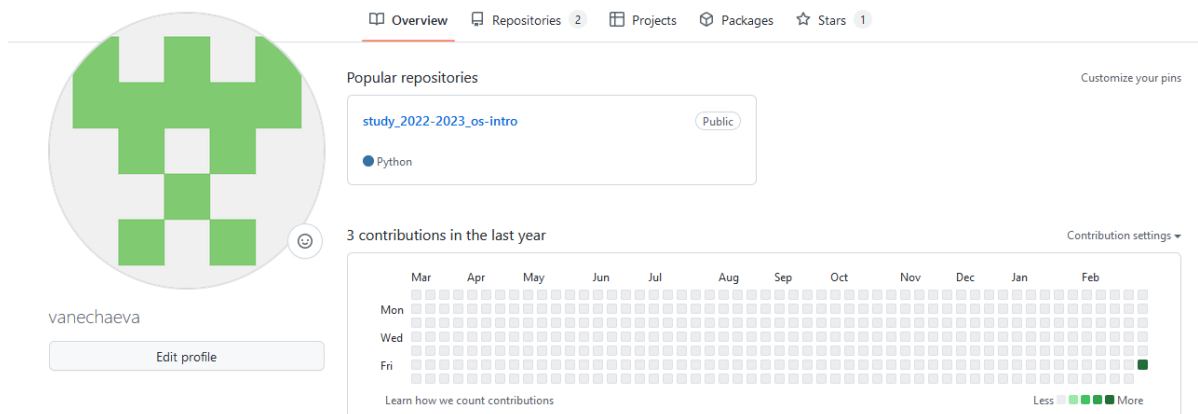


Рис. 1: Рисунок 1. Настроенный аккаунт github

## Установка программного обеспечения

Установка git-flow в Ubuntu (рис. 2)

```
Обзор Терминал Пт, 10 марта 20:02
vanechaeva@vanechaeva: ~

vanechaeva@vanechaeva:~$ sudo apt update
[sudo] пароль для vanechaeva:
Сущ:1 http://ru.archive.ubuntu.com/ubuntu jammy InRelease
Пол:2 http://ru.archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
Пол:3 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Пол:4 http://ru.archive.ubuntu.com/ubuntu jammy-backports InRelease [107 kB]
Пол:5 http://ru.archive.ubuntu.com/ubuntu jammy/main Translation-ru [344 kB]
Пол:6 http://security.ubuntu.com/ubuntu jammy-security/main i386 Packages [270 kB]
Пол:7 http://ru.archive.ubuntu.com/ubuntu jammy/restricted Translation-ru [896 B]
Пол:8 http://ru.archive.ubuntu.com/ubuntu jammy/universe Translation-ru [1 369 kB]
Пол:9 http://security.ubuntu.com/ubuntu jammy-security/main amd64 Packages [687 kB]
Пол:10 http://security.ubuntu.com/ubuntu jammy-security/main Translation-en [141 kB]
Пол:11 http://security.ubuntu.com/ubuntu jammy-security/main amd64 DEP-11 Metadata [41,4 kB]
Пол:12 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 DEP-11 Metadata [17,0 kB]
Пол:13 http://ru.archive.ubuntu.com/ubuntu jammy/multiverse Translation-ru [68,5 kB]
Пол:14 http://ru.archive.ubuntu.com/ubuntu jammy-updates/main amd64 DEP-11 Metadata [101 kB]
Пол:15 http://ru.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 DEP-11 Metadata [269 kB]
Пол:16 http://ru.archive.ubuntu.com/ubuntu jammy-updates/multiverse amd64 DEP-11 Metadata [940 B]
Пол:17 http://ru.archive.ubuntu.com/ubuntu jammy-backports/main amd64 DEP-11 Metadata [8 004 B]
Пол:18 http://ru.archive.ubuntu.com/ubuntu jammy-backports/universe amd64 DEP-11 Metadata [12,4 kB]
Получено 3 666 kB за 24с (150 kB/s)
Чтение списков пакетов... Готово
Построение дерева зависимостей... Готово
Чтение информации о состоянии... Готово
Может быть обновлено 44 пакета. Запустите «apt list --upgradable» для их показа.
vanechaeva@vanechaeva:~$ sudo apt install git-flow
Чтение списков пакетов... Готово
Построение дерева зависимостей... Готово
Чтение информации о состоянии... Готово
Будут установлены следующие дополнительные пакеты:
  git git-man liberror-perl
Предлагаемые пакеты:
  git-daemon-run | git-daemon-sysvinit git-doc git-email git-gui gitk gitweb git-cvs
  git-mediawiki git-svn
Следующие НОВЫЕ пакеты будут установлены:
  git git-flow git-man liberror-perl
Обновлено 0 пакетов, установлено 4 новых пакетов, для удаления отмечено 0 пакетов, и 44 пакетов не обновлено.
Необходимо скачать 4 159 kB архивов.
После данной операции объём занятого дискового пространства возрастёт на 21,2 MB.
Хотите продолжить? [д/н] y
Пол:1 http://ru.archive.ubuntu.com/ubuntu jammy/main amd64 liberror-perl all 0.17029-1 [26,5 kB]
Пол:2 http://ru.archive.ubuntu.com/ubuntu jammy-updates/main amd64 git-man all 1:2.34.1-1ubuntu1.8 [953 kB]
Пол:3 http://ru.archive.ubuntu.com/ubuntu jammy-updates/main amd64 git amd64 1:2.34.1-1ubuntu1.8 [3 141 kB]
Пол:4 http://ru.archive.ubuntu.com/ubuntu jammy/universe amd64 git-flow all 1.12.3-3 [38,6 kB]
Получено 4 159 kB за 10с (402 kB/s)
Выбор ранее не выбранного пакета liberror-perl.
(Чтение базы данных ... на данный момент установлено 177892 файла и каталога.)
Подготовка к распаковке .../liberror-perl_0.17029-1_all.deb ...
```

Рис. 2: Рисунок 2. Установка git-flow

Установка gh в Ubuntu (рис.3, рис.4)



```

vanechaeva@vanechaeva:~$ type -p curl >/dev/null || sudo apt install curl -y
Чтение списков пакетов... Готово
Построение дерева зависимостей... Готово
Чтение информации о состоянии... Готово
Будут установлены следующие дополнительные пакеты:
  libcurl4
Следующие НОВЫЕ пакеты будут установлены:
  curl
Следующие пакеты будут обновлены:
  libcurl4
Обновлено 1 пакетов, установлено 1 новых пакетов, для удаления отмечено 0 пакетов, и 43 пакетов не обновлено.
Необходимо скачать 484 kB архивов.
После данной операции объём занятого дискового пространства возрастёт на 455 kB.
Пол:1 http://ru.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libcurl4 amd64 7.81.0-1ubuntu1.8 [290 kB]
Пол:2 http://ru.archive.ubuntu.com/ubuntu jammy-updates/main amd64 curl amd64 7.81.0-1ubuntu1.8 [194 kB]
Получено 484 kB за 1с (469 kB/s)
(Чтение базы данных ... на данный момент установлено 178942 файла и каталога.)
Подготовка к распаковке .../libcurl4_7.81.0-1ubuntu1.8_amd64.deb ...
Распаковывается libcurl4:amd64 (7.81.0-1ubuntu1.8) на замену (7.81.0-1ubuntu1.7) ...
Выбор ранее не выбранного пакета curl.
Подготовка к распаковке .../curl_7.81.0-1ubuntu1.8_amd64.deb ...
Распаковывается curl (7.81.0-1ubuntu1.8) ...
Настраивается пакет libcurl4:amd64 (7.81.0-1ubuntu1.8) ...
Настраивается пакет curl (7.81.0-1ubuntu1.8) ...
Обрабатываются триггеры для man-db (2.10.2-1) ...
Обрабатываются триггеры для libc-bin (2.35-0ubuntu3.1) ...
vanechaeva@vanechaeva:~$ curl -fsSL https://cli.github.com/packages/githubcli-archive-keyring.gpg | sudo dd of=/usr/share/keyrings/g
ithubcli-archive-keyring.gpg \
&& sudo chmod go+r /usr/share/keyrings/githubcli-archive-keyring.gpg \
&& echo "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/githubcli-archive-keyring.gpg] https://cli.github.com/
packages/stable/main" | sudo tee /etc/apt/sources.list.d/github-cli.list > /dev/null \
&& sudo apt update \
&& sudo apt install gh -y
4+1 записей получено
4+1 записей отправлено

```

Рис. 3: Рисунок 3. Установка gh

```

2270 байт (2,3 kB, 2,2 KiB) скопирован, 0,658552 s, 3,4 kB/s
Суц:1 http://ru.archive.ubuntu.com/ubuntu jammy InRelease
Суц:2 http://ru.archive.ubuntu.com/ubuntu jammy-updates InRelease
Суц:3 http://ru.archive.ubuntu.com/ubuntu jammy-backports InRelease
Суц:4 http://security.ubuntu.com/ubuntu jammy-security InRelease
Пол:5 https://cli.github.com/packages stable InRelease [3 917 B]
Пол:6 https://cli.github.com/packages stable/main amd64 Packages [346 B]
Получено 4 263 B за 1с (4 365 B/s)
Чтение списков пакетов... Готово
Построение дерева зависимостей... Готово
Чтение информации о состоянии... Готово
Может быть обновлено 43 пакета. Запустите «apt list --upgradable» для их показа.
Чтение списков пакетов... Готово
Построение дерева зависимостей... Готово
Чтение информации о состоянии... Готово
Следующие НОВЫЕ пакеты будут установлены:
  gh
Обновлено 0 пакетов, установлено 1 новых пакетов, для удаления отмечено 0 пакетов, и 43 пакетов не обновлено.
Необходимо скачать 10,6 МБ архивов.
После данной операции объем занятого дискового пространства возрастёт на 40,7 МБ.
Пол:1 https://cli.github.com/packages stable/main amd64 gh amd64 2.24.3 [10,6 MB]
Получено 10,6 МБ за 11с (993 kB/s)
Выбор ранее не выбранного пакета gh.
(Чтение базы данных ... на данный момент установлено 178949 файлов и каталогов.)
Подготовка к распаковке ../archives/gh_2.24.3_amd64.deb ...
Распаковывается gh (2.24.3) ...
Настраивается пакет gh (2.24.3) ...
Обрабатываются триггеры для man-db (2.10.2-1) ...
vanechaeva@vanechaeva:~$ sudo apt update
Суц:1 http://ru.archive.ubuntu.com/ubuntu jammy InRelease
Суц:2 http://ru.archive.ubuntu.com/ubuntu jammy-updates InRelease
Суц:3 http://ru.archive.ubuntu.com/ubuntu jammy-backports InRelease
Суц:4 http://security.ubuntu.com/ubuntu jammy-security InRelease
Суц:5 https://cli.github.com/packages stable InRelease
Чтение списков пакетов... Готово
Построение дерева зависимостей... Готово
Чтение информации о состоянии... Готово
Может быть обновлено 43 пакета. Запустите «apt list --upgradable» для их показа.
vanechaeva@vanechaeva:~$ sudo apt install gh
Чтение списков пакетов... Готово
Построение дерева зависимостей... Готово

```

Рис. 4: Рисунок 4. Установка gh

## Базовая настройка git

Рисунок 5.

– Зададим имя и email владельца репозитория:

git config --global user.name "Name Surname"

git config --global user.email "work@mail"

– Настроим utf-8 в выводе сообщений git: git config --global core.quotePath false

– Настройте верификацию и подписание коммитов git.

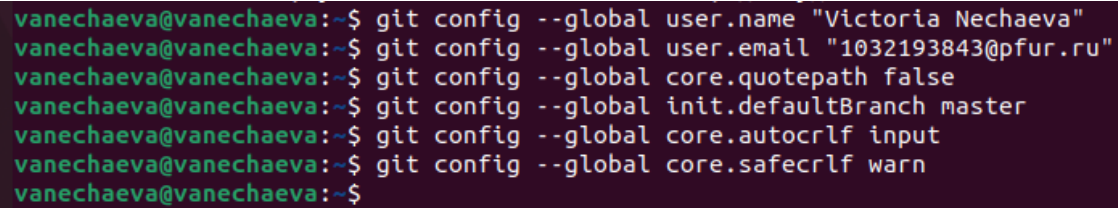
– Зададим имя начальной ветки (будем называть её master): 30 Лабораторная

работа No 2. Управление версиями

git config --global init.defaultBranch master – Параметр autocrlf:

git config --global core.autocrlf input – Параметр safecrlf:

git config --global core.safecrlf warn

A screenshot of a terminal window with a dark background and light-colored text. It shows a series of git configuration commands being entered at a prompt. The commands are: 'git config --global user.name "Victoria Nechaeva"', 'git config --global user.email "1032193843@pfur.ru"', 'git config --global core.quotepath false', 'git config --global init.defaultBranch master', 'git config --global core.autocrlf input', and 'git config --global core.safecrlf warn'. Each command is preceded by the prompt 'vanechaeva@vanechaeva:~\$'.

```
vanechaeva@vanechaeva:~$ git config --global user.name "Victoria Nechaeva"
vanechaeva@vanechaeva:~$ git config --global user.email "1032193843@pfur.ru"
vanechaeva@vanechaeva:~$ git config --global core.quotepath false
vanechaeva@vanechaeva:~$ git config --global init.defaultBranch master
vanechaeva@vanechaeva:~$ git config --global core.autocrlf input
vanechaeva@vanechaeva:~$ git config --global core.safecrlf warn
vanechaeva@vanechaeva:~$
```

Рис. 5: Рисунок 5. Настройка git

## Создайте ключи ssh

Рисунок 6.

– по алгоритму rsa с ключём размером 4096 бит:

ssh-keygen -t rsa -b 4096

– по алгоритму ed25519:

ssh-keygen -t ed25519

```
vanechaeva@vanechaeva:~$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/vanechaeva/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/vanechaeva/.ssh/id_rsa
Your public key has been saved in /home/vanechaeva/.ssh/id_rsa.pub
The key fingerprint is:
1b:06:8c:1f:1c:4a:87:1f:1c:4a:87:1f:1c:4a:87:1f
The key's randomart image is:
+---[RSA 4096]-----+
|          +o+o*+ |
|          +oOo=.B|
|          ..*=* B*|
|          E+ *.+*|
|          . S o == =|
|          o . +.= o|
|          . ....|
|          ..|
|          ..|
+-----[SHA256]-----+
vanechaeva@vanechaeva:~$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/vanechaeva/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/vanechaeva/.ssh/id_ed25519
Your public key has been saved in /home/vanechaeva/.ssh/id_ed25519.pub
The key fingerprint is:
1b:06:8c:1f:1c:4a:87:1f:1c:4a:87:1f:1c:4a:87:1f
The key's randomart image is:
+--[ED25519 256]--+
|..o+BX0o. |
|o+++=Eo+ |
|=o=o0.o |
|o+.= * |
|.o = + S |
|. .o + |
|. . . |
|. + . |
|.+++. |
+-----[SHA256]-----+
```

Рис. 6: Рисунок 6. Создание ключей ssh

## Создайте ключи gpg

Рисунки 7 и 8. – Генерируем ключ `gpg --full-generate-key`

Из предложенных опций выбираем:

- тип RSA and RSA;
- размер 4096;
- выберите срок действия; значение по умолчанию — 0 (срок действия не истекает никогда).

GPG запросит личную информацию, которая сохранится в ключе:

- Имя (не менее 5 символов).
- Адрес электронной почты.
- При вводе email убедитесь, что он соответствует адресу, используемому на GitHub.
- Комментарий. Можно ввести что угодно или нажать клавишу ввода, чтобы оставить это поле пустым.

```
vanechaeva@vanechaeva:~$ gpg --full-generate-key
gpg (GnuPG) 2.2.27; Copyright (C) 2021 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Выберите тип ключа:
  (1) RSA и RSA (по умолчанию)
  (2) DSA и Elgamal
  (3) DSA (только для подписи)
  (4) RSA (только для подписи)
  (14) Имеющийся на карте ключ
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: Victoria Nechaeva
Адрес электронной почты: mckinleey@yandex.ru
Примечание:
Вы выбрали следующий идентификатор пользователя:
  "Victoria Nechaeva <mckinleey@yandex.ru>"

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: ключ _____ помечен как абсолютно доверенный
gpg: создан каталог '/home/vanechaeva/.gnupg/openpgp-revocs.d'
gpg: сертификат отзыва записан в '/home/vanechaeva/.gnupg/openpgp-revocs.d/78F0AF7F69F7F98F7786072F_____.rev'.
открытый и секретный ключи созданы и подписаны.
```

Рис. 7: Рисунок 7. Генерация и настройка ключа gpg

```
pub  rsa4096 2023-03-10 [SC]
    78F0AF7F69F7F98F7786072F
uid                               Victoria Nechaeva <mckinleey@yandex.ru>
sub  rsa4096 2023-03-10 [E]
```

Рис. 8: Рисунок 8. Сформированный pgr ключ

## Добавление PGP ключа в GitHub

- Выводим список ключей и копируем отпечаток приватного ключа: (рис.9)

`gpg --list-secret-keys --keyid-format LONG`

```
vanechaeva@vanechaeva:~$ 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
/home/vanechaeva/.gnupg/pubring.kbx
-----
sec  rsa4096/171FACCD8A1C6260 2023-03-10 [SC]
    78F0AF7F69F7F98F7786072F
uid      [ абсолютно ] Victoria Nechaeva <mckinleey@yandex.ru>
ssb  rsa4096/ 2023-03-10 [E]
```

Рис. 9: Рисунок 9. Вывод списка ключей

- Формат строки:

sec Алгоритм/Отпечаток\_ключа Дата\_создания [Флаги] [Годеи\_до]ID\_ключа

- Скопируйте ваш сгенерированный PGP ключ в буфер обмена:

`gpg --armor --export | xclip -sel clip`

Указываю сначала с кавычками-скобочками. Исправляю.

```

vanechaeva@vanechaeva:~$ gpg --armor --export <[redacted]> | xclip -sel clip
bash: синтаксическая ошибка рядом с неожиданным маркером «|»
vanechaeva@vanechaeva:~$ gpg --armor --export [redacted] | xclip -sel clip
Команда «xclip» не найдена, но может быть установлена с помощью:
sudo apt install xclip
vanechaeva@vanechaeva:~$ sudo apt install xclip
[sudo] пароль для vanechaeva:
Чтение списков пакетов... Готово
Построение дерева зависимостей... Готово
Чтение информации о состоянии... Готово
Следующие НОВЫЕ пакеты будут установлены:
  xclip
Обновлено 0 пакетов, установлено 1 новых пакетов, для удаления отмечено 0 пакетов, и 43 пакетов не обновлено.
Необходимо скачать 18,3 кВ архивов.
После данной операции объем занятого дискового пространства возрастёт на 60,4 кВ.
Пол:1 http://ru.archive.ubuntu.com/ubuntu jammy/universe amd64 xclip amd64 0.13-2 [18,3 кВ]
Получено 18,3 кВ за 3с (6 061 В/с)
Выбор ранее не выбранного пакета xclip.
(Чтение базы данных ... на данный момент установлено 179096 файлов и каталогов.)
Подготовка к распаковке ./xclip_0.13-2_amd64.deb ...
Распаковывается xclip (0.13-2) ...
Настраивается пакет xclip (0.13-2) ...
Обрабатываются триггеры для man-db (2.10.2-1) ...
vanechaeva@vanechaeva:~$ gpg --armor --export [redacted] | xclip -sel clip

```

Рис. 10: Рисунок 10. Ввод gpg ключа в буфер

– Перейдите в настройки GitHub (<https://github.com/settings/keys>), нажмите на кнопку New GPG key и вставьте полученный ключ в поле ввода. (рис. 11)

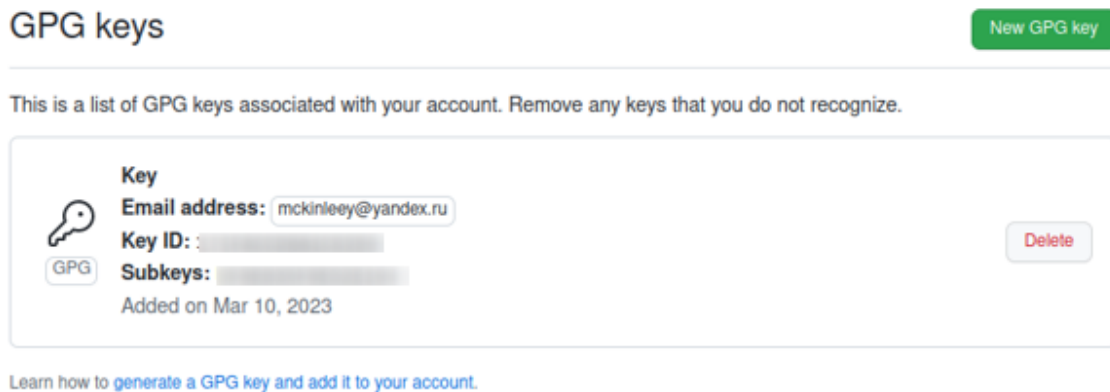
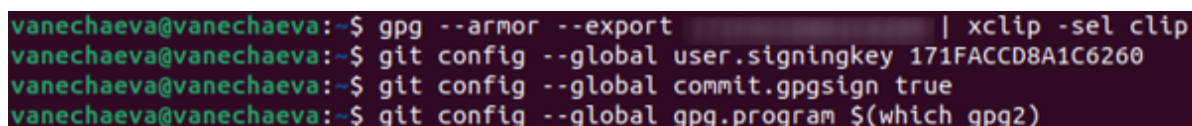


Рис. 11: Рисунок 11. GPG ключ в гитхабе

## Настройка автоматических подписей коммитов git

– Используя введенный email, укажите Git применять его при подписи коммитов (рис. 12):

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



```
vanechaeva@vanechaeva:~$ gpg --armor --export [redacted] | xclip -sel clip  
vanechaeva@vanechaeva:~$ git config --global user.signingkey 171FACCD8A1C6260  
vanechaeva@vanechaeva:~$ git config --global commit.gpgsign true  
vanechaeva@vanechaeva:~$ git config --global gpg.program $(which gpg2)
```

Рис. 12: Рисунок 12. GPG ключ в гитхабе

## Создание репозитория курса на основе шаблона

Шаблон для рабочего пространства

- Репозиторий: <https://github.com/yamadharma/course-directory-student-template>.
- Необходимо создать шаблон рабочего пространства.
- Например, для 2022–2023 учебного года и предмета «Операционные системы» (код предмета os-intro) создание репозитория примет следующий вид:

```
mkdir -p ~/work/study/2021-2022/“Операционные системы”  
cd ~/work/study/2021-2022/“Операционные системы”  
gh repo create study_2021-2022_os-intro-template=yamadharmacourse-directory-  
student-template --public  
git clone --recursive git@github.com:/study_2021-2022_os-intro.git os-intro
```

При создании репозитория нам напоминают, что надо авторизоваться (рис.13). Для этого добавляем SSH ключ (рис.14) в аккаунте гитхаба, через `gh auth login` авторизовываемся:

Выбираем `github.com`, `SSH`, `Skip`, `Paste an authentication token` (формируем в гх),



вставляем его в консоли. (рис. 15) Завершаем создание и клонирование репозитория (рис.16)

```
vanechaeva@vanechaeva:~$ mkdir -p ~/work/study/2022-2023/"Операционные системы"
vanechaeva@vanechaeva:~$ ls
Desktop Documents Downloads Music Pictures Public snap Templates Videos work
vanechaeva@vanechaeva:~$ cd work
vanechaeva@vanechaeva:~/work$ cd study/2022-2023
vanechaeva@vanechaeva:~/work/study/2022-2023$ ls
"Операционные системы"
vanechaeva@vanechaeva:~/work/study/2022-2023$ cd "Операционные системы"
vanechaeva@vanechaeva:~/work/study/2022-2023/Операционные системы$ gh repo create study_2022-2023_os-intro --template=yamadharma/course-directory-student-template --public
To get started with GitHub CLI, please run:  gh auth login
Alternatively, populate the GH_TOKEN environment variable with a GitHub API authentication token.
```


Рис. 13: Рисунок 13. Напоминание об авторизации

## SSH keys

[New SSH key](#)

This is a list of SSH keys associated with your account. Remove any keys that you do not recognize.

### Authentication Keys

**SSH**

Added on Mar 10, 2023  
Last used within the last week — Read/write

Delete

Check out our guide to [generating SSH keys](#) or troubleshoot [common SSH problems](#).

Рис. 14: Рисунок 14. Добавление ключа ssh

```
vanechaeva@vanechaeva:~/work/study/2022-2023/Операционные системы$ gh auth login
? What account do you want to log into? GitHub.com
? What is your preferred protocol for Git operations? SSH
? Upload your SSH public key to your GitHub account? Skip
? 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'.
? Paste your authentication token: *****
- gh config set -h github.com git_protocol ssh
✓ Configured git protocol
✓ Logged in as vanechaeva
```

Рис. 15: Рисунок 15. Авторизация в github в консоли по токenu

```

vanechaeva@vanechaeva:~/work/study/2022-2023/Операционные системы$ git clone --recursive git@github.com:vanechaeva/study_2022-2023_o
s-intro.git os-intro
Клонирование в «os-intro»...
remote: Enumerating objects: 27, done.
remote: Counting objects: 100% (27/27), done.
remote: Compressing objects: 100% (26/26), done.
remote: Total 27 (delta 1), reused 11 (delta 0), pack-reused 0
Получение объектов: 100% (27/27), 16.93 КиБ | 178.00 КиБ/с, готово.
Определение изменений: 100% (1/1), готово.
Подмодуль «template/presentation» (https://github.com/yamadharma/academic-presentation-markdown-template.git) зарегистрирован по пут
и «templat
e/report»
Клонирование в «/home/vanechaeva/work/study/2022-2023/Операционные системы/os-intro/template/presentation»...
remote: Enumerating objects: 82, done.
remote: Counting objects: 100% (82/82), done.
remote: Compressing objects: 100% (57/57), done.
remote: Total 82 (delta 28), reused 77 (delta 23), pack-reused 0
Получение объектов: 100% (82/82), 92.90 КиБ | 580.00 КиБ/с, готово.
Определение изменений: 100% (28/28), готово.
Клонирование в «/home/vanechaeva/work/study/2022-2023/Операционные системы/os-intro/template/report»...
remote: Enumerating objects: 101, done.
remote: Counting objects: 100% (101/101), done.
remote: Compressing objects: 100% (70/70), done.
remote: Total 101 (delta 40), reused 88 (delta 27), pack-reused 0
Получение объектов: 100% (101/101), 327.25 КиБ | 18.00 КиБ/с, готово.
Определение изменений: 100% (40/40), готово.
Submodule path 'template/presentation': checked out 'b1be3800ee91f5809264cb755d316174540b753e'
Submodule path 'template/report': checked out '1d1b61dcac9c287a83917b82e3aef11a33b1e3b2'

```

Рис. 16: Рисунок 16. Клонирование репозитория к себе на гитхаб

## Настройка каталога курса

Рисунок 17.

– Перейдите в каталог курса:

```
cd ~/work/study/2021-2022/“Операционные системы”/os-intro
```

– Удалите лишние файлы:

```
rm package.json
```

– Создайте необходимые каталоги:

Лабораторная работа No 2. Управление версиями

```
make COURSE=os-intro
```

```

vanechaeva@vanechaeva:~/work/study/2022-2023/Операционные системы$ cd ~/work/study/2022-2023/"Операционные системы"/os-intro
vanechaeva@vanechaeva:~/work/study/2022-2023/Операционные системы/os-intro$ rm package.json
vanechaeva@vanechaeva:~/work/study/2022-2023/Операционные системы/os-intro$ make COURSE=os-intro
Команда «make» не найдена, но может быть установлена с помощью:
sudo apt install make # version 4.3-4.1build1, or
sudo apt install make-guile # version 4.3-4.1build1
vanechaeva@vanechaeva:~/work/study/2022-2023/Операционные системы/os-intro$ sudo apt install make
[sudo] пароль для vanechaeva:
Чтение списков пакетов... Готово
Построение дерева зависимостей... Готово
Чтение информации о состоянии... Готово
Предлагаемые пакеты:
  make-doc
Следующие НОВЫЕ пакеты будут установлены:
  make
Обновлено 0 пакетов, установлено 1 новых пакетов, для удаления отмечено 0 пакетов, и 43 пакетов не обновлено.
Необходимо скачать 180 kB архивов.
После данной операции объем занятого дискового пространства возрастет на 426 kB.
Пол:1 http://ru.archive.ubuntu.com/ubuntu jammy/main amd64 make amd64 4.3-4.1build1 [180 kB]
Получено 180 kB за 0с (559 kB/s)
Выбор ранее не выбранного пакета make.
(Чтение базы данных ... на данный момент установлено 179108 файлов и каталогов.)
Подготовка к распаковке .../make_4.3-4.1build1_amd64.deb ...
Распаковывается make (4.3-4.1build1) ...
Настраивается пакет make (4.3-4.1build1) ...
Обрабатываются триггеры для man-db (2.10.2-1) ...
vanechaeva@vanechaeva:~/work/study/2022-2023/Операционные системы/os-intro$ make COURSE=os-intro

```

– Отправьте файлы на сервер (рис. 16, рис.17):

git add .

git commit -am 'feat(main): make course structure'

git push

```

vanechaeva@vanechaeva:~/work/study/2022-2023/Операционные системы/os-intro$ git add
Ничего не проиндексировано.
подсказка: Возможно вы хотели сделать «git add .»?
подсказка: Можно отключить это сообщение командой
подсказка: «git config advice.addEmptyPathsSpec false»
vanechaeva@vanechaeva:~/work/study/2022-2023/Операционные системы/os-intro$ git add .
vanechaeva@vanechaeva:~/work/study/2022-2023/Операционные системы/os-intro$ git commit -am 'feat(main): make course structure'
[master 37cae3] feat(main): make course structure
360 files changed, 100326 insertions(+), 14 deletions(-)
create mode 100644 labs/README.md
create mode 100644 labs/README.ru.md
create mode 100644 labs/lab01/presentation/Makefile
create mode 100644 labs/lab01/presentation/image/kulyabov.jpg
create mode 100644 labs/lab01/presentation/presentation.md
create mode 100644 labs/lab01/report/Makefile
create mode 100644 labs/lab01/report/bib/cite.bib
create mode 100644 labs/lab01/report/image/placeimg_800_600_tech.jpg
create mode 100644 labs/lab01/report/pandoc/csl/gost-r-7-0-5-2008-numeric.csl
create mode 100755 labs/lab01/report/pandoc/filters/pandoc_eqnos.py
create mode 100755 labs/lab01/report/pandoc/filters/pandoc_fignos.py
create mode 100755 labs/lab01/report/pandoc/filters/pandoc_secnos.py
create mode 100755 labs/lab01/report/pandoc/filters/pandoc_tablenos.py
create mode 100644 labs/lab01/report/pandoc/filters/pandocxnos/__init__.py
create mode 100644 labs/lab01/report/pandoc/filters/pandocxnos/core.py
create mode 100644 labs/lab01/report/pandoc/filters/pandocxnos/main.py
create mode 100644 labs/lab01/report/pandoc/filters/pandocxnos/pandocattributes.py
create mode 100644 labs/lab01/report/report.md
create mode 100644 labs/lab02/presentation/Makefile
create mode 100644 labs/lab02/presentation/image/kulyabov.jpg
create mode 100644 labs/lab02/presentation/presentation.md
create mode 100644 labs/lab02/report/Makefile
create mode 100644 labs/lab02/report/bib/cite.bib
create mode 100644 labs/lab02/report/image/placeimg_800_600_tech.jpg
create mode 100644 labs/lab02/report/pandoc/csl/gost-r-7-0-5-2008-numeric.csl
create mode 100755 labs/lab02/report/pandoc/filters/pandoc_eqnos.py
create mode 100755 labs/lab02/report/pandoc/filters/pandoc_fignos.py
create mode 100755 labs/lab02/report/pandoc/filters/pandoc_secnos.py
create mode 100755 labs/lab02/report/pandoc/filters/pandoc_tablenos.py
create mode 100644 labs/lab02/report/pandoc/filters/pandocxnos/__init__.py
create mode 100644 labs/lab02/report/pandoc/filters/pandocxnos/core.py
create mode 100644 labs/lab02/report/pandoc/filters/pandocxnos/main.py
create mode 100644 labs/lab02/report/pandoc/filters/pandocxnos/pandocattributes.py
create mode 100644 labs/lab02/report/report.md
create mode 100644 labs/lab03/presentation/Makefile
create mode 100644 labs/lab03/presentation/image/kulyabov.jpg
create mode 100644 labs/lab03/presentation/presentation.md
create mode 100644 labs/lab03/report/Makefile

```

Рис. 17: Рисунок 18. git add, git commit

```

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
vanechaeva@vanechaeva:~/work/study/2022-2023/Операционные системы/os-intro$ git push
Перечисление объектов: 38, готово.
Подсчет объектов: 100% (38/38), готово.
Сжатие объектов: 100% (30/30), готово.
Запись объектов: 100% (37/37), 343.00 Киб | 201.00 Киб/с, готово.
Всего 37 (изменений 4), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
remote: Resolving deltas: 100% (4/4), completed with 1 local object.
To github.com:vanechaeva/study_2022-2023_os-intro.git
 887c7b3..37cae3 master -> master

```

Рис. 18: Рисунок 19. git push

## Выводы

По итогу выполнения лабораторной работы удалось познакомиться с идеологией и инструментами системы управления версиями git

# Контрольные вопросы

- 1) Что такое системы контроля версий (VCS) и для решения каких задач они предназначаются?

VCS — это практика отслеживания изменений программного кода и управления им. Системы контроля версий — это программные инструменты, помогающие командам разработчиков управлять изменениями в исходном коде с течением времени. В свете усложнения сред разработки они помогают командам разработчиков работать быстрее и эффективнее.

- 2) Объясните следующие понятия VCS и их отношения: хранилище, commit, история, рабочая копия.

Хранилище версий – или репозиторий - в нем хранятся все документы вместе с историей их изменения и другой служебной информацией. commit делает для проекта снимок текущего состояния изменений, добавленных в раздел проиндексированных файлов. Такие подтвержденные снимки состояния можно рассматривать как «безопасные» версии проекта — VCS не будет их менять, пока вы явным образом не попросите об этом.

log или история перечисляет коммиты, сделанные в репозитории в обратном к хронологическому порядку — последние коммиты находятся вверху. Тут же можно увидеть различие одного коммита от другого

Рабочая копия является снимком одной версии проекта. Эти файлы извлекаются из сжатой базы данных в каталоге Git и помещаются на диск, для того чтобы их можно было использовать или редактировать

Отношения: правки вносятся в рабочую копию, делаете коммит. Коммиты хра-

няться в репозиториях, `log` (история) позволяет посмотреть историю коммитов в репо.

- 3) Что представляют собой и чем отличаются централизованные и децентрализованные VCS? Приведите примеры VCS каждого вида.

Централизованные системы контроля версий представляют собой приложения типа клиент-сервер, когда репозиторий проекта существует в единственном экземпляре и хранится на сервере. Доступ к нему осуществлялся через специальное клиентское приложение. Примеры - CVS, Subversion.

Децентрализованные VCS позволяют хранить репозиторий у каждого разработчика, работающего с данной системой. При этом можно выделить центральный репозиторий (условно), в который будут отправляться изменения из локальных и, с ним же эти локальные репозитории будут синхронизироваться. При работе с такой системой, пользователи периодически синхронизируют свои локальные репозитории с центральным и работают непосредственно со своей локальной копией. Примеры – Git, Mercurial.

- 4) Опишите действия с VCS при единоличной работе с хранилищем.

Установить и настроить VCS клиента. Создать репозиторий. Это можно сделать с помощью команды `“git init”` (если используется Git)

Добавить файлы в репозиторий. Это можно сделать с помощью команды `“git add”`

Создать коммит. Коммит можно создать с помощью команды `“git commit”` (или аналогичной команды в другой VCS).

Просматривать историю коммитов. Это можно сделать с помощью команды `“git log”`

Восстановить предыдущую версию проекта. Это можно сделать с помощью команды `“git checkout”`.

Создавать и удалять ветки (branch).

- 5) Опишите порядок работы с общим хранилищем VCS.

Получение копии проекта из общего хранилища. Для этого нужно выполнить команду “git clone” (если используется Git).

Создание новой ветки. Если вы планируете внести изменения в проект, то для этого необходимо создать новую ветку (branch) в вашем локальном репозитории

Внесение изменений. Коммит изменений. После внесения изменений в файлы проекта, необходимо выполнить команду “git commit”

Отправка изменений на сервер. Для этого выполните команду “git push”

Обновление локальной копии проекта. Для этого выполните команду “git pull”

6) Каковы основные задачи, решаемые инструментальным средством git?

- Возврат к любой версии кода из прошлого.
- Просмотр истории изменений.
- Совместная работа без боязни потерять данные или затереть чужую работу.

7) Назовите и дайте краткую характеристику командам git.

Описано в вопросах 4, 5.

8) Приведите примеры использования при работе с локальным и удалённым репозиториями.

Локально:

Создание локального репозитория. “git init” в терминале. Добавление файлов в репозиторий. После создания репозитория вы можете добавить файлы проекта в него, используя команду “git add ”.

Создание коммита. После добавления файлов вы можете создать коммит, используя команду “git commit -m ‘Commit message’”. Просмотр истории коммитов. Вы можете просмотреть историю коммитов, используя команду “git log”.

Восстановление предыдущей версии. Если в проекте была допущена ошибка или нужно вернуться к предыдущей версии проекта, это можно сделать с помощью команды “git checkout ”.

Удаленно: Клонирование удаленного репозитория. Чтобы получить локальную



копию проекта, вы можете клонировать репозиторий с помощью команды “git clone ”.

Добавление изменений в локальный репозиторий. После того, как вы получили копию проекта, вы можете вносить изменения и добавлять их в локальный репозиторий с помощью команд “git add ” и “git commit -m ‘Commit message’”.

Отправка изменений в удаленный репозиторий. После добавления изменений в локальный репозиторий вы можете отправить их в удаленный репозиторий, используя команду “git push”.

Получение изменений из удаленного репозитория. Если в удаленном репозитории были внесены изменения, вы можете получить их и обновить свою локальную копию проекта, используя команду “git pull”.

Восстановление предыдущей версии. Если в проекте была допущена ошибка или нужно вернуться к предыдущей версии проекта, это можно сделать с помощью команды “git checkout ” в локальном репозитории. Если нужно откатить изменения в удаленном репозитории, можно использовать команду "git revert

#### 9) Что такое и зачем могут быть нужны ветви (branches)?

Ветви нужны для того, чтобы разделять код. Например одна ветка у нас может быть основная для разработки. Если мы делаем новый функционал, то мы создаем новую ветку под него, а после окончания работы сливаем то, что мы сделали в основную ветку. Это дает нам возможность легко откатывать код, если вдруг мы передумаем его сливать в основную ветку, либо делать несколько различных изменений в разных ветках.

#### 10) Как и зачем можно игнорировать некоторые файлы при commit?

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. В Git нет специальной команды для указания игнорируемых файлов: вместо этого необходимо вручную

отредактировать файл `.gitignore`, чтобы указать в нем новые файлы, которые должны быть проигнорированы. Файлы `.gitignore` содержат шаблоны, которые сопоставляются с именами файлов в репозитории для определения необходимости игнорировать эти файлы.