

Лабораторная работа №2

Первоначальная настройка git

Варвара Алексеевна Буценко

Содержание

Цель работы	5
Задание	6
Теоретическое введение	7
Выполнение лабораторной работы	8
Контрольные вопросы	15
Выводы	21
Список литературы	22

Список иллюстраций

1	версия git	8
2	версия gh	8
3	базовые настройки	9
4	ключи ssh	10
5	ключ pgr	10
6	настройки github	11
7	PGP ключ в GitHub	11
8	PGP ключ в GitHub	11
9	подписи коммитов	12
10	аутентификации в GitHub CLI	13
11	каталог курса	13
12	каталог курса	14
13	файлы на сервере	14

Список таблиц

Цель работы

Целью данной работы является изучение идеологии и применение средств контроля версий. Освоить умения по работе с git.

Задание

- 1) Создать базовую конфигурацию для работы с git.
- 2) Создать ключ SSH.
- 3) Создать ключ PGP.
- 4) Настроить подписи git.
- 5) Зарегистрироваться на Github.
- 6) Создать локальный каталог для выполнения заданий по предмету.

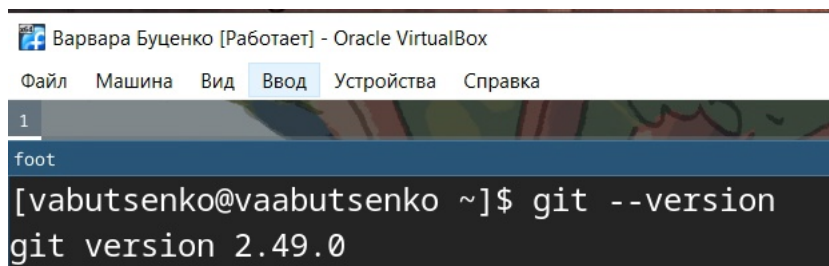
Теоретическое введение

Здесь описываются теоретические аспекты, связанные с выполнением работы.

Вся необходимая теория по лабораторной работе №2 находится в разделе курса “Операционные системы” по ссылке <https://esystem.rudn.ru/mod/page/view.php?id=1103908>

Выполнение лабораторной работы

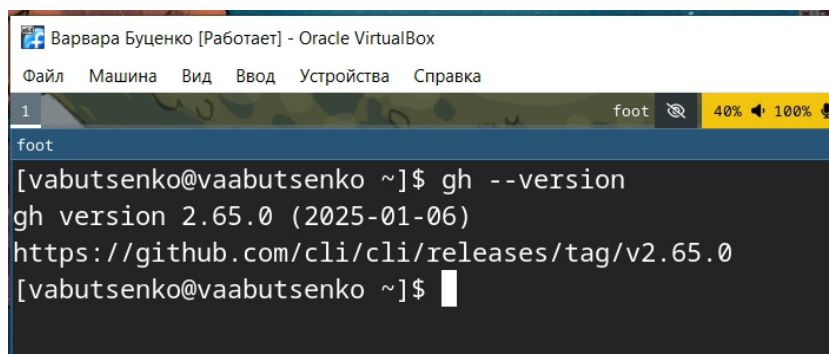
1) Устанавливаю git. Заранее сделала это, поэтому использую команду `git --version`, чтобы показать свою версию.



```
Варвара Буценко [Работает] - Oracle VirtualBox
Файл  Машина  Вид  Ввод  Устройства  Справка
1
foot
[vabutsenko@vaabutsenko ~]$ git --version
git version 2.49.0
```

Рис. 1: версия git

2) Устанавливаю gh. Заранее сделала это, поэтому использую команду `gh --version`, чтобы показать свою версию.



```
Варвара Буценко [Работает] - Oracle VirtualBox
Файл  Машина  Вид  Ввод  Устройства  Справка
1
foot
[vabutsenko@vaabutsenko ~]$ gh --version
gh version 2.65.0 (2025-01-06)
https://github.com/cli/cli/releases/tag/v2.65.0
[vabutsenko@vaabutsenko ~]$
```

Рис. 2: версия gh

3) Настраиваю git: задаю имя и email владельца: `- git config --global user.name "Name Surname"`, `- git config --global user.email "work@mail"`.

- Настраиваю utf-8 в выводе сообщений git:
- `git config --global core.quotePath false`
- Настраиваю верификацию и подписание коммитов git.
- Задаю имя начальной ветки (буду называть её master):
- `git config --global init.defaultBranch master`
- Параметр autocrlf:
- `git config --global core.autocrlf input`
- Параметр safecrlf:
- `git config --global core.safecrlf warn`
- С помощью команды `git config --list --show-origin` проверяю базовые настройки git.

```
[vabutsenko@vaabutsenko ~]$ git config --list --show-origin
file:/home/vabutsenko/.gitconfig user.name=Varvara Butsenko
file:/home/vabutsenko/.gitconfig user.email=1032200547@pfur.ru
file:/home/vabutsenko/.gitconfig core.quotePath=false
file:/home/vabutsenko/.gitconfig lfs.cachePath=/tmp/git-lfs-cache
file:/home/vabutsenko/.gitconfig credential.helper=https://github.com.helper=
file:/home/vabutsenko/.gitconfig credential.helper=https://github.com.helper=/usr/bin/gh auth git-credential
file:/home/vabutsenko/.gitconfig credential.helper=https://gist.github.com.helper=
file:/home/vabutsenko/.gitconfig credential.helper=https://gist.github.com.helper=/usr/bin/gh auth git-credential
file:/usr/share/git-core/config core.repositoryformatversion=0
file:/usr/share/git-core/config core.filemode=true
file:/usr/share/git-core/config core.bare=false
file:/usr/share/git-core/config core.logallrefupdates=true
file:/usr/share/git-core/config lfs.repositoryformatversion=0
file:/usr/share/git-core/config lfs.https://github.com/vabucenko/opera2025.git/info/lfs.locksverify=false
file:/usr/share/git-core/config remote.origin.url=https://github.com/vabucenko/study_2024-2025_os-intro.git
file:/usr/share/git-core/config remote.origin.fetch=+refs/heads/*:refs/remotes/origin/*
file:/usr/share/git-core/config branch.main.remote=origin
file:/usr/share/git-core/config branch.main.merge=refs/heads/main
[vabutsenko@vaabutsenko ~]$
```

Рис. 3: базовые настройки

4) Создаю ключ ssh по алгоритму rsa с ключём размером 4096 бит:

- `ssh-keygen -t rsa -b 4096`
- Создаю ключ ssh по алгоритму ed25519:
- `ssh-keygen -t ed25519`

- Проверяю созданные SSH-ключи:
- `ls -al ~/.ssh`
- `cat ~/.ssh/id_ed25519.pub`

```
vabutsenko@vaabutsenko ~]$ ls -al ~/.ssh
total 16
-rwx----- 1 vabutsenko vabutsenko 100 мая 6 19:39 .
-rwx----- 1 vabutsenko vabutsenko 1272 мая 14 13:36 ..
-rw----- 1 vabutsenko vabutsenko 444 мая 6 15:44 id_ed25519
-rw-r--r-- 1 vabutsenko vabutsenko 121 мая 6 15:44 id_ed25519.pub
-rw----- 1 vabutsenko vabutsenko 828 мая 6 19:39 known_hosts
-rw-r--r-- 1 vabutsenko vabutsenko 92 мая 6 19:39 known_hosts.old
vabutsenko@vaabutsenko ~]$ cat ~/.ssh/id_ed25519.pub
ssh-ed25519 AAAAC3NzaC1lZD11NTE5AAAAIJZfUP6BN2G96qqtr0Jmmf9H0ytr8UwE2ZlmiRpGm9Rd2 Vabutsenko Varvara <1032200547@pfur.ru>
```

Рис. 4: ключи ssh

5) Генерирую ключ pgp

- `gpg --full-generate-key` Из предложенных опций выбираю:
- тип RSA and RSA;
- размер 4096;
- выбираю срок действия; значение по умолчанию — 0 (срок действия не истекает никогда).
- GPG запросит личную информацию, которая сохранится в ключе:
- Имя: vabutsenko
- Адрес электронной почты: 1032200547@pfur.ru

```
[vabutsenko@vaabutsenko ~]$ gpg --list-secret-keys --keyid-format LONG
[keyboxd]
-----
sec   rsa4096/3D14B339EB1DA26B 2025-05-06 [SC]
      D69F196E9B9F0AC8BFF459233D14B339EB1DA26B
uid           [ абсолютно ] varvara <1032200547@pfur.ru>
ssb   rsa4096/478D8F17DC2A3345 2025-05-06 [E]

[vabutsenko@vaabutsenko ~]$
```

Рис. 5: ключ pgp

- 6) Настраиваю github. В прошлом году при прохождении курса “архитектура компьютера” я уже создавала учётную запись, так что использую для выполнения задания именно её.

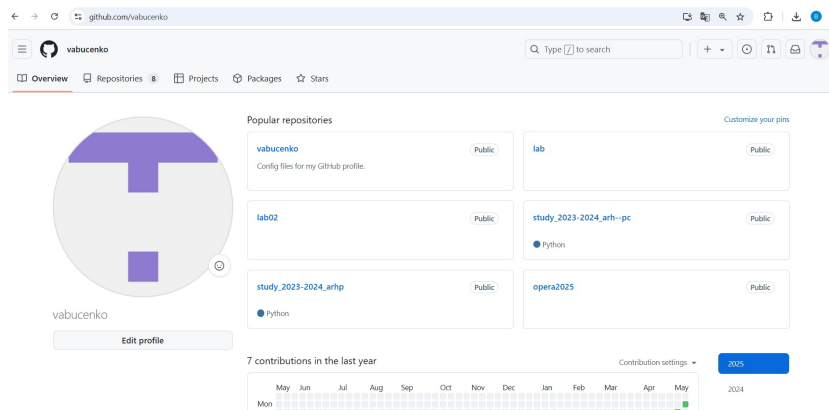


Рис. 6: настройки github

7) Добавляю PGP ключ в GitHub

- Вывожу список ключей и копирую отпечаток приватного ключа:
- `gpg --list-secret-keys --keyid-format LONG`
- Перехожу в настройки GitHub (<https://github.com/settings/keys>), нажимаю на кнопку New GPG key и вставляю полученный ключ в поле ввода.

```
[vabutsenko@vaabutsenko ~]$ gpg --list-secret-keys --keyid-format LONG
[keyboxd]
-----
sec   rsa4096/3D14B339EB1DA26B 2025-05-06 [SC]
      D69F196E9B9F0AC8BFF459233D14B339EB1DA26B
uid           [ абсолютно ] varvara <1032200547@pfur.ru>
ssb   rsa4096/478D8F17DC2A3345 2025-05-06 [E]

[vabutsenko@vaabutsenko ~]$
```

Рис. 7: PGP ключ в GitHub

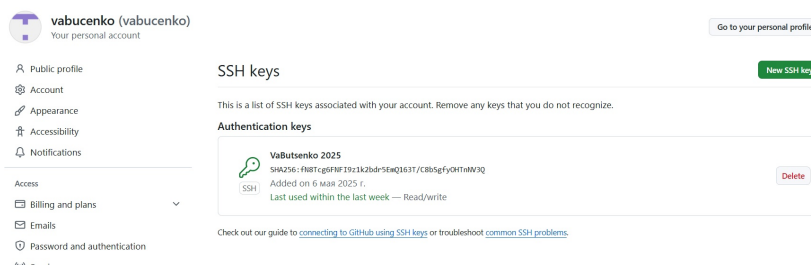
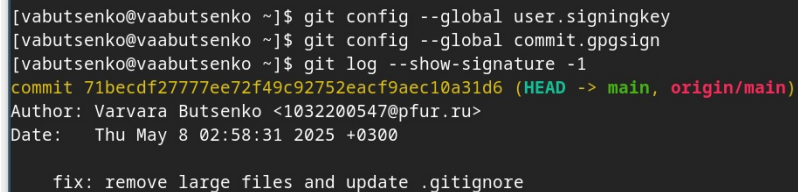


Рис. 8: PGP ключ в GitHub

8) Проверяю подписи коммитов

- `git config --global user.signingkey`
- `git config --global commit.gpgsign`
- `git log --show-signature -1`



```
[vabutsenko@vaabutsenko ~]$ git config --global user.signingkey
[vabutsenko@vaabutsenko ~]$ git config --global commit.gpgsign
[vabutsenko@vaabutsenko ~]$ git log --show-signature -1
commit 71becdf27777ee72f49c92752eacf9aec10a31d6 (HEAD -> main, origin/main)
Author: Varvara Butsenko <1032200547@pfur.ru>
Date: Thu May 8 02:58:31 2025 +0300

fix: remove large files and update .gitignore
```

Рис. 9: подписи коммитов

9) Проверяю аутентификации в GitHub CLI

- `gh auth status`
- Создаю шаблон рабочего пространства.
- `mkdir -p ~/work/study/2024-2025/"Operacionnie systems"`
- `cd ~/work/study/2024-2025/"Operacionnie systems"`
- `gh repo create study_2024-2025_os-intro`
- `--template=yamadharma/course-directory-student-template --public git clone --recursive git@github.com:/study_2024-2025_os-intro.git os-intro`
- Перехожу в каталог курса:
- `cd ~/work/study/2024-2025/"Operacionnie systems"/os-intro`
- Удаляю лишние файлы:
- `rm package.json`
- Создаю необходимые каталоги:

- `echo os-intro > COURSE`
- `make`
- Отправляю файлы на сервер:
- `git add .`
- `git commit -am 'feat(main): make course structure'`
- `git push`

```

[vaabutsenko@vaabutsenko ~]$ gh auth status
github.com
✓ Logged in to github.com account vabucenko (keyring)
- Active account: true
- Git operations protocol: https
- Token: gho_*****
- Token scopes: 'gist', 'read:org', 'repo', 'workflow'
[vaabutsenko@vaabutsenko ~]$

```

Рис. 10: аутентификации в GitHub CLI

```

[vaabutsenko@vaabutsenko work]$ tree -L 3
.
├── rus_fixed_full_fixed.map
├── study
│   └── 2024-2025
│       └── Operacionnie systems
└── ...

4 directories, 1 file
[vaabutsenko@vaabutsenko work]$

```

Рис. 11: каталог курса

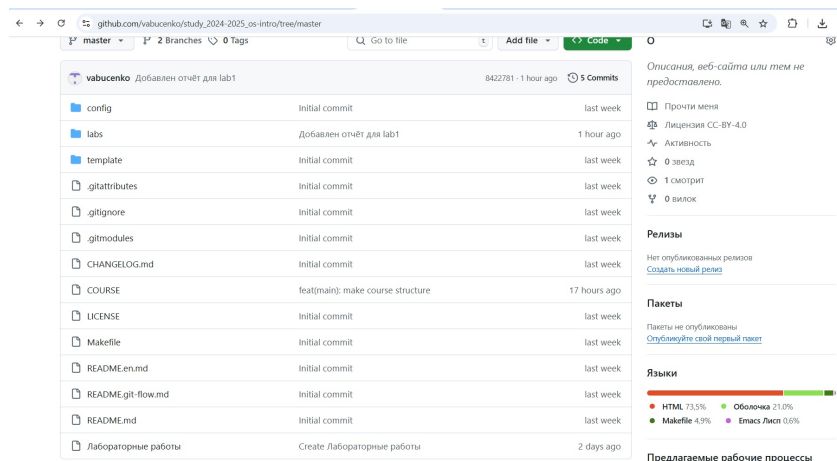


Рис. 12: каталог курса

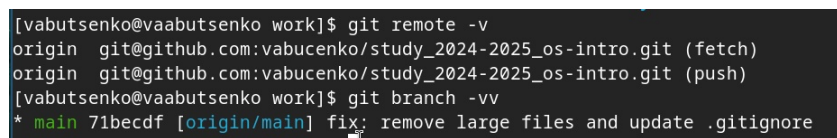


Рис. 13: файлы на сервере

Контрольные вопросы

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

Системы контроля версий (VCS) — это программные инструменты для отслеживания изменений в файлах (чаще всего в исходном коде) и координации работы нескольких участников. Они позволяют фиксировать историю изменений, возвращаться к предыдущим версиям и совместно работать над проектами.

Для решения каких задач предназначены: - Хранение истории изменений - Командная разработка - Резервное копирование - Анализ изменений

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

- Хранилище - база данных, хранящая всю историю проекта (файлы, изменения, авторов).
- Коммит - снимок состояния файлов на определённый момент времени.
- История - цепочка коммитов, отражающая эволюцию проекта.
- Рабочая копия - текущие файлы в вашей папке проекта, с которыми вы работаете.

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

- Ключевое различие — в способе хранения истории изменений и организации работы.

- Централизованные это системы, где вся история проекта хранится на едином сервере. Разработчики получают из него последние версии файлов и отправляют изменения обратно.
- Децентрализованные - это системы, где каждый участник проекта имеет полную копию репозитория со всей историей изменений. Работа ведётся локально, а синхронизация между копиями происходит через команды push/pull.

Централизованные: - SVN - CVS - Perforce

Децентрализованные: - Git - Mercurial - Bazaar

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

- Инициализация репозитория
- Ежедневные действия (проверить изменения, добавить файлы в индекс, зафиксировать изменения)
- Просмотр истории
- Работа с ветками
- Синхронизация с удалённым репозиторием
- Отмена изменений (при ошибках)

5) Опишите порядок работы с общим хранилищем VCS.

- Клонирование репозитория
- Создание своей ветки
- Ежедневная работа (внесение изменений в файлы, регулярное сохранение изменений (коммиты))
- Синхронизация с общим репозиторием
- Создание Pull/Merge Request
- Рецензирование и исправления
- Слияние изменений

- Удаление отработанной ветки

6) Каковы основные задачи, решаемые инструментальным средством git?

- Контроль версий
- Командная разработка
- Ветвление и слияние
- Резервное копирование
- Отслеживание изменений
- Тестирование идей
- Автоматизация процессов
- Распределённая разработка

7) Назовите и дайте краткую характеристику командам git.

- Настройка:
- `git config` Настройка параметров Git (имя пользователя, email, редактор и др.).
- `git config --global user.name "Ваше Имя"` Создание и клонирование репозитория
- `git init` Создает новый локальный репозиторий в текущей папке.
- `git clone` Клонирование удаленного репозитория на локальную машину.
- Работа с изменениями:
- `git status` Показывает состояние файлов (измененные, добавленные, неотслеживаемые).
- `git add` Добавляет файлы в индекс (staging area) для последующего коммита.

- `git commit -m "сообщение"` Фиксирует изменения в репозитории с комментарием.
- `git restore` Отменяет изменения в файле (до последнего коммита).
- Просмотр истории:
- `git log` Выводит историю коммитов (автор, дата, сообщение).
- `git diff` Показывает разницу между текущими изменениями и последним коммитом. Ветвление и слияние
- `git branch` Показывает список веток (текущая помечена *).
- `git checkout` Переключается на указанную ветку.
- `git merge` Вливает изменения из указанной ветки в текущую.
- Работа с удаленными репозиториями
- `git remote add` Добавляет удаленный репозиторий (например, origin).
- `git push` Отправляет локальные изменения на удаленный сервер.
- `git pull` Забирает изменения с удаленного репозитория и сливает с локальным.
- Отмена изменений:
- `git reset --hard HEAD` Сбрасывает все изменения до последнего коммита (осторожно!).
- `git revert` Создает новый коммит, отменяющий указанный.

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

- Работа с локальными репозиториями

- Создание локального репозитория: `git init my_project`
- Клонирование удалённого репозитория в локальный: `git clone https://github.com/username/repo`
- Добавление файлов в локальный репозиторий: `git add filename.txt`
- Коммит изменений в локальном репозитории: `git commit -m "Добавлен новый файл"`
- Просмотр статуса локального репозитория: `git status`
- Работа с удалёнными репозиториями
- Добавление удалённого репозитория в локальный (если он уже существует): `git remote add origin https://github.com/username/repo.git`
- Получение (пулл) изменений из удалённого репозитория: `git pull origin main`
- Отправка (пуш) локальных изменений в удалённый репозиторий: `git push origin main`
- Просмотр списка подключённых удалённых репозиториях: `git remote -v`
- Удаление удалённого репозитория: `git remote remove origin`

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

- Ветвь — это независимая линия разработки, позволяющая изолировать изменения от основного кода (обычно `main/master`). Каждая ветка содержит свою историю коммитов.
- Зачем нужны ветки?
- Параллельная разработка
- Изоляция экспериментов

- Гибкое управление версиями
- Контроль качества
- Упрощение Code Review

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

- Чтобы игнорировать файлы в Git, используется файл `.gitignore`, который находится в корневой директории репозитория. Этот файл содержит шаблоны, которые указывают, какие файлы или директории должны быть проигнорированы.
- Зачем игнорировать файлы?
- Не коммитить временные/системные файлы Логи, кэш, бинарники (например, `.log`, `.tmp`, `.exe`).
- Избегать личных настроек Файлы IDE (`.idea/`, `.vscode/`), конфиги с паролями.
- Снижать «шум» в репозитории Автогенерируемые файлы (`node_modules/`, **`pusache/`**).

Выводы

- В ходе выполнения лабораторной работы были успешно выполнены все поставленные задачи, связанные с изучением идеологии и применением средств контроля версий, а также освоением навыков работы с системой Git. В процессе работы были достигнуты следующие результаты:
- 1) Настройка базовой конфигурации Git: Были выполнены основные настройки Git, включая указание имени пользователя, email, настройку кодировки UTF-8, а также параметров для работы с ветками и окончаниями строк. Это обеспечило корректную работу системы контроля версий.
 - 2) Создание и настройка SSH и PGP ключей: Были сгенерированы ключи SSH (алгоритмы RSA и ed25519) и PGP, что позволило обеспечить безопасное взаимодействие с удалёнными репозиториями и подписывание коммитов. Ключи были успешно добавлены в аккаунт GitHub.
 - 3) Работа с GitHub: Была выполнена аутентификация в GitHub CLI, создан шаблон рабочего пространства, а также организована структура каталога для выполнения заданий. Локальные изменения были зафиксированы и отправлены на удалённый репозиторий.
 - 4) Освоение команд Git: В процессе работы были изучены и применены основные команды Git, такие как `git init`, `git clone`, `git add`, `git commit`, `git push`, `git pull`, а также команды для работы с ветками и историей изменений. Это позволило эффективно управлять версиями проекта.

Список литературы

1. Dash, P. Getting Started with Oracle VM VirtualBox / P. Dash. – Packt Publishing Ltd, 2013. – 86 сс.
2. Colvin, H. VirtualBox: An Ultimate Guide Book on Virtualization with VirtualBox. VirtualBox / H. Colvin. – CreateSpace Independent Publishing Platform, 2015. – 70 сс.
3. Vugt, S. van. Red Hat RHCSA/RHCE 7 cert guide : Red Hat Enterprise Linux 7 (EX200 and EX300) : Certification Guide. Red Hat RHCSA/RHCE 7 cert guide / S. van Vugt. – Pearson IT Certification, 2016. – 1008 сс.
4. Робачевский, А. Операционная система UNIX / А. Робачевский, С. Немнюгин, О. Стесик. – 2-е изд. – Санкт-Петербург : БХВ-Петербург, 2010. – 656 сс.
5. Немет, Э. Unix и Linux: руководство системного администратора. Unix и Linux / Э. Немет, Г. Снайдер, Т.Р. Хейн, Б. Уэйли. – 4-е изд. – Вильямс, 2014. – 1312 сс.
6. Колисниченко, Д.Н. Самоучитель системного администратора Linux : Системный администратор / Д.Н. Колисниченко. – Санкт-Петербург : БХВ-Петербург, 2011. – 544 сс.
7. Robbins, A. Bash Pocket Reference / A. Robbins. – O'Reilly Media, 2016. – 156 сс.