**РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ**

**Факультет физико-математических и естественных наук**

**Кафедра прикладной информатики и теории вероятностей**

**ОТЧЕТ**

**по лабораторной работе № 2**

**Система контроля версии git**

*дисциплина: Архитектура компьютера*

Студент: Обрезкова Анастасия Владимировна

Группа: НММбд-02-22

**МОСКВА**

2022

# Оглавление.

[Оглавление. 2](#_Toc116739170)

[1. Цель работы. 3](#_Toc116739171)

[2. Задание. 3](#_Toc116739172)

[3. Теоретическое введение. 3](#_Toc116739173)

[3.1. Системы контроля версий. Общие понятия. 3](#_Toc116739174)

[3.2. Система контроля версий Git. 5](#_Toc116739175)

[3.3. Стандартные процедуры работы при наличии центрального репозитория. 5](#_Toc116739176)

[4. Выполнение лабораторной работы. 7](#_Toc116739177)

[4.1. Настройка github. 7](#_Toc116739178)

[4.2. Базовая настройка git. 7](#_Toc116739179)

[4.3. Создание SSH ключа. 8](#_Toc116739180)

[4.4. Создание рабочего пространства и репозитория курса на основе шаблона. 9](#_Toc116739181)

[4.5.Создание репозитория курса на основе шаблона. 9](#_Toc116739182)

[4.6. Настройка каталога курса. 11](#_Toc116739183)

[5. Задания для самостоятельной работы. 13](#_Toc116739184)

[6. Вывод. 15](#_Toc116739185)

[7. Список литературы. 15](#_Toc116739186)

[8. Список иллюстраций. 16](#_Toc116739187)

# 1. Цель работы.

Изучить идеологию и применение средств контроля версий. Приобрести практические навыки по работе с системой GIT.

# 2. Задание.

Ознакомиться с системой контроля версий GIT.

# 3. Теоретическое введение.

## 3.1. Системы контроля версий. Общие понятия.

**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.

В классических системах контроля версий используется централизованная модель, предполагающая наличие единого репозитория 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 файлов. 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 измененных файлов, а производить так называемую дельту-компрессию – сохранять только изменения между последовательными версиями, что позволяет уменьшить объем 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.

В отличие от классических, в распределённых системах контроля версий центральный репозиторий не является обязательным.

Среди классических VCS наиболее известны CVS, Subversion, а среди распределённых — Git, Bazaar, Mercurial. Принципы их работы схожи, отличаются они в основном синтаксисом используемых в работе команд.

## 3.2. Система контроля версий Git.

**Система контроля версий Git** представляет собой набор программ командной строки. Доступ к ним можно получить из терминала посредством ввода команды git с различными опциями.

Благодаря тому, что Git является распределенной системой контроля версий, резервную копию локального хранилища можно сделать простым копированием или архивацией.

## 3.3. Стандартные процедуры работы при наличии центрального репозитория.

Работа пользователя со своей веткой начинается с проверки и получения изменений из центрального репозитория(при этом в локальное дерево до начала этой процедуры не должно было вноситься изменений):

git checkout master

git pull git

checkout -b имя\_ветки

Затем можно вносить изменения в локальном дереве и/или ветке.

После завершения внесения какого-то изменения в файлы и/или каталоги проекта необходимо разместить их в центральном репозитории. Для этого необходимо проверить, какие файлы изменились к текущему моменту:

git status

и при необходимости удаляем лишние файлы, которые не хотим отправлять в центральный репозиторий.

Затем полезно просмотреть текст изменений на предмет соответствия правилам ведения чистых коммитов:

git diff

Если какие-либо файлы не должны попасть в коммит, то помечаем только те файлы, изменения которых нужно сохранить. Для этого используем команды добавления и/или удаления с нужными опциями:

git add имена\_файлов

git rm имена\_файлов

Если нужно сохранить все изменения в текущем каталоге, то используем:

git add .

Затем сохраняем изменения, поясняя, что было сделано:

git commit -am "Some commit message"

и отправляем в центральный репозиторий:

git push origin имя\_ветки

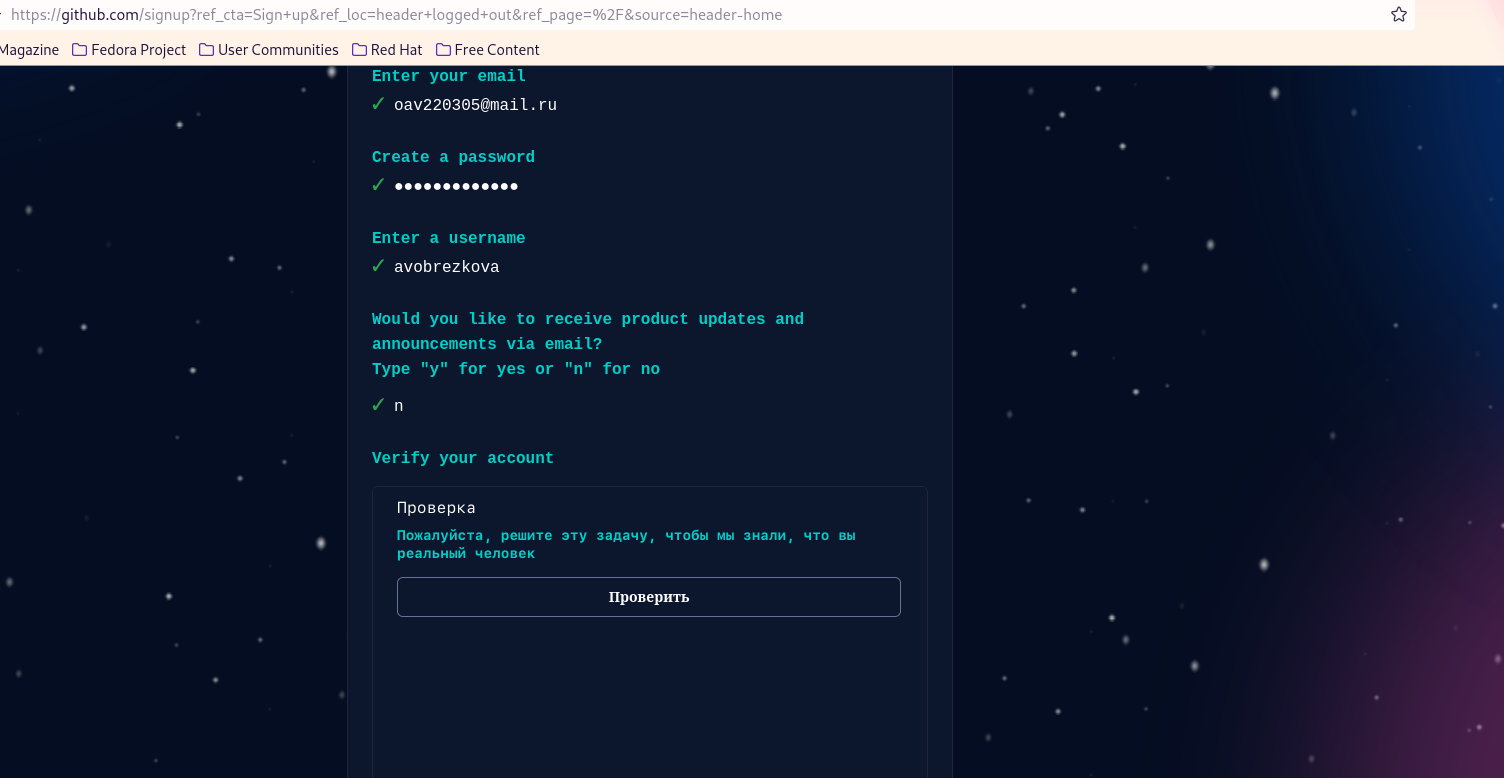
или

git p

# 4. Выполнение лабораторной работы.

## 4.1. Настройка github.

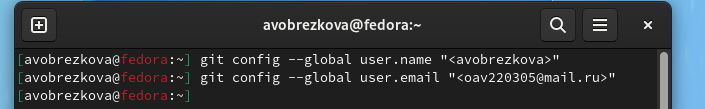
**1.** Создала учетную запись на сайте Github, заполнила основные данные.



### Рис. 4.1.1. Создание учетной записи.

## 4.2. Базовая настройка git.

**1.** Создала предварительную конфигурацию git.



### Рис. 4.2.1. Предварительная конфигурация git.

**2.** Настроила utf-8 в выводе сообщений git.

**4.PNG**

Рис. 4.2.2. Настройка utf-8.

**3.** Задала имя начальной ветки.

**5.PNG**Рис. 4.2.3. Начальная строка.

**4.** Параметр autocrlf.

6.PNG

### Рис. 4.2.4. Параметр autocrlf.

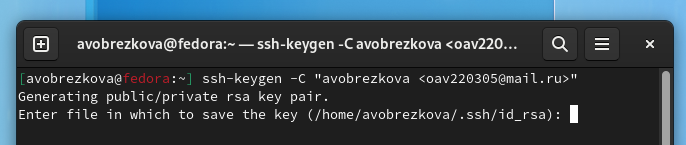
**5.** Параметр safecrlf.

7.PNG

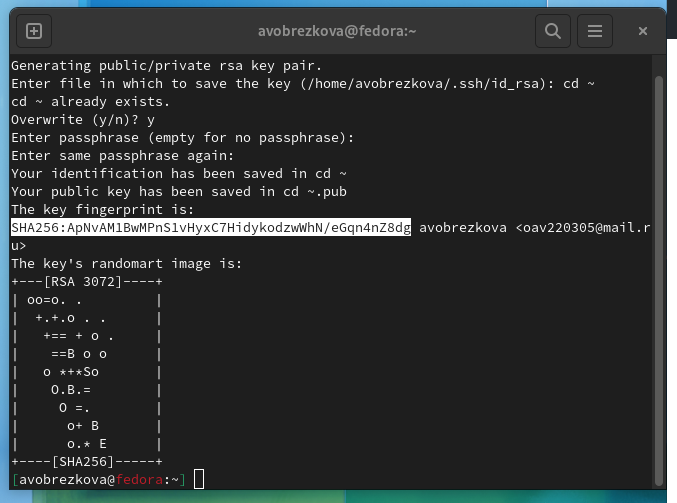
### Рис. 4.2.5. Параметр safecrjf.

## 4.3. Создание SSH ключа.

**1.** Генерирование ключей.

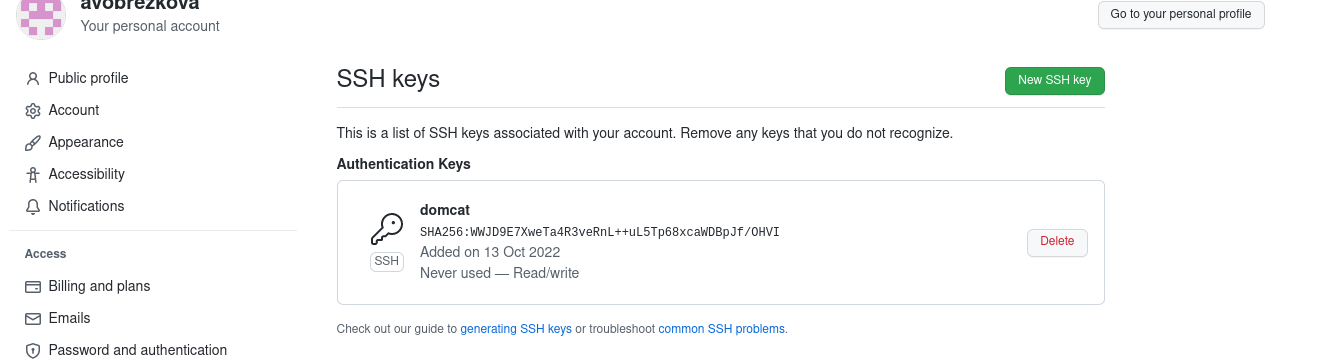


### Рис. 4.3.1.1. Генерирование ключей.



### Рис. 4.3.1.2. Генерирование ключей.

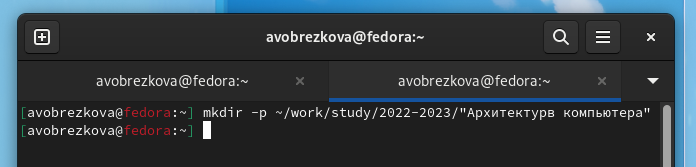
**2.** С помощью команды cat скопировала из локальной консоли ключ в буфер обмена и вставила его в появившееся на сайте поле. Указала для ключа имя и создала его.

****

### Рис. 4.3.2. Создание ключа.

## 4.4. Создание рабочего пространства и репозитория курса на основе шаблона.

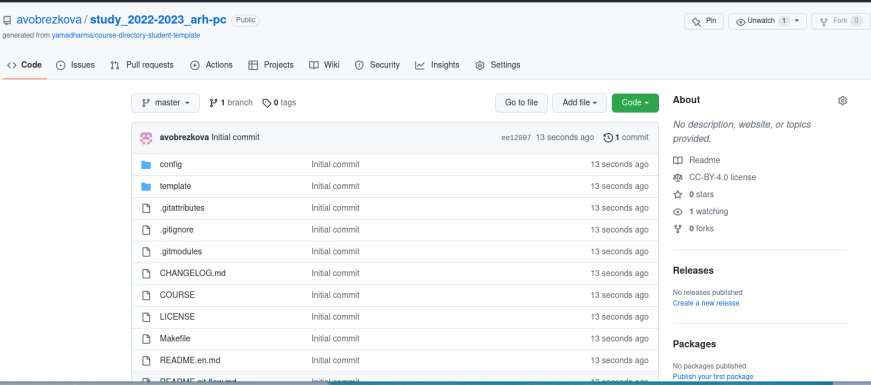
**1.** Создала каталог для предмета «Архитектура компьютера».



### Рис. 4.4.1. Создание каталога для предмета.

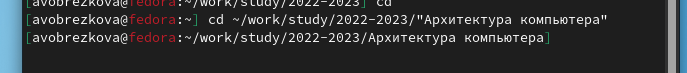
## 4.5.Создание репозитория курса на основе шаблона.

**1.** Перешла на страницу репозитория с шаблоном курса, задала имя репозитория и создала его.



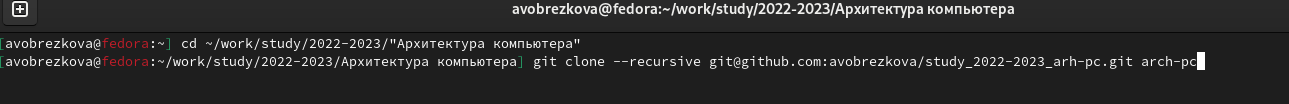
### Рис. 4.5.1. Страница репозитория.

**2.** Перешла в каталог курса.

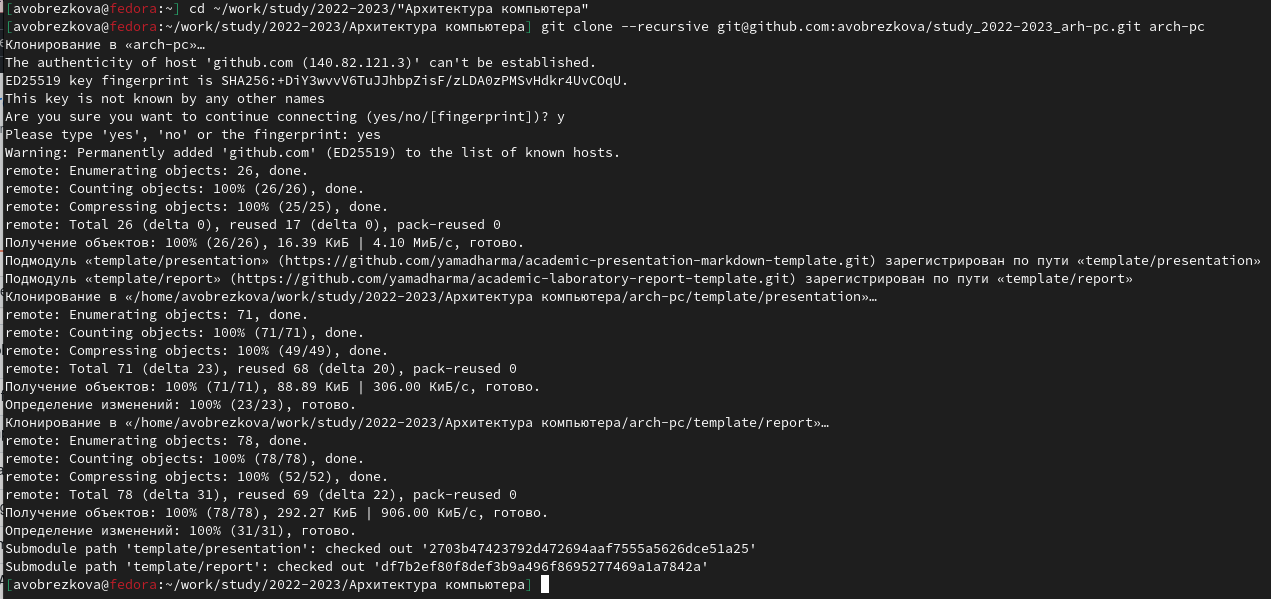


### Рис. 4.5.2. Переход в каталог курса.

**3.** Клонировала созданный репозиторий.



### Рис. 4.5.3. Клонирование репозитория.



### Рис. 4.5.3. Результат копировани

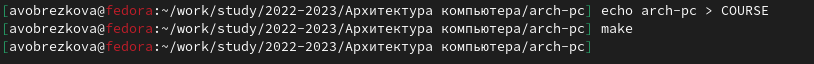
## 4.6. Настройка каталога курса.

**1.** Перешла в каталог курса и удалила лишние файлы.

21.PNG

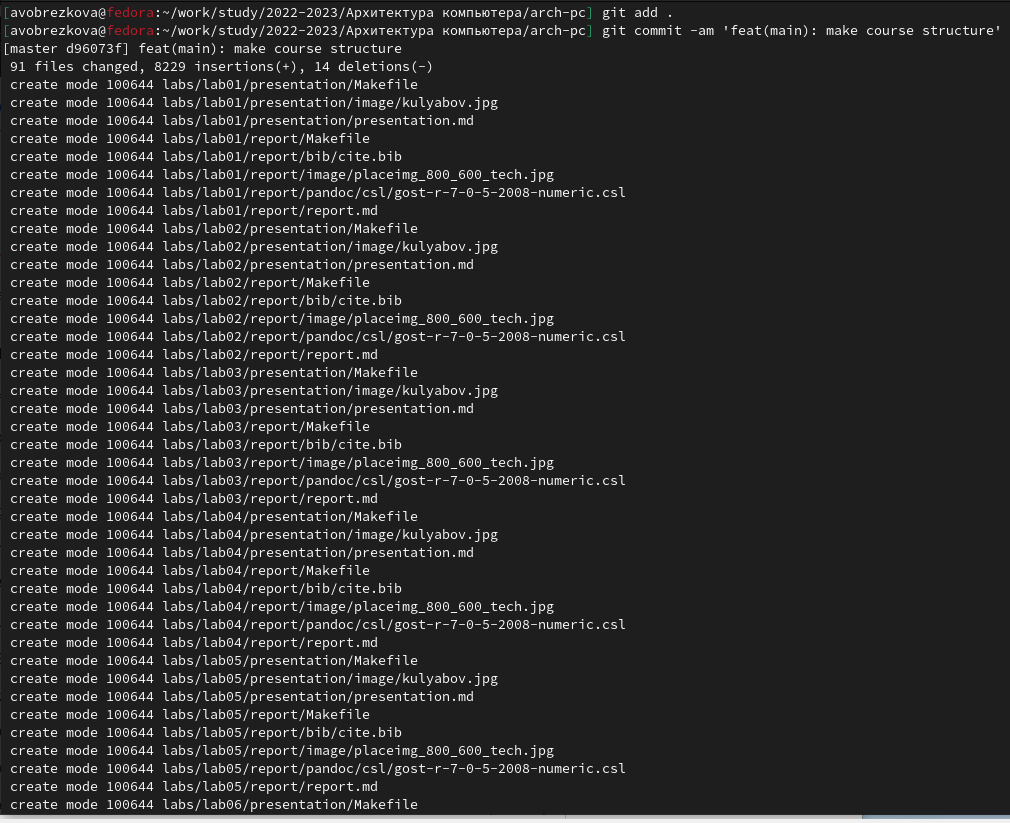
### Рис. 4.6.1. Удаление лишних файлов.

**2.** Создала необходимые каталоги.

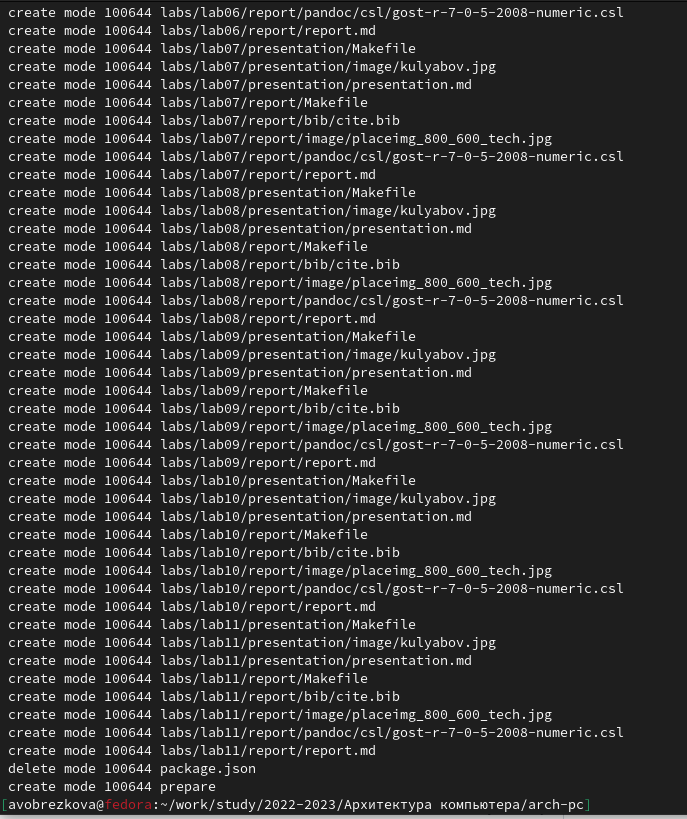


### Рис. 4.6.2. Создание каталогов.

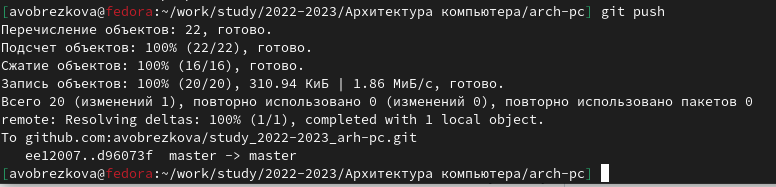
**3.** Отправила файлы на сервер.



### Рис. 4.6.3.1. Отправка файлов.



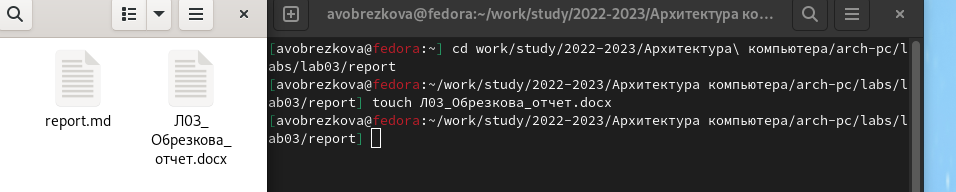
### Рис. 4.6.3.2. Отправка файлов.



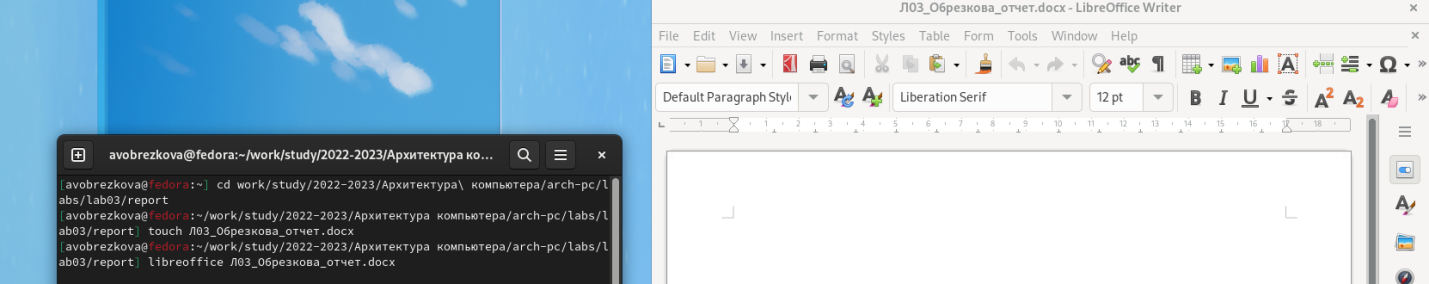
### Рис. 4.6.3.3. Отправка файлов.

# 5. Задания для самостоятельной работы.

**1.** Создала отчет по выполнению лабораторной работы в соответствующем каталоге рабочего пространства.

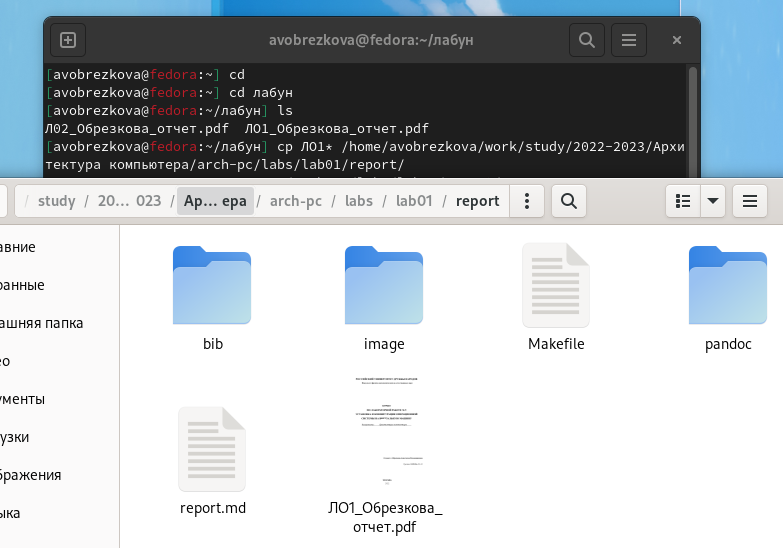


### Рис. 5.1.1. Создание файла отчета.

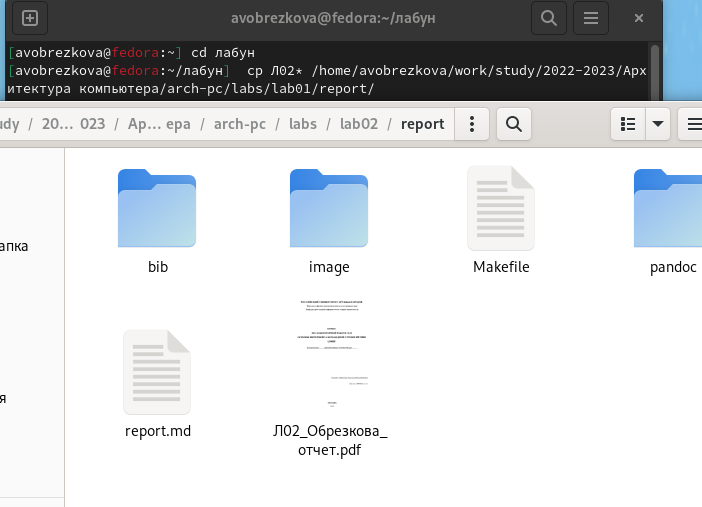


### Рис. 5.1.2. Начало создание отчета.

**2.** Скопировала отчеты по выполнению предыдущих лабораторных работ в соответствующие каталоги. Отчеты выполнялись на виртуальной машине, поэтому я просто перенесла их из одного каталога в другой.

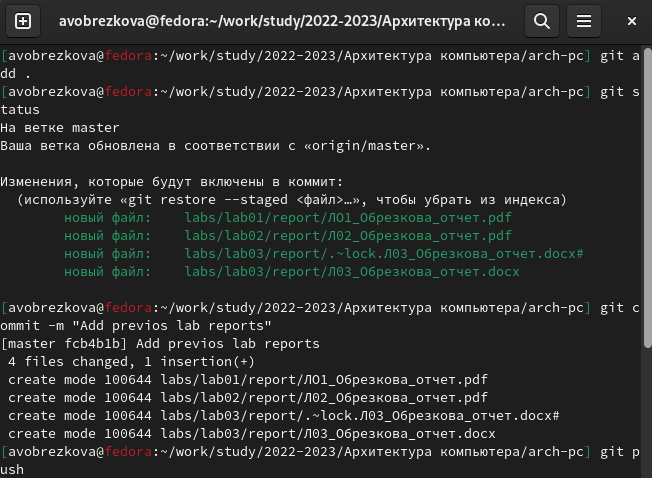


### Рис. 5.2.1. Перенос Лабораторной работы №1.



### Рис. 5.2.2. Перенос лабораторной работы №2.

**3.** Загрузка файлов на github.



### Рис. 5.3.1. Добавленные изменения.

### 31.PNG

### Рис. 5.3.3. Запустим изменения.

Данные изменения можно проверить по ссылке: [study\_2022-2023\_arh-pc/labs at master · avobrezkova/study\_2022-2023\_arh-pc (github.com)](https://github.com/avobrezkova/study_2022-2023_arh-pc/tree/master/labs)

# 6. Вывод.

Я ознакомилась с системой контроля версий GIT. Создала репозиторий из шаблона сервиса GitHub и внесла в него необходимые изменения. Я освоила базовые команды утилита git.

# 7. Список литературы.

1. [Архитектура ЭВМ (rudn.ru)](https://esystem.rudn.ru/pluginfile.php/1584255/mod_resource/content/1/%D0%9B%D0%B0%D0%B1%D0%BE%D1%80%D0%B0%D1%82%D0%BE%D1%80%D0%BD%D0%B0%D1%8F%20%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%B0%20%E2%84%963.pdf)

2. [GitHub - yamadharma/academic-laboratory-report-template: Report template on laboratory work](https://github.com/yamadharma/academic-laboratory-report-template)

# 8. Список иллюстраций.

[Рис. 4.1.1. Создание учетной записи. 7](#_Toc116741169)

[Рис. 4.2.1. Предварительная конфигурация git. 7](#_Toc116741170)

[Рис. 4.2.4. Параметр autocrlf. 8](#_Toc116741171)

[Рис. 4.2.5. Параметр safecrjf. 8](#_Toc116741172)

[Рис. 4.3.1.1. Генерирование ключей. 8](#_Toc116741173)

[Рис. 4.3.1.2. Генерирование ключей. 8](#_Toc116741174)

[Рис. 4.3.2. Создание ключа. 9](#_Toc116741175)

[Рис. 4.4.1. Создание каталога для предмета. 9](#_Toc116741176)

[Рис. 4.5.1. Страница репозитория. 9](#_Toc116741177)

[Рис. 4.5.2. Переход в каталог курса. 10](#_Toc116741178)

[Рис. 4.5.3. Клонирование репозитория. 10](#_Toc116741179)

[Рис. 4.5.3. Результат копировани 10](#_Toc116741180)

[Рис. 4.6.1. Удаление лишних файлов. 11](#_Toc116741181)

[Рис. 4.6.2. Создание каталогов. 11](#_Toc116741182)

[Рис. 4.6.3.1. Отправка файлов. 11](#_Toc116741183)

[Рис. 4.6.3.2. Отправка файлов. 12](#_Toc116741184)

[Рис. 4.6.3.3. Отправка файлов. 12](#_Toc116741185)

[Рис. 5.1.1. Создание файла отчета. 13](#_Toc116741186)

[Рис. 5.1.2. Начало создание отчета. 13](#_Toc116741187)

[Рис. 5.2.1. Перенос Лабораторной работы №1. 13](#_Toc116741188)

[Рис. 5.2.2. Перенос лабораторной работы №2. 14](#_Toc116741189)

[Рис. 5.3.1. Добавленные изменения. 14](#_Toc116741190)

[Рис. 5.3.3. Запустим изменения. 15](#_Toc116741191)