Отчёта по лабораторной работе №2

Дисциплина: архитектура компьютера

Худдыева Дженнет

Содержание

# 1 Цель работы

Целью данной работы является приобретение практических навыков работы с операционной системой на уровне командной строки (организация файловой системы, навигация по файловой системе, создание и удаление файлов и директорий.

# 2 Задание

1.Настройка Github. 2.Базовая настройка Git. 3.Создание SSH-ключа. 4.Создание рабочего пространства и рнпозитория курса на основе шаблона. 5.Создание репозитория курса на основе шаблона. 6.Настройка каталога курса. 7.Выполнение заданий для самостоятельной работы.

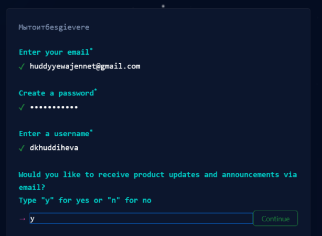
# 3 Теоретическое введение

Системы контроля версий (Version Control System,VCS) применяются при работе нескольких человек над одним проектом.Обычно основное дерево проекта хранится в локальном или удалённым репозитории, к которому настроен доступ доступ для участников проекта.При внесении изменений в содержание проекта система контроля версий позволяет их фиксировать,совмещать изменения произведённые разными участниками проекта,производит откат к любой более ранней версии проекта,если это требуется. В классических системах контроля версий используется централизованная модель,предпологающая 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 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 хра- нимых данных.Системы контроля версий поддерживают возможность отслеживания и разрешения конфликтов,которые могут возникнуть при работе нескольких человек над одним файлом.Можно объединить изменения,сделанные разными участниками,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 припятствует изменению рабочей копии файла средствами файловой системы ОС,обеспечивая таким образом привилегированный доступ только одному пользователю,работающему с файлом.Системы контроля версий также могут обес- печивать 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.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.Система контроля версий Git представляет собой набор программ командной строки. Доступ к ним можно получить из терминала посредством ввода команды git с различными опциями.Работа пользователя со своей веткой начинается с проверки и получения изменений из центрального репозиитория (при этом в локальное дерево до начала этой про- цедуры не должно было вноситься изменений ).После заввершения внесения какого-то изменения в файлы или каталогга необходимо разместить их в ценнтральном репоозитории.

# 4 Выполнение лабораторной работы

##Настройка Github

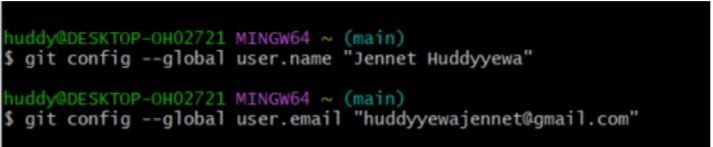
Создаю учетную запись на сайте Github.Далее я заполнила основные данные учетной записи.



Заполнение данных

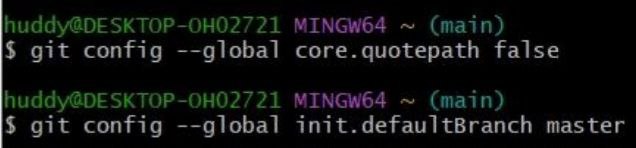
###Базовая настройка git

Открывая виртуальную машину,открываю терминал и делаю предварительную конфигурацию git.Ввожу команду git config-global user.name ” ” , указываю в ней свою электронную почту git config-global user.email “work@mail”.



Предварительная конфигурация git

Настраиваю utf-8 в выводе сообщений git для корректного отоброжение символов.Задаю имя “master” дояначальной ветки



Настройка кодировки

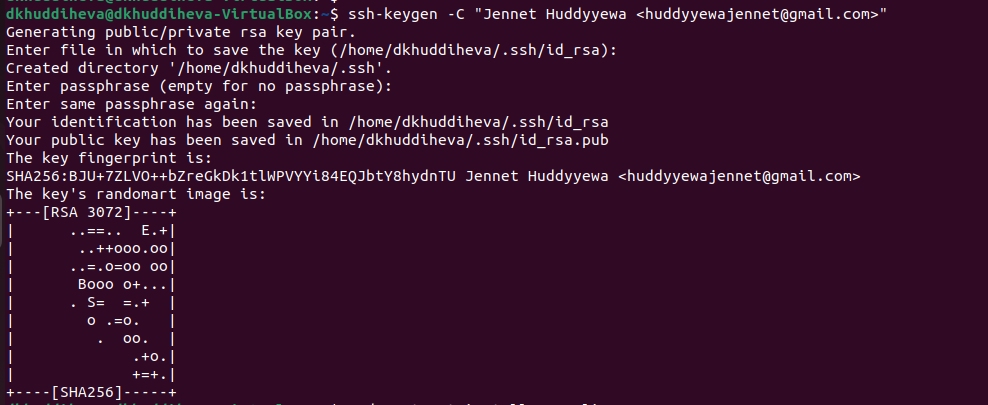
Задаю параметр autocrlf со значением input.Затем задаю параметр safecrlf со значением warn.



параметр autocrlf,safecrlf

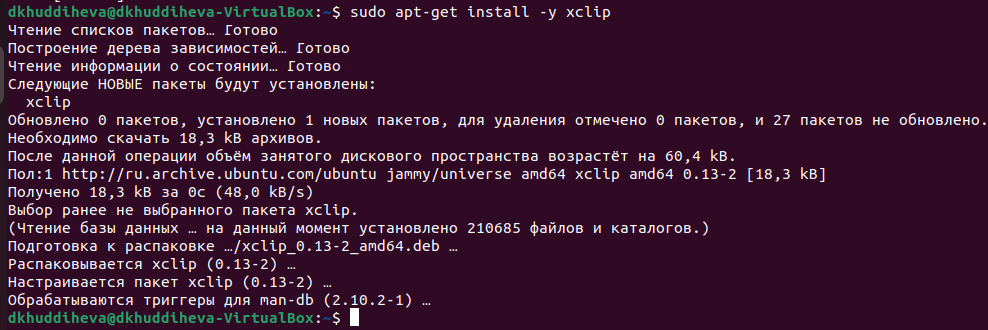
##Создание SSH-ключа

Ввожу команду ssh-keygen -C “Имя\_фамилия,work@mail”, указываю свою имя и фамилию,а так же электронную почту



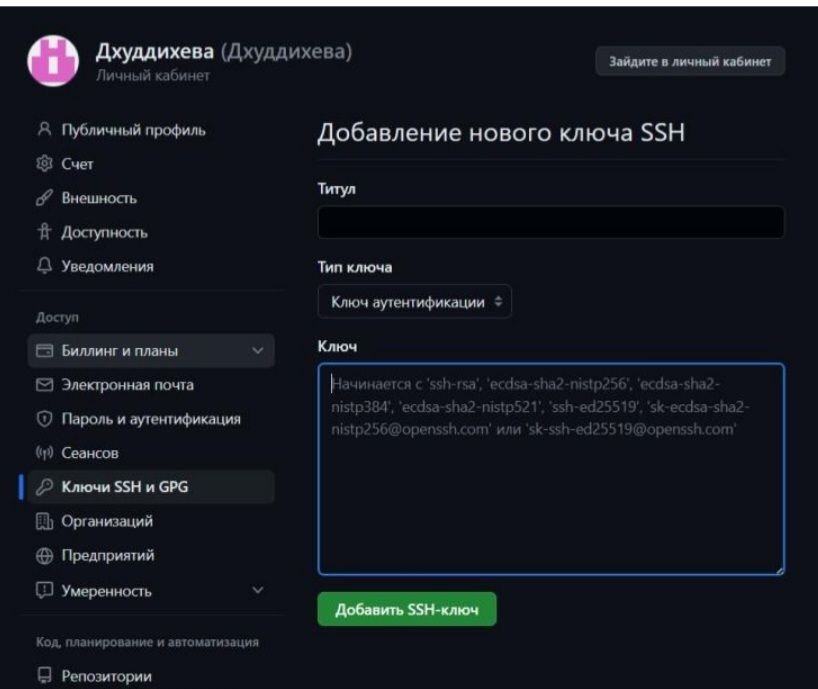
Генерация SSH-ключа

Xclip-утилита позволяющая скопировать любой текст через терминал.Устанавливаю xclip с помощью команды apt-get install с ключом,введя с начало команду sudo



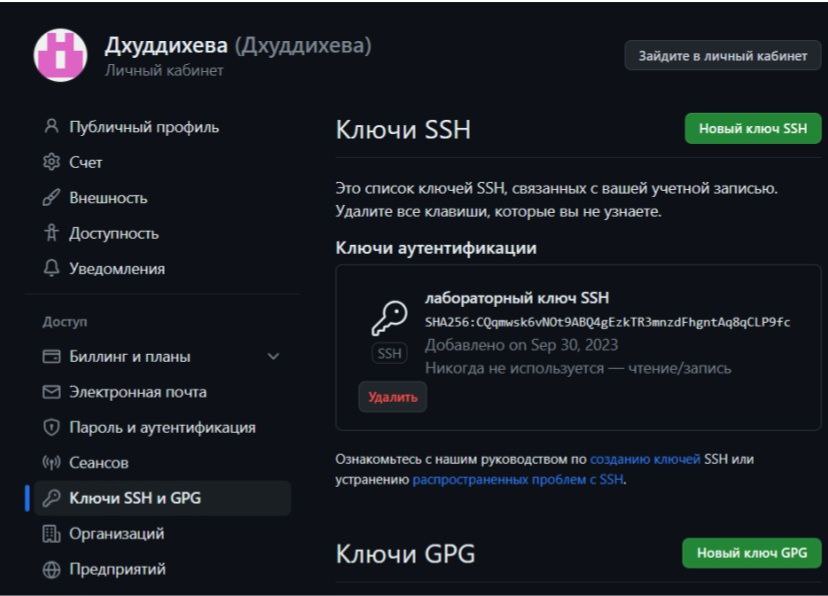
Установка утилиты xclip

Копирую открытый ключ из директории в которой он был сохранён,с помощью утилиты xclip.Открываю свой профиль и выбираю страницу “SSH and GPS keys”



Окно SSH and GPS keys

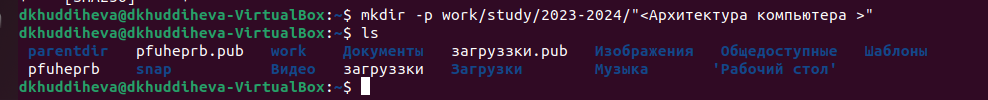
Вставляю скопированный ключ в поле “key”.Title указываю имя для ключа.Нажимаю “ADD SSH-key”чтобы завершить добавление ключа



Добавление ключа

##Создание рабочего пронстранства и репозитория курса на основе шаблона

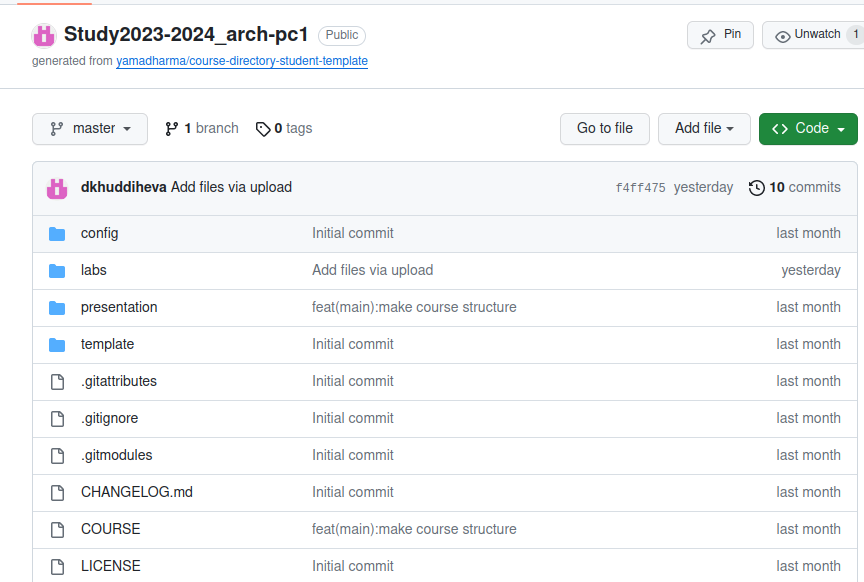
Открываю терминал.Создаю директорию,рабочего пронстранства с помощью утилиты mkdir,благодаря ключу -p создаю все директории после домашней ~/work/study/2023-2024/“Архитектура компьютера”.Далее с помощью ls проверяю.



Создание рабочего пространства

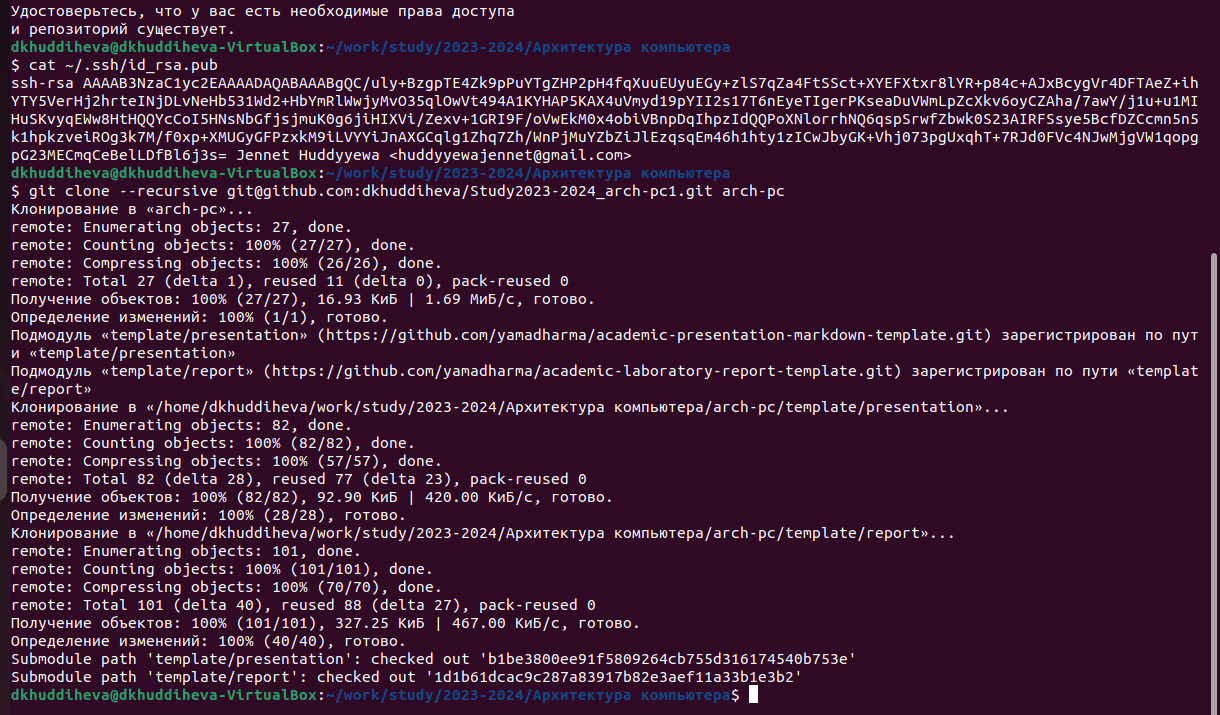
##Создание репозитория курса на основе шаблона

В браузере перехожу на страницу репозитория с шаблоном курса по адресу https://github.com/yamadharma/course-directory-student- -template.Далее выбираю “Use this template”,чтобы использовать этот шаблон для своего репозитория



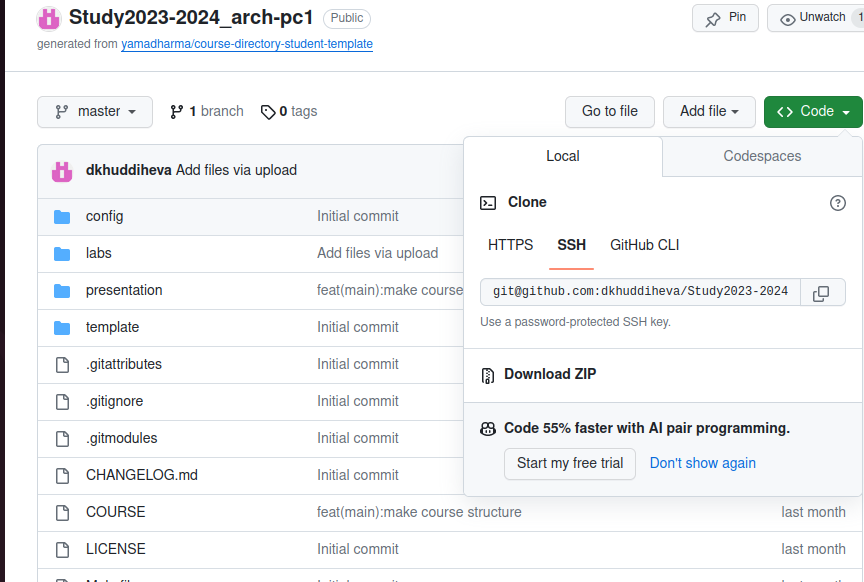
Страница шаблона для репозиторий

Через терминал перехожу в созданной каталог курса с помощью утилиты cd



Перемещение между директориями

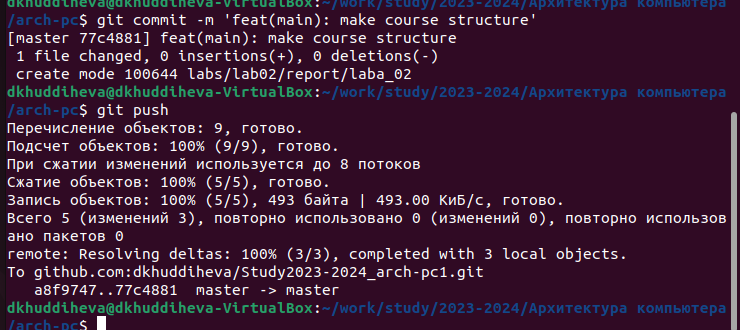
Копирую ссылку для клонирования на странице созданного репозитория,сначало перейдя в окно “code”,далее выбрав в окно вкладку “SSH”



Клонирование репозитория

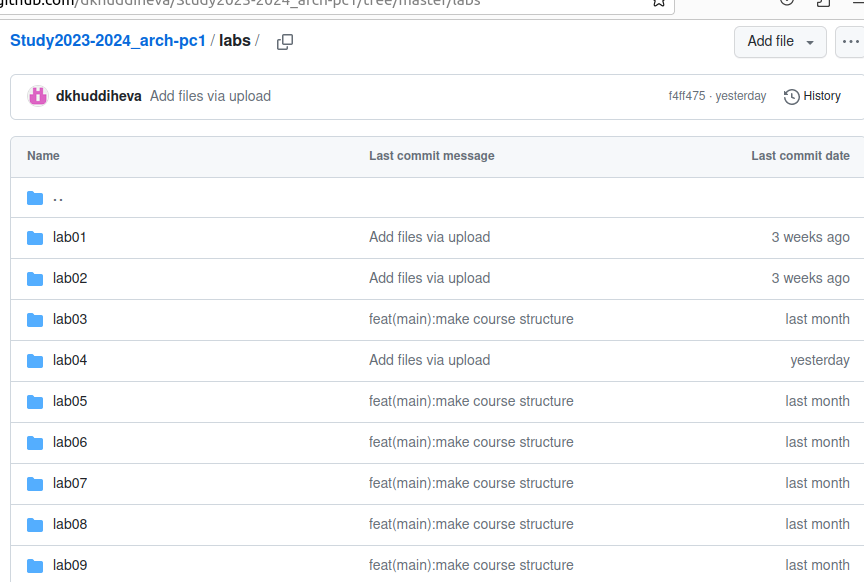
##Настройка каталога курса

Перехожу в каталог arch-pc с помощью утилиты cd. Удаляю лишние файлы с помощью утилиты rm. Создаю необходимые каталоги.Отправляю созданные каталоги с локального репозитория на сервер.Добавляю все созданные каталоги с помощью git add.,комментирую и сохраняю изменени на сервере как добавление курса с помощью.Отправляю всё на сервер с помощью git push.



Перемещение между директориями и удаление файла

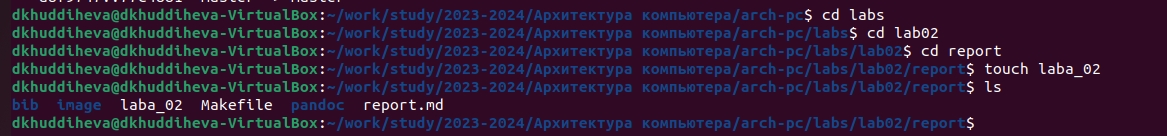
Проверяю правильность выполнения работы сначало на самом сайте



Страница репозитория

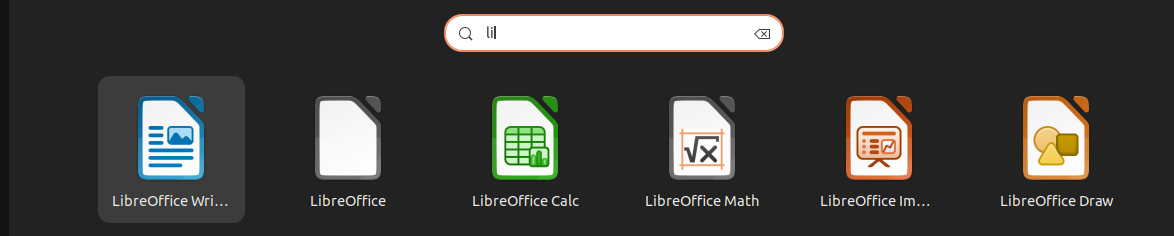
##Выполнение заданий для самостоятельной работы

1.Перехожу в директорию labs/lab02/report с помощью cd.Создаю в каталоге файл для отчёта лабораторной работе с попмощью утилит touch



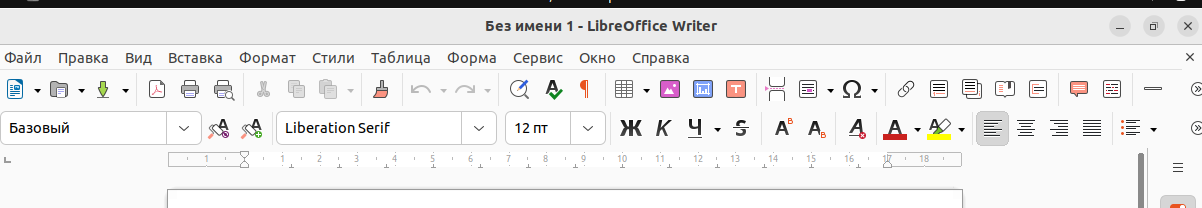
Создание файла

Оформить отчёт я смогу в текстовом процессоре LibreOficce Writer, найдя его в меню преложений



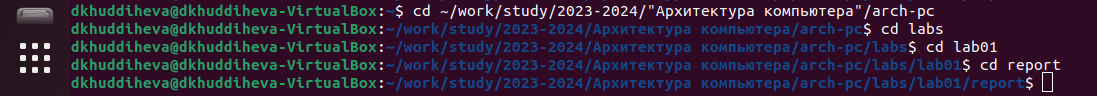
Меню приложений

После открытия текстового пройессора открываю в нём созданный файл и могу в нём работать



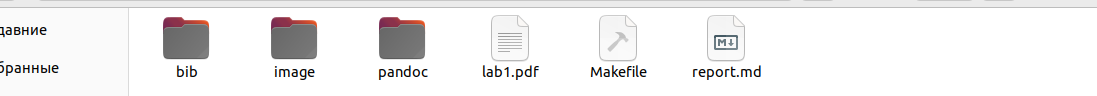
Работа с отчётом в текстовом процессоре

Перехожу из подкаталога lab02/report в подкаталог lab01/report с помощью утилиты cd



Перемещение между директориями

Копирую первую лабораторную и проверяю выполненную команду с помощью lab01/report



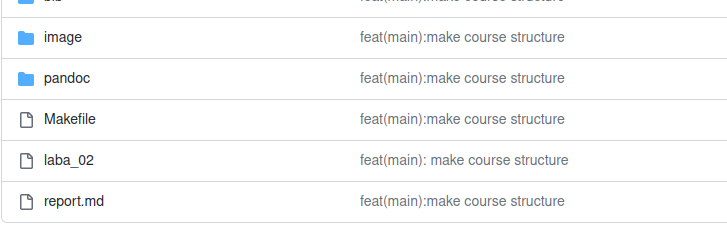
Копирование файла

Перехожу в директорию labs/lab02/report.Добавляю файл laba\_02.pdf

Прщверка местонахождение файла

Прщверка местонахождение файла

Проверяю на сайте правильность выполнения заданий



Страница каталоги в Github

# 5 Выводы

При выполнении данной лабораторной работы я изучила идеологию и применение средства контроля версий,а также приобрела практические навыки по работе системой git.

# Список литературы

1.{Архитектура ЭВМ} 2.{Git-gitattributes Документация}