

# Лабораторная работа 3

---

Исаев Р. К.

Российский университет дружбы народов, Москва, Россия

## Информация

---

- Исаев Рамазан Курбанович
- студент НКАБД 01-24
- Российский университет дружбы народов
- [https://github.com/aksa077/study\\_2024-2025\\_os-intro.git](https://github.com/aksa077/study_2024-2025_os-intro.git)

Научиться оформлять отчёты с помощью легковесного языка разметки Markdown.

## Задание

---

Сделать отчёт по предыдущей лабораторной работе в формате Markdown. В качестве отчёта просьба предоставить отчёты в 3 форматах: pdf, docx и md(в архиве, поскольку он должен содержать скриншоты, Makefile и т.д.)

## Теоретическое введение

---

Чтобы создать заголовок, используйте знак ( # ). Чтобы задать для текста полужирное начертание, заключите его в двойные звездочки. Чтобы задать для текста курсивное начертание, заключите его в одинарные звездочки. Чтобы задать для текста полужирное и курсивное начертание, заключите его в тройные звездочки. Блоки цитирования создаются с помощью символа >. Неупорядоченный (маркированный) список можно отформатировать с помощью звездочек или тире. Чтобы вложить один список в другой, добавьте отступ для элементов дочернего списка. Упорядоченный список можно отформатировать с помощью соответствующих цифр. Чтобы вложить один список в другой, добавьте отступ для элементов дочернего списка. Синтаксис Markdown для встроенной ссылки состоит из части [link text] , представляющей текст гиперссылки, и части (file-name.md) – URL-адреса или имени файла, на который дается ссылка. Markdown поддерживает как встраивание фрагментов кода в предложение, так и их размещение между предложениями в виде отдельных огражденных блоков. Огражденные блоки кода — это простой способ выделить синтаксис для фрагментов кода. Внутритекстовые формулы делаются аналогично формулам LaTeX.



Для обработки файлов в формате Markdown будем использовать Pandoc. Конкретно, нам понадобится программа pandoc , pandoc-citeproc <https://github.com/jgm/pandoc/releases>, pandoc-crossref <https://github.com/lierdakil/pandoc-crossref/releases>. Преобразовать файл README.md можно следующим образом: 1 pandoc README.md -o README.pdf или так 1 pandoc README.md -o README.docx

## Выполнение лабораторной работы

---

1. Открываем подготовленный файл .md и заменяем на свои данные ( имя автора, тема лабораторной работы)

```
---  
## Front matter  
title: " отчёт по лабораторной работе"  
author: "Исаев Рамазан Курбанович"
```

Рис. 1: титульный лист

## 2. Записываем цель работы и теоретическое введение

### # Цель работы

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

### # Задание

1. Создать базовую конфигурацию для работы с `git`
2. Зарегистрироваться на `github`.
3. Создать клон `git`.
4. Создать клон `git`.
5. Настроить подписи `git`.
6. Создать локальный каталог для выполнения заданий по предмету.

### # Теоретическое введение

Системы контроля версий (**V**ersion **C**ontrol **S**ystem, **VCS**) применяются при работе нескольких человек над одним проектом. Обычно основное дерево проекта хранится в локальном или удалённом **репозитории**, к которому настроен доступ для участников проекта. При внесении изменений в содержание проекта система контроля версий позволяет их фиксировать, совмещать изменения, произведённые разными участниками проекта, производить откат к любой более ранней версии проекта, если это требуется.

В классических системах контроля версий используется централизованная модель, предполагающая наличие одного **депозитория** для хранения файлов. Выполнение большинства функций по управлению версиями осуществляется специальным сервером. Участник проекта (пользователь) перед началом работы посредством определённых команд получает нужную ему версию файла. После внесения изменений, пользователь размещает новую версию в хранилище. При этом предыдущие версии не удаляются из центрального хранилища и к ним можно вернуться в любой момент. Сервер может сохранять не полную версию изменённых файлов, а производить так называемую дельта-компрессию – сохранять только изменения между последовательными версиями, что позволяет уменьшить объём хранимых данных.

Системы контроля версий поддерживают возможность отслеживания и разрешения конфликтов, которые могут возникнуть при работе нескольких человек над одним файлом. Можно объединить (слить) изменения, сделанные разными участниками (автоматически или вручную), вручную выбрать нужную версию, отменить изменения вовсе или заблокировать файлы для изменения. В зависимости от настроек блокировка не позволяет другим пользователям получить рабочую копию или препятствует изменению рабочей копии файла средствами файловой системы ОС, обеспечивая таким образом, привилегированный доступ только одному пользователю, работающему с файлом.

Системы контроля версий также могут обеспечивать дополнительные, более гибкие функциональные возможности. Например, они могут поддерживать работу с несколькими версиями одного файла, сохраняя общую историю изменений до точки ветвления версий и собственные истории изменений каждой ветви. Кроме того, обычно доступна информация о том, кто из участников, когда и какие изменения вносил. Обычно такого рода информация хранится в журнале изменений, доступ к которому можно ограничить.

Рис. 2: цель работы

### 3. Начинаем выполнять основную часть лабораторной работы

```
# Выполнение лабораторной работы

## Установил git перейдя на роль суперпользователя

![Установка git](image/1.png){#fig:001 width=70%}

Установил gh

![Установка gh](image/2.png){#fig:002 width=70%}

## Базовая настройка Git.

Задал имя и email владельца репозитория.

![Владелец репозитория](image/3.png){#fig:003 width=70%}

## Создание ключа SSH.

Создаю ключ по алгоритму rsa с размером 4096 бит и по алгоритму ed25519.
|
![Создание ключа SSH](image/4.png){#fig:004 width=70%}

## Создание ключа PGP.

Сгенерировал ключ.

![Создание ключа PGP](image/5.png){#fig:005 width=70%}

![Создание ключа PGP](image/6.png){#fig:006 width=70%}
```

## 4. Подводим итоги и отвечаем на контрольные вопросы

### # Контрольные вопросы

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

Системы контроля версий (СКС) – это инструмент, который помогает разработчикам отслеживать изменения в коде, управлять различными версиями файлов и координировать работу в команде.

Задачи, для решения которых предназначаются системы контроля версий:

Отслеживание изменений. Можно узнать, когда, кем и зачем был создан, удалён или отредактирован какой-либо файл в **репозитории**. Это облегчает процесс поиска ошибок и отладки кода, помогает избегать путаницы и конфликтов. 243

Защита исходного кода. Система контроля версий предотвращает случайное или намеренное удаление, а также изменение важных файлов или функций, что может привести к нарушению работы программы.

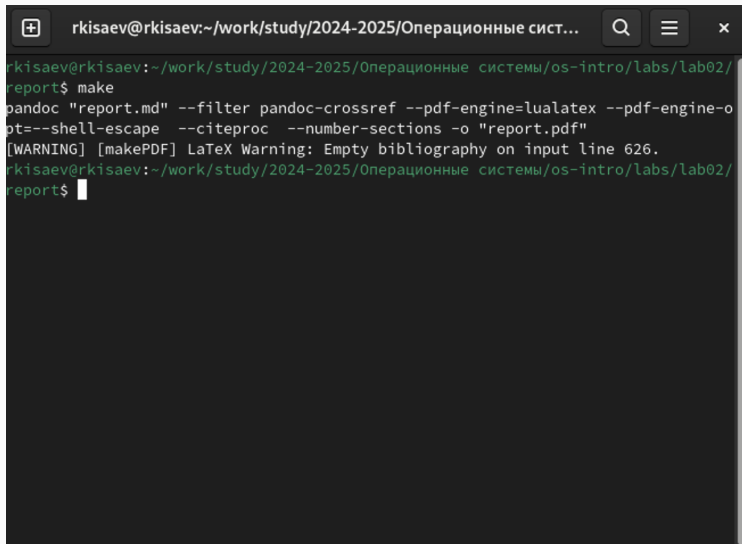
Возможность отката. Система контроля версий позволяет восстанавливать предыдущие версии кода. Это может быть необходимо в случае ошибки или для сравнения с текущей версией. 3

Командная работа. С помощью системы контроля версий удобно организовывать совместную работу над проектом. Каждый участник может выполнять свою часть работы, а система помогает объединять результаты.

2. Объясните следующие понятия СКС и их отношения: **хранилище**, **commit**, история, рабочая копия.

Рис. 4: контрольные вопросы

5. С помощью команды make создаем отчет в форматах docx и pdf



```
rkisaev@rkisaev:~/work/study/2024-2025/Операционные сист...  
report$ make  
pandoc "report.md" --filter pandoc-crossref --pdf-engine=lualatex --pdf-engine-opt=--shell-escape --citeproc --number-sections -o "report.pdf"  
[WARNING] [makePDF] LaTeX Warning: Empty bibliography on input line 626.  
rkisaev@rkisaev:~/work/study/2024-2025/Операционные системы/os-intro/labs/lab02/  
report$
```

Рис. 5: создание отчета

## Выводы

---



Мы научились создавать отчет с помощью легковесного языка разметки Markdown.