Отчёт по лабораторной работе №2

Дисциплина: архитектура компьютера

Неустроева Ирина

Содержание

# 1 Цель работы

Получить практические навыки по работе с системой git, изучить идеологию и применение средств контроля версий.

# 2 2 Задание

1. Цель работы
2. Теоретическое введение
3. Выполнение лабораторной работы
4. Вывод

# 3 Теоретическое введение

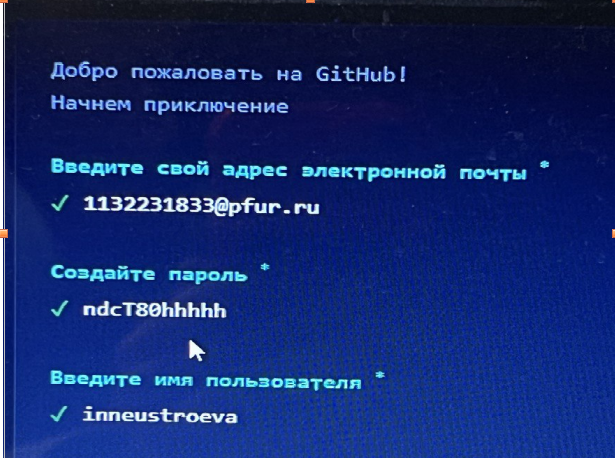
Системы контроля версий (Version Control System, VCS) применяются при работе нескольких человек над одним проектом. Обычно основное дерево проекта хранится в локальном или удалённом репозитории, к которому настроен доступ для участников проекта. При внесении изменений в содержание проекта система контроля версий позволяет их фиксировать, совмещать изменения, произведённые разными участниками проекта, производить откат к любой более ранней версии проекта. 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 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 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. 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.

| Команда | Описание |
| --- | --- |
| git init | Создание основного дерева репозитория |
| git pull | Получение обновлений (изменений) текущего дерева из центрального репозитория |
| git push | Отправка всех произведённых изменений локального дерева в центральный репозиторий |
| git status | Просмотр списка изменённых файлов в текущей директории |
| git diff | Просмотр текущих изменения |
| git add . | Добавить все изменённые и/или созданные файлы и/или каталоги |
| git add файл | Добавить конкретные изменённые и/или созданные файлы и/или каталоги |
| git rm файл | Удалить файл и/или каталог из индекса репозитория (при этом файл и/или каталог остаётся в локальной директории) |
| git commit -am | Сохранить все добавленные изменения и все изменённые файлы |
| git checkout -b | Создание новой ветки, базирующейся на текущей |
| git checkout | Переключение на некоторую ветку |
| git push origin | Отправка изменений конкретной ветки в центральный репозиторий |
| git merge --no-ff | Слияние ветки с текущим деревом |
| git branch -d | Удаление локальной уже слитой с основным деревом ветки |
| git branch -D | Принудительное удаление локальной ветки |
| git push origin | Удаление ветки с центрального репозитория |

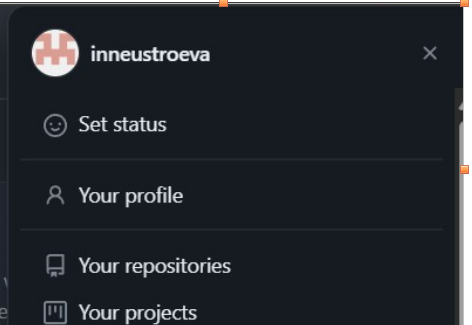
# 4 Выполнение лабораторной работы

### 4.0.1 Настройка GitHub

*1* Переходим на сайт GitHub и создаю учетную запись, заполняя основные данные (рис.@fig:006)



*3* Открываю терминал на вертуальной машине и задаю предварительную конфигурацию git, введя команду git config –global user.name “”, указывая свое имя и команду gitc onfig –global user.email “work@mail”, указывая свою электронную почту (рис.@fig:007)



*4* Настроим utf-8 в выводе сообщений git следующей командой: git config –global core.quotepath false (рис.@fig:009)

Настройка utf-8

*5* Далее задаю имя “master” для начальной ветки, введя команду: git config –glob init.defaultBranch master (рис.@fig:010)

Именование начальной ветки.

*6* Следующим шагом задаю параметр autocrlf со значением input следующей командой: git config –global core.autocrlf input, чтобы все переводы строк текстовых файлов в главном репозитории одинаковыми (конвертирую CRLF в LF только при коммитах. CR и LF – это символы, которые используют для обозначения разрыва строки в текстовых файлах (рис.@fig:011)

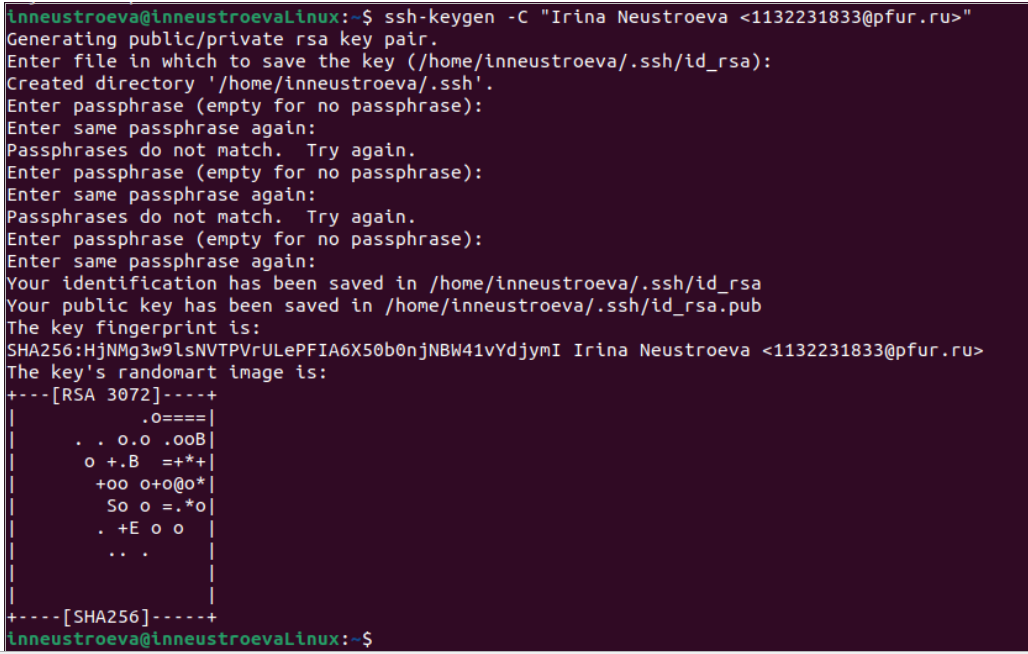
Параметр autocrlf со значением inpu

*7* Задаю параметр safecrlf со значением warn, таким образом Git будет проверять является ли окончаний строк обратимым для текущей настройки core.autocrlf, будет печатать предупреждение, но коммиты с необратимым переходом принимаются (рис.@fig:012)

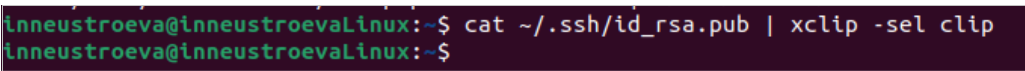
Параметр safecrllf со значением warn

### 4.0.2 Создание SSH ключа.

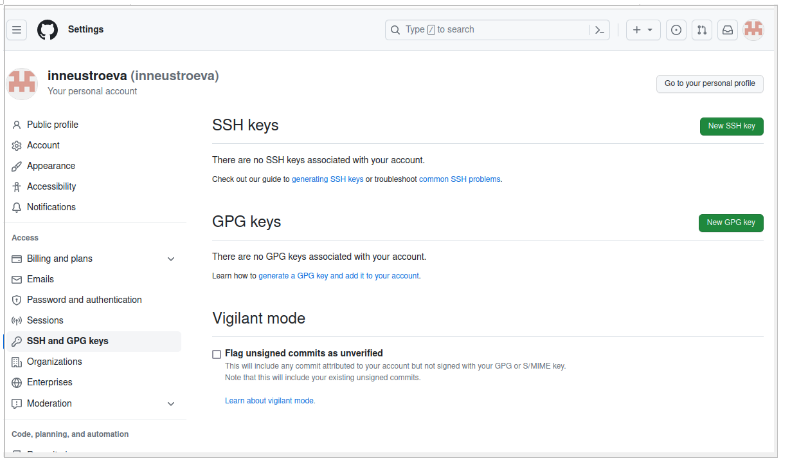
*1* Для последующей идентификации пользователя на сервере репозиториев необходимо сгенерировать приватный и открытый ключи, следующей командой: ssh- keygen -C “Имя Фамилия, почта”, указывая свои данные, ключ автоматически сохранится в каталоге ~/.ssh/ (рис.@fig:013)



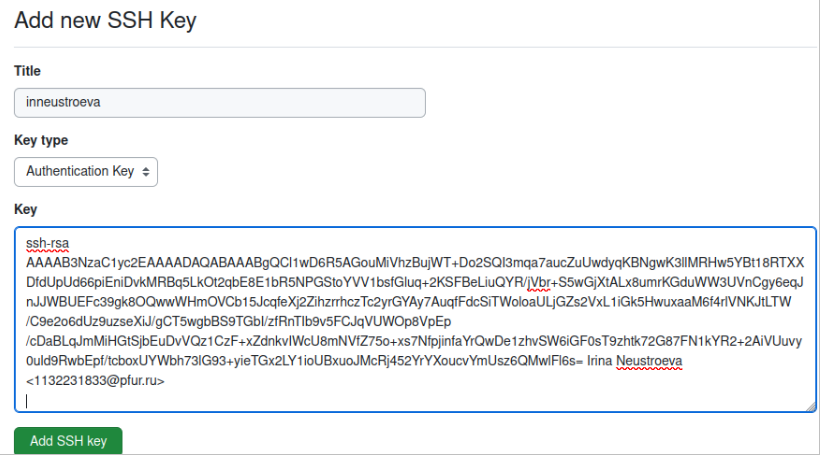
*2* Копирую открытый ключ из директории с помощью утилиты xclip(рис.@fig:014)



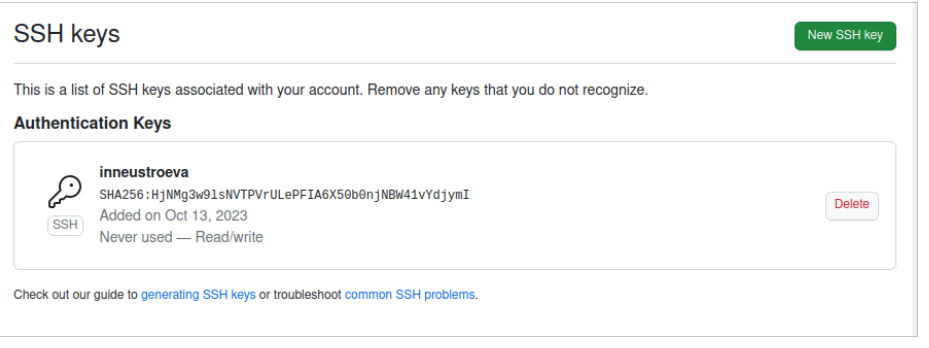
*3* Затем перехожу на сайт GitHub В своем профилье выбираю в боковом меню «SSH and GPG keys», нажимаю кнопку «New SSH key»(рис.@fig:015)



*4* Далее в поле Title указываю свое имя для ключа. Вставляю скопированный ключ в поле «Key». Нажимаю «Add SSH-key», чтобы завершить добавление ключа(рис.@fig:016)

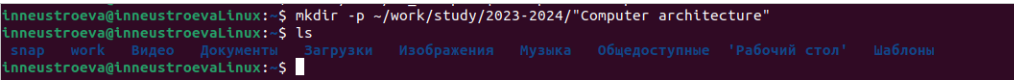


*5* Убедились, что создали ключ(рис.@fig:017)



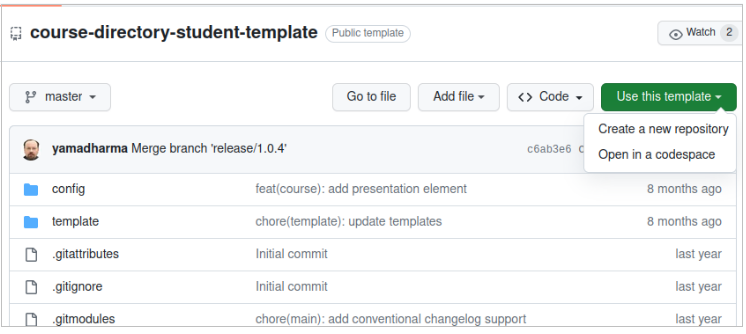
### 4.0.3 Создание рабочего пространства и репозитория курса на основе шаблона

*1* открываю терминал, создаю рабочее пространство, с помощью команды mkdir и ключа -p создаю каталог для предмета Архитектура компьютера, по средством ввода следующей команды: ~/work/study/2023-2024/“Архитектура компьютера” Далее проверяю с помощью команды ls на создание необходимого мне каталога.(рис.@fig:018)

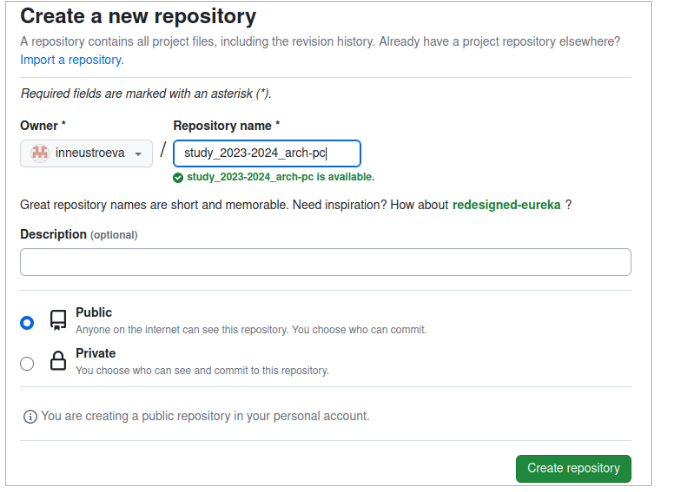


### 4.0.4 Создание репозитория курса на основе шаблона

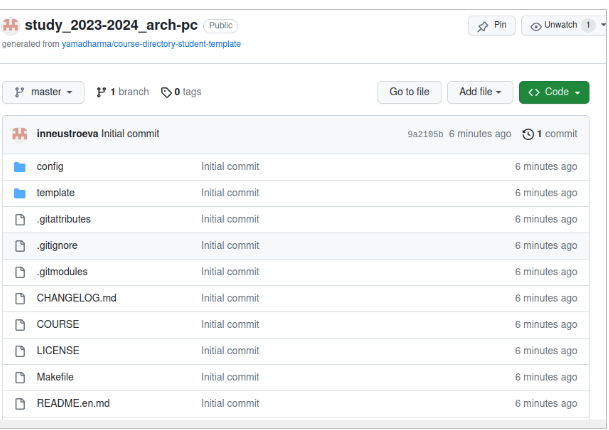
*1* Перешла на страницу репозитория с шаблоном курса по адресу https://github.com/yamadharma/course-directory-student-template. Далее выбираю «Use this template», для того чтобы использовать этот шаблон (рис.@fig:019)



*2* В открывшемся окне задаю имя репозитория (Repository name): study\_2023-2024\_arch- pc и создаю репозиторий, нажимаю на кнопку «Create repository»(рис.@fig:020)



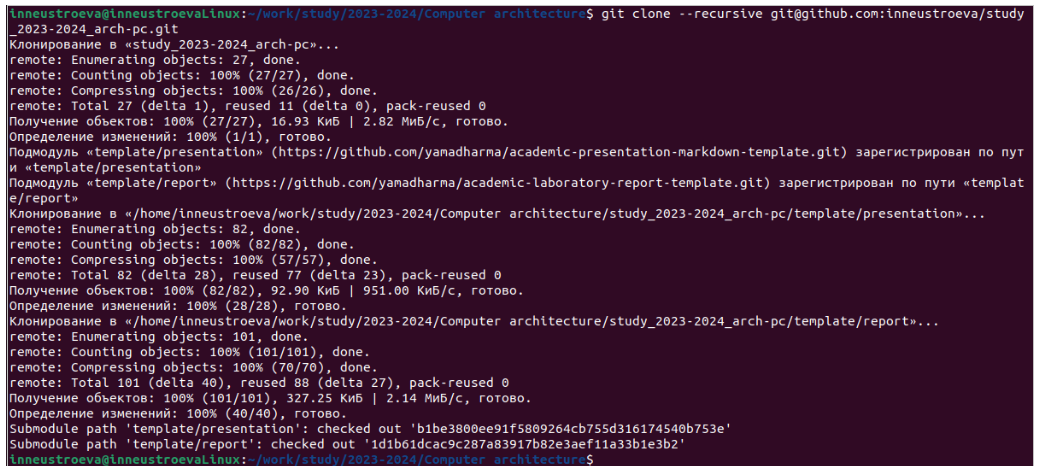
*3* Проверяем, что репозиторий study\_2023–2024\_arhpc успешно создан (рис.@fig:021)



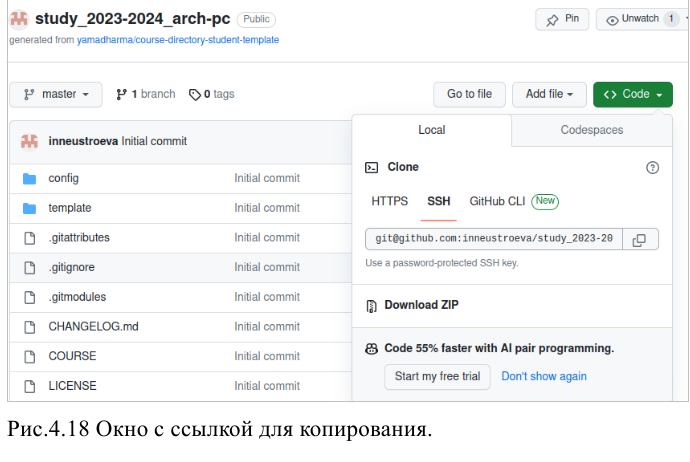
*4* Открыла терминал и с помощью команды cd перешла в созданный каталог курса (рис.@fig:022)

перемещение в дерикторию созданого курса

*5* Следующим шагом, клонирую созданный репозиторий с помощью команды git clone –recursive git@github.com: inneustroeva/study\_2023–2024\_arh-pc.git arch-pc (рис.@fig:023)



*6* Затем на странице созданного репозитория копирую ссылку, перейдя в окно «code», далее выбрав в окне вкладку «SSH» (рис.@fig:024)



### 4.0.5 Настройка каталога курса

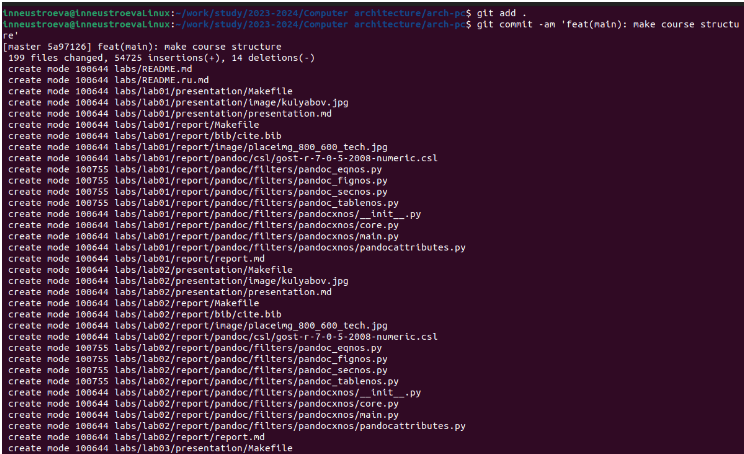
*1* Перехожу в каталог arch-pc с помощью команды cd (рис.@fig:025)

переход в каталог курса.

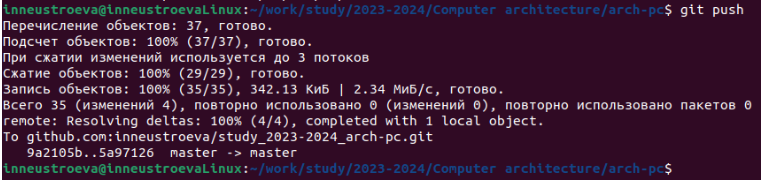
*2* Далее создаю необходимые каталоги (рис.@fig:026)

создание каталогов.

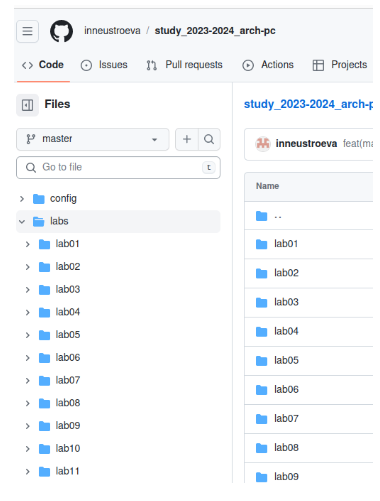
*3* Отправляю созданные файлы с локального репозитория на сервер: добавляю все созданные каталоги с помощью git add, сохраняю изменения на сервере с помощью git commit (рис.@fig:027)



*4* Отправляю все на сервер с помощью команды git push (рис.@fig:028)

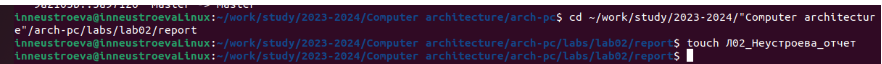


*5* Проверяю правильность создания иерархии рабочего пространства в локальном репозитории на сайте github (рис.@fig:029)

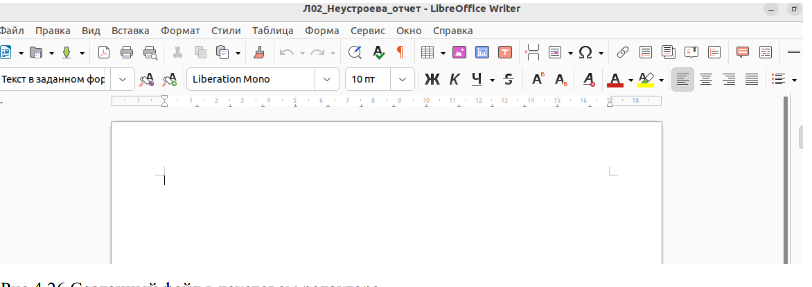


### 4.0.6 Выполнение заданий для самостоятельной работы

*1* Перехожу в директорию labs/lab02/report с помощью команды cd. Создаю в каталоге файл для отчета по лабораторной работе с помощью утилиты touch (рис.@fig:030)



*2* Открываю текстовый редактор LibreOffice Writer, через меню приложений. И открываю в нём созданный файл (рис.@fig:031)



*3* Далее перехожу в каталог lab01/report (рис.@fig:032)

переход в каталог lab01.

*4* Следующим шагом проверяю нахождение файла первой лабораторной работы, файл находиться в Загрузках (рис.@fig:033)

проверка нахождения файла.

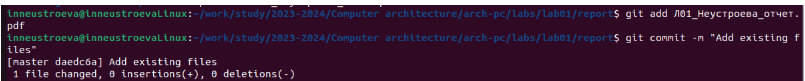
*5* Копирую первую лабораторную работу, с помощью команды cp, и перемещаю её в нужный нам каталог (рис.@fig:034)

Копирование файла Л01_Неустроева_отчет.pdf

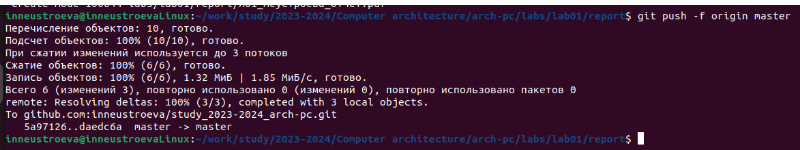
*6* Проверяем правильность выполнения с помощью команды “ls” (рис.@fig:035)

Проверка местонахождения файла Л01_Неустроева_отчет.pdf.

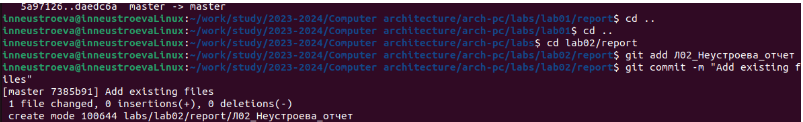
*7* Добавляю файл Л01\_Неустроева\_отчет.pdf на сервер, с помощью команды git add и сохраняю изменения на сервере с помощью git commit (рис.@fig:036)



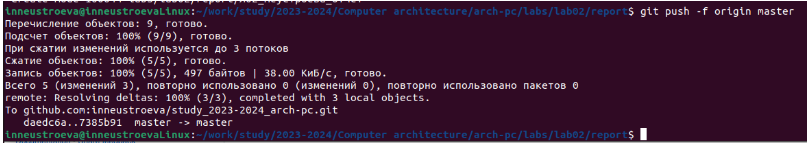
*8* Отправляю в центральный репозиторий сохраненные изменения командой git push -f origin master (рис.@fig:037)



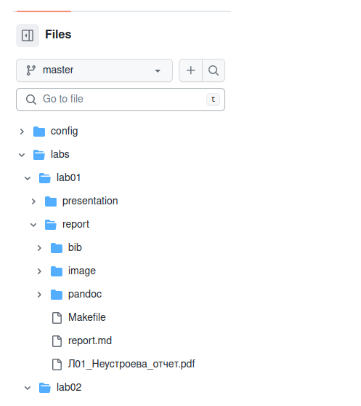
*9* То же самое делаю для второй лабораторной работы: перехожу в директорию labs/lab02/report с помощью cd, добавляю с помощью git add нужный файл Л02\_Неустроева\_отчет, сохраняю изменения с помощью git commit (рис.@fig:038)

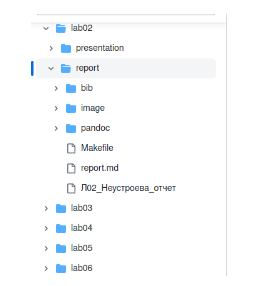


*10* Отправляю в центральный репозиторий сохраненные изменения командой git push -f origin master (рис.@fig:039)



*11* Проверяю на сайте GitHub правильность выполнения заданий. Вижу, что отчет по лабораторной работе номер 1 находятся в соответствующем каталоге (рис.@fig:040) и отчет по лабораторной работе номер 2 тоже находятся в соответствующем каталоге (рис.@fig:041)





# 5 Выводы

Приобрела практические навыки по работе с сиcтемой git. Изучила идеологию и применение средств контроля версий