

# **Лабораторная работа №2**

**Первоначальная настройка git**

Юсупова Ксения Равиловна

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Выполнение лабораторной работы</b>	<b>6</b>
2.1	Установка программного обеспечения . . . . .	6
2.2	Базовая настройка git . . . . .	7
2.3	Создание ключей ssh . . . . .	7
2.4	Создание ключей pgr . . . . .	8
2.5	Настройка github . . . . .	9
2.6	Добавление PGP ключа в GitHub . . . . .	10
2.7	Настройка автоматических подписей коммитов git . . . . .	11
2.8	Настройка gh . . . . .	11
2.9	Создание репозитория курса на основе шаблона . . . . .	12
2.10	Настройка каталога курса . . . . .	13
<b>3</b>	<b>Выводы</b>	<b>16</b>
<b>4</b>	<b>Ответы на контрольные вопросы.</b>	<b>17</b>

## Список иллюстраций

2.1	Установили git . . . . .	6
2.2	Установили gh . . . . .	6
2.3	Задали имя и email владельца репозитория . . . . .	7
2.4	Настроили utf-8 в выводе сообщений git и задали имя начальной ветки . . . . .	7
2.5	параметр autocrlf и параметр safecrlf . . . . .	7
2.6	Создали ключи ssh . . . . .	8
2.7	Сгенерировали ключ pgr . . . . .	9
2.8	Создали учетную запись . . . . .	10
2.9	Вывели список ключей и скопировали отпечаток приватного ключа . . . . .	10
2.10	Скопировали наш сгенерированный PGP ключ в буфер обмена и далее создали его на GitHub . . . . .	11
2.11	Используя email, указали Git применять его при подписи коммитов . . . . .	11
2.12	авторизовались . . . . .	12
2.13	Создаём необходимый репозиторий . . . . .	13
2.14	Переходим в каталог курса . . . . .	13
2.15	Удаляем лишние файлы, создаём необходимые каталоги и отправляем файлы на сервер . . . . .	14
2.16	Отправили файлы на сервер . . . . .	15

## **Список таблиц**

# 1 Цель работы

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

## 2 Выполнение лабораторной работы

### 2.1 Установка программного обеспечения

Установка git(рис. 2.1).

```
[ksyusha@ksyusha ~]$ sudo -i
[sudo] пароль для ksyusha:
[root@ksyusha ~]# dnf install git
Обновление и загрузка репозитория:
Fedora 41 - x86_64 - Updates      100% | 78.2 KiB/s | 24.3 KiB | 00m00s
Fedora 41 - x86_64 - Updates      100% | 654.6 KiB/s | 2.8 MiB | 00m04s
Репозитории загружены.
```

Рис. 2.1: Установили git

Установка gh(рис. 2.2).

```
[root@ksyusha ~]# dnf install gh
Обновление и загрузка репозитория:
Репозитории загружены.
Пакет      Арх.  Версия      Репозиторий  Размер
Установка:
gh          x86_64  2.65.0-1.fc41  updates      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% | 7.5 MiB/s | 10.3 MiB | 00m01s
-----
[1/1] Total                          100% | 4.2 MiB/s | 10.3 MiB | 00m02s
Выполнение транзакции
[1/3] Проверить файлы п 100% | 7.0 B/s | 1.0 B | 00m00s
[2/3] Подготовить тран100% | 0.0 B/s | 1.0 B | 00m01s
[3/3] Установка gh-0:2.65.0-1 100% | 7.4 MiB/s | 42.7 MiB | 00m06s
Завершено!
```

Рис. 2.2: Установили gh

## 2.2 Базовая настройка git

Зададим имя и email владельца репозитория(рис. 2.3).

```
[root@ksyusha ~]# git config --global user.name "yaneksyusha"
[root@ksyusha ~]# git config --global user.email "1132247531@pfur.ru"
```

Рис. 2.3: Задали имя и email владельца репозитория

Настроим utf-8 в выводе сообщений git и зададим имя начальной ветки (будем называть её master)(рис. 2.4).

```
[root@ksyusha ~]# git config --global core.quotePath false
[root@ksyusha ~]# git config --global init.defaultBranch master
```

Рис. 2.4: Настроили utf-8 в выводе сообщений git и задали имя начальной ветки

Настроим параметр autocrlf и параметр safecrlf(рис. 2.5).

```
[root@ksyusha ~]# git config --global core.autocrlf input
[root@ksyusha ~]# git config --global core.safecrlf warn
```

Рис. 2.5: параметр autocrlf и параметр safecrlf

## 2.3 Создание ключей ssh

Создаём ключи ssh по алгоритму rsa с ключём размером 4096 бит и по алгоритму ed25519(рис. 2.6).

```

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:
SHA256:72nLa9Laznle5QgkoUnoJcQe3X1qiDAGfXcletP6jk root@ksyusha
The key's randomart image is:
+---[RSA 4096]-----+
|... ++o..= .|
|, o o =oo= =|
|.o .+.++ o o|
| . . . . + |
| . . S . . |
| . . . . o +|
| . . + o .|
| .+=Eoo|
| .+%Bo|
+-----[SHA256]-----+
[root@ksyusha ~]# 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:
SHA256:6aolADluZ/FFpt0YrcU1RY5Bx2dyQirETw32hkZ+Jys root@ksyusha
The key's randomart image is:
+--[ED25519 256]--+
| .+.OB=|
| . @.=.X= +|
| + . = Bo* **.|
| . o o =+.o +|
| o + o S E .|
| . o . .|
| . o .|
| . o|
| ...|
+-----[SHA256]-----+

```

Рис. 2.6: Создали ключи ssh

## 2.4 Создание ключей pgr

Генерируем ключ pgr с учетом необходимых опций(рис. 2.7).



```
[ksyusha@ksyusha ~]$ 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.

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: Юсупова Ксения Равиловна
Адрес электронной почты: 1132247531@pfur.ru
Примечание:
Используется таблица символов 'utf-8'.
Вы выбрали следующий идентификатор пользователя:
  "Юсупова Ксения Равиловна <1132247531@pfur.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
```

Рис. 2.7: Сгенерировали ключ gpg

## 2.5 Настройка github

Создаём учётную запись и заполнили основные данные на <https://github.com>. (рис. 2.8).

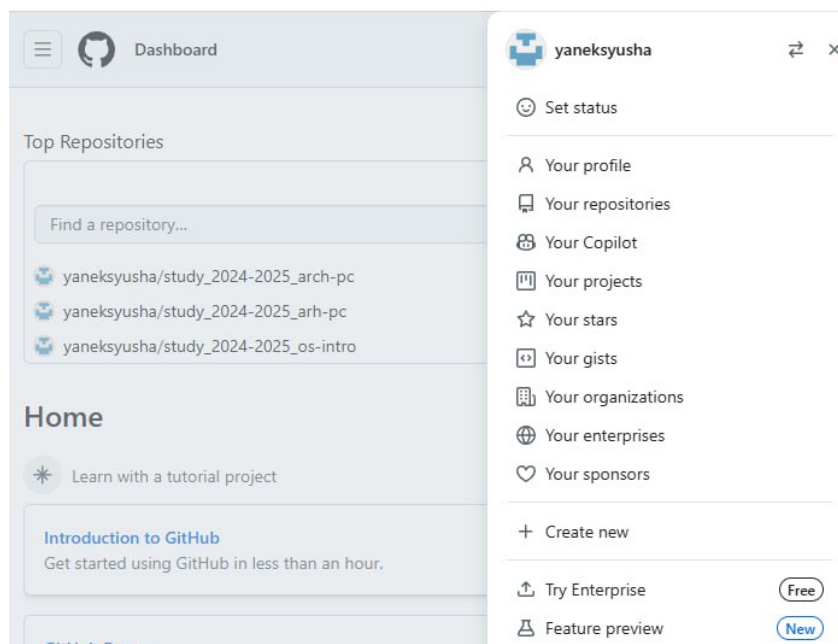


Рис. 2.8: Создали учетную запись

## 2.6 Добавление PGP ключа в GitHub

Выводим список ключей и копируем отпечаток приватного ключа(рис. 2.9).

```
[ksyusha@ksyusha ~]$ 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, 0
f, 1u
[keyboard]
-----
sec rsa4096/ADF28D71B04B1145 2025-03-05 [SC]
      89EAD5EC19D5014645F2B4E2ADF28D71B04B1145
uid          [ абсолютно ] Юсупова Ксения Равиловна <1132247531@pfur.ru>
ssb rsa4096/1F6CD197E310E913 2025-03-05 [E]
```

Рис. 2.9: Вывели список ключей и скопировали отпечаток приватного ключа

Скопировали наш сгенерированный PGP ключ в буфер обмена. Перешли в настройки GitHub, нажали на кнопку New GPG key и вставили полученный ключ в поле ввода.(рис. 2.10).

```
[ksyusha@ksyusha ~]$ gpg --armor --export 1132247531@pfur.ru | xclip -sel clip
```

Рис. 2.10: Скопировали наш сгенерированный PGP ключ в буфер обмена и далее создали его на GitHub

## 2.7 Настройка автоматических подписей коммитов git

Используя введённый email, указали Git применять его при подписи коммитов(рис. 2.11).

```
[ksyusha@ksyusha ~]$ git config --global user.signingkey 1132247531@pfur.ru  
[ksyusha@ksyusha ~]$ git config --global commit.gpgsign true  
[ksyusha@ksyusha ~]$ git config --global gpg.program $(which gpg2)
```

Рис. 2.11: Используя email, указали Git применять его при подписи коммитов

## 2.8 Настройка gh

Для начала авторизируемся(рис. 2.12).

```

[ksyusha@ksyusha ~]$ gh auth login
? Where do you use GitHub? GitHub.com
? What is your preferred protocol for Git operations on this host? SSH
? Generate a new SSH key to add to your GitHub account? Yes
? Enter a passphrase for your new SSH key (Optional):
? Title for your SSH key: Sway
? How would you like to authenticate GitHub CLI? Login with a web browser

! First copy your one-time code: E19F-2D5C
Press Enter to open https://github.com/login/device in your browser...
✓ Authentication complete.
- gh config set -h github.com git_protocol ssh
✓ Configured git protocol
✓ Uploaded the SSH key to your GitHub account: /home/ksyusha/.ssh/id_ed25519.pub
b
✓ Logged in as yaneksyusha
[ksyusha@ksyusha ~]$ mkdir -p ~/work/study/2024-2025/"Операционные системы"
[ksyusha@ksyusha ~]$ cd ~/work/study/2024-2025/"Операционные системы"
[ksyusha@ksyusha Операционные системы]$ gh repo create study_2024-2025_os-intro
--template=yamadharma/course-directory-student-template --public
✓ Created repository yaneksyusha/study_2024-2025_os-intro on GitHub
https://github.com/yaneksyusha/study_2024-2025_os-intro
[ksyusha@ksyusha Операционные системы]$ git clone --recursive git@github.com:yaneksyusha/study_2024-2025_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:+DiY3wvvV6TuJJhbpZisF/zLDA0zPMSvHdkr4UvC0qU.
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: yes
Warning: Permanently added 'github.com' (ED25519) to the list of known hosts.
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), 19.38 Киб | 9.69 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»

```

Рис. 2.12: авторизовались

## 2.9 Создание репозитория курса на основе шаблона

Создаём репозиторий курса на основе шаблона для 2024–2025 учебного года и предмета «Операционные системы» (рис. 2.13).

```
[ksyusha@ksyusha ~]$ mkdir -p ~/work/study/2024-2025/"Операционные системы"
[ksyusha@ksyusha ~]$ cd ~/work/study/2024-2025/"Операционные системы"
[ksyusha@ksyusha Операционные системы]$ gh repo create study_2024-2025_os-intro
--template=yamadharma/course-directory-student-template --public
✓ Created repository yaneksyusha/study_2024-2025_os-intro on GitHub
https://github.com/yaneksyusha/study_2024-2025_os-intro
[ksyusha@ksyusha Операционные системы]$ git clone --recursive git@github.com:ya
neksyusha/study_2024-2025_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:+DiY3wvV6TuJJhbpZisF/zLDA0zPMSvHdkr4UvC0qU.
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: yes
Warning: Permanently added 'github.com' (ED25519) to the list of known hosts.
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), 19.38 КиБ | 9.69 МиБ/с, готово.
Определение изменений: 100% (1/1), готово.
Подмодуль «template/presentation» (https://github.com/yamadharma/academic-prese
ntation-markdown-template.git) зарегистрирован по пути «template/presentation»
```

Рис. 2.13: Создаём необходимый репозиторий

## 2.10 Настройка каталога курса

Переходим в каталог курса (рис.2.14).

```
[ksyusha@ksyusha Операционные системы]$ cd ~/work/study/2024-2025/"Операционные
системы"/os-intro
```

Рис. 2.14: Переходим в каталог курса

Удаляем лишние файлы, создаём необходимые каталоги и отправляем файлы на сервер(рис. 2.15).

```

[ksyusha@ksyusha os-intro]$ git add .
[ksyusha@ksyusha os-intro]$ git commit -am 'feat(main): make course structure'
error: gpg failed to sign the data:
[GNUPG:] KEY_CONSIDERED 89EAD5EC19D5014645F2B4E2ADF28D71B04B1145 2
[GNUPG:] BEGIN_SIGNING H8
[GNUPG:] PINENTRY_LAUNCHED 2065 gnome3 1.3.1-unknown /dev/pts/0 foot :0 20620/1000/5
1000/1000 0
gpg: подписать не удалось: Фраза-пароль не задана
[GNUPG:] FAILURE sign 67109041
gpg: signing failed: Фраза-пароль не задана

fatal: сбой записи объекта коммита
[ksyusha@ksyusha os-intro]$ git commit -am 'feat(main): make course structure'
[master d17f440] feat(main): make course structure
405 files changed, 98413 insertions(+), 14 deletions(-)
create mode 100644 labs/README.md
create mode 100644 labs/README.ru.md
create mode 100644 labs/lab01/presentation/.projectile
create mode 100644 labs/lab01/presentation/.texlabroot
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/.projectile
create mode 100644 labs/lab02/presentation/.texlabroot
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

```

Рис. 2.15: Удаляем лишние файлы, создаём необходимые каталоги и отправляем файлы на сервер

Отправляем файлы на сервер(рис. 2.16).



```
[ksyusha@ksyusha os-intro]$ git push
Перечисление объектов: 40, готово.
Подсчет объектов: 100% (40/40), готово.
При сжатии изменений используется до 4 потоков
Сжатие объектов: 100% (30/30), готово.
Запись объектов: 100% (38/38), 342.31 КиБ | 13.17 МБ/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:yaneksyusha/study_2024-2025_os-intro.git
   e9c8a73..d17f440  master -> master
[ksyusha@ksyusha os-intro]$
```

Рис. 2.16: Отправили файлы на сервер

## **3 Выводы**

В ходе лабораторной работы мы изучили идеологию и применение средств контроля версий, а также освоили умения по работе с git.



## 4 Ответы на контрольные вопросы.

1. Системы контроля версий (VCS)- это инструменты, которые позволяют отслеживать изменения в файлах и координировать работу нескольких людей над этим проектом. Они предназначены для отслеживания изменений в коде и файлах, удобно для использования в совместной работе над проектами для нескольких разработчиков.
2. Хранилище - это репозиторий, в котором хранятся все версии проекта и его файлы Commit - это операция, которая сохраняет текущее состояние файлов в хранилища с комментарием, описывающим эти изменения. История- это последовательность всех коммитов репозитория. Рабочая копия- копия, сделанная из версий репозитория, с которой непосредственно работает сам разработчик.
3. Централизованные VCS представляют собой все изменения, которые хранятся на одном центральном сервере, с ним работают все разработчики. В пример можно привести CVS.

Децентрализованные VCS представляют собой системы, в которых используется множество репозиториях одного проекта у каждого из разработчиков. Пример: Git, Mercurial.

4. Порядок работы с хранилищем при единоличной работе: Создание репозитория, разработка проекта и при внесении изменений в файлы отправляются на сервер

## 5. Порядок работы с общим хранилищем VCS.

Клонирование репозитория, создание новой ветки, внесение изменений: Изменение файлов в рабочей копии. Добавление и коммит изменений, синхронизация и отправка изменений.

## 6. Основные задачи Git.

Отслеживание изменений: Git отслеживает все изменения в файлах проекта.

Ведение истории: Git сохраняет историю всех коммитов, что позволяет видеть, кто и когда вносил изменения.

Управление ветвями и слияниями: Git позволяет легко создавать ветки для новых функций и сливать их обратно в основную ветку после завершения работы.

Поддержка совместной работы: Git упрощает работу нескольких разработчиков над одним проектом, позволяя им синхронизировать свои изменения.

## 7. Команды git.

`git init`: Инициализация нового репозитория.

`git clone` : Клонирование удалённого репозитория.

`git add` : Добавление изменённых файлов в индекс.

`git commit -m "message"`: Фиксация изменений с сообщением.

`git status`: Проверка статуса рабочей копии.

`git log`: Просмотр истории коммитов.

`git branch`: Управление ветками.

`git checkout` : Переключение между ветками.

`git merge` : Слияние указанной ветки с текущей.

`git pull`: Получение обновлений из удалённого репозитория.

`git push origin` : Отправка изменений в удалённый репозиторий.

## 8. Примеры работы с локальным и удалённым репозиториями.

Локальный репозиторий: Инициализация нового репозитория `git init`

Создание файла и добавление текста `echo "Hello World" > hello.txt`

Добавление файла в индекс `git add hello.txt`

Фиксация изменений `git commit -m "Initial commit"`

Удалённый репозиторий: Клонирование удалённого репозитория `git clone https://github.com/user/repo.git`

Переход в директорию клонированного репозитория `cd repo`

Создание нового файла `echo "New feature" > feature.txt`

Добавление нового файла в индекс `git add feature.txt`

Фиксация изменений `git commit -m "Add new feature"`

Отправка изменений в удалённый репозиторий `git push origin main`

## 9. Ветви (branches).

Ветви в Git позволяют разработчикам работать над новыми функциями или исправлениями ошибок, не затрагивая основную (обычно это ветка `main` или `master`). Это позволяет изолировать изменения до тех пор, пока они не будут готовы к интеграции.

## 10. Игнорирование файлов при commit.

Файл `.gitignore` используется для указания файлов и директорий, которые не должны отслеживаться системой контроля версий. Это полезно для исключения временных файлов, конфиденциальной информации или других данных, которые не должны попадать в репозиторий.