

Выполнение лабораторной работы 3

Скрипникова С. Д.

21 февраля 2023

Российский университет дружбы народов, Москва, Россия

Информация

- Скрипникова София Дмитриевна
- Российский университет дружбы народов
- 1132226523@pfur.ru

Этапы работы

1 этап

```

|--
## Front matter
title: "Лабораторная работа 2"
author: "Скрипникова София "

## Generic options
lang: ru-RU
toc-title: "Содержание"

## Bibliography
bibliography: bib/cite.bib
csl: pandoc/csl/gost-r-7-0-5-2008-numeric.csl

## Pdf output format
toc: true # Table of contents
toc-depth: 2
lof: true # List of figures
lot: true # List of tables
fontsize: 12pt
linestretch: 1.5
papersize: a4
documentclass: scrreprt

## I18n polyglossia
polyglossia-lang:
  name: russian
  options:
    - spelling=modern
    - babelshorthands=true
polyglossia-otherlangs:
  name: english

## I18n babel
babel-lang: russian
babel-otherlangs: english

## Fonts
mainfont: PT Serif
```

2 этап

```
babel-lang: russian
babel-otherlangs: english

## Fonts
mainfont: PT Serif
romanfont: PT Serif
sansfont: PT Sans
monofont: PT Mono
mainfontoptions: Ligatures=TeX
romanfontoptions: Ligatures=TeX
sansfontoptions: Ligatures=TeX, Scale=MatchLowercase
monofontoptions: Scale=MatchLowercase, Scale=0.9

## Biblatex
bibtex: true
biblio-style: "gost-numeric"
bibtexoptions:
  - parenttracker=true
  - backend=biber
  - hyperref=auto
  - language=auto
  - autolang=other*
  - citestyle=gost-numeric

## Pandoc-crossref LaTeX customization
figureTitle: "Рис."
tableTitle: "Таблица"
listingTitle: "Листинг"
loftitle: "Список иллюстраций"
lotTitle: "Список таблиц"
lolTitle: "Листинги"

## Misc options
indent: true
header-includes:
  - \usepackage[indentfirst]
  - \usepackage{float} # keep figures where there are in the text
  - \floatplacement{figure}{H} # keep figures where there are in the text
```

Цель работы

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

Выполнение лабораторной работы

В прошлом семестре мною была выполнена привязка `git`, все функционирует исправно (рис. @fig:001).

![Исправное функционирование](image/Снимок лаб1.11.PNG) {#fig:001 width=70%}

Генерирование ключей `PGP` и их настройка (рис. @fig:002).

![Генерирование ключей](image/Снимок лаб1.5.PNG) {#fig:002 width=70%}

Далее выводим список ключей (рис. @fig:003).

![Вывод списков](image/Снимок лаб1.7.PNG) {#fig:003 width=70%}

Копируем наш сгенерированный `PGP` ключ в буфер обмена (рис. @fig:004).

![Копирование](image/Снимок лаб1.8.PNG) {#fig:004 width=70%}

Переходим в настройки `GitHub` и вставляем полученный ключ в поле ввода (рис. @fig:005).


![Копирование](image/Снимок лаб1.9.PNG) {#fig:005 width=70%}

Настройка автоматических подписей (рис. @fig:006).

![Автоматические подписи](image/Снимок лаб1.10.PNG) {#fig:006 width=70%}




Далее вручную я создала репозиторий по шаблону (рис. @fig:007); (рис. @fig:008).

![Репозиторий](image/лаб1.2.PNG) {#fig:007 width=70%}

Открыть ▾ 

report2.md

~\work\study\2022-2023\Операционные системы\os-intro\labs\lab03\report

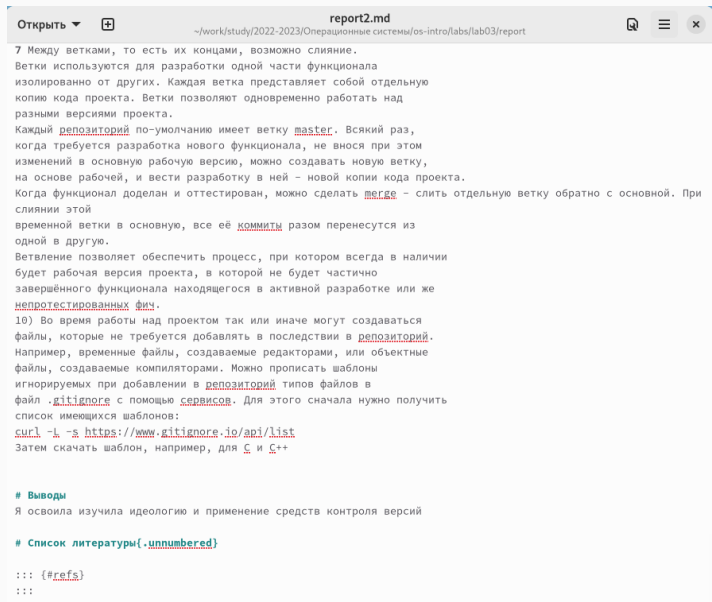
![[Репозиторий рабочий]](image/n1.1.PNG) {#fig:008 width=70%}

#Контрольные вопросы

1) Система контроля версий (Version Control System, VCS) – программное обеспечение для облегчения работы с изменяющейся информацией. VCS позволяет хранить несколько версий одного и того же документа, при необходимости возвращаться к более ранним версиям, определять, кто и когда сделал то или иное изменение, и многое другое. Системы контроля версий (Version Control System, 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. 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.

2) Хранилище (репозиторий) – в нем хранятся все документы вместе с историей их изменения и другой служебной информацией. Рабочую копию необходимо периодически синхронизировать с

5 этап



```
Открыть + report2.md
~/work/study/2022-2023/Операционные системы/os-intro/labs/lab03/report

7 Между ветками, то есть их концами, возможно слияние.
Ветки используются для разработки одной части функционала
изолированно от других. Каждая ветка представляет собой отдельную
копию кода проекта. Ветки позволяют одновременно работать над
разными версиями проекта.
Каждый репозиторий по-умолчанию имеет ветку master. Всякий раз,
когда требуется разработка нового функционала, не внося при этом
изменений в основную рабочую версию, можно создавать новую ветку,
на основе рабочей, и вести разработку в ней – новой копии кода проекта.
Когда функционал доделан и оттестирован, можно сделать merge – слить отдельную ветку обратно с основной. При
слиянии этой
временной ветки в основную, все её коммиты разом перенесутся из
одной в другую.
Ветвление позволяет обеспечить процесс, при котором всегда в наличии
будет рабочая версия проекта, в которой не будет частично
завершённого функционала находящегося в активной разработке или же
непротестированных фич.
10) Во время работы над проектом так или иначе могут создаваться
файлы, которые не требуется добавлять в последствии в репозиторий.
Например, временные файлы, создаваемые редакторами, или объектные
файлы, создаваемые компиляторами. Можно прописать шаблоны
игнорируемых при добавлении в репозиторий типов файлов в
файл .gitignore с помощью сервисов. Для этого сначала нужно получить
список имеющихся шаблонов:
curl -L -s https://www.gitignore.io/api/list
Затем скачать шаблон, например, для C и C++

# Выводы
Я освоила изучила идеологию и применение средств контроля версий

# Список литературы{.unnumbered}

::: {#refs}
:::
```

- Я научилась делать отчеты в формате Markdown