

# Отчет по лабораторной работе №2

## Архитектура компьютеров

Ева Константиновна Пестова

Содержание: 1. Цель работы 2. Задание 3. Теоретическое введение  
4. Выполнение лабораторной работы 5. Выводы 6. Источники

### Цель работы

Целью данной работы является изучение идеологии и применение средств контроля версий, а также приобретение практических навыков по работе с системой git.

### Задание

- 1) настройка GitHub
- 2) базовая настройка Git
- 3) создание SSH-ключа
- 4) создание рабочего пространства
- 5) создание репозитория курса на основе шаблона
- 6) настройка каталога курса
- 7) выполнение заданий для самостоятельной работы

### Теоретическое введение

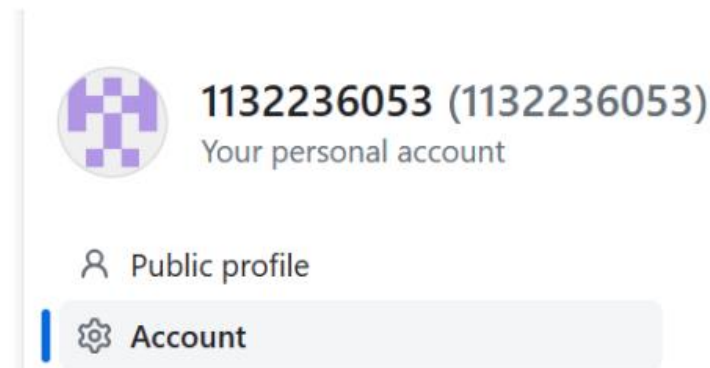
Системы контроля версий (Version Control System, VCS) применяются при работе нескольких человек над одним проектом. Обычно основное дерево проекта хранится в локальном или удалённом репозитории, к которому настроен доступ для участников проекта. При внесении изменений в содержание проекта система контроля версий позволяет их фиксировать, совмещать изменения, произведённые разными участниками проекта, производить откат к любой более ранней версии проекта, 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 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 изменения, сделанные разными участниками, вручную выбрать нужную версию, отменить изменения вовсе или заблокировать файлы для изменения. В зависимости от настроек блокировка не позволяет 7 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. 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 5 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. 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 распределённой системой контроля версий, резервную копию локального хранилища можно сделать простым копированием или архивацией. 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). 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.

## Выполнение лабораторной работы

### 1. Настройка GitHub

Создание личного профиля на сайте GitHub



## 2. Базовая настройка Git

Открываю виртуальную машину, затем открываю терминал и делаю предварительную конфигурацию git. Ввожу команду `git config --global user.name ""`, указывая свое имя и команду `git config --global user.email "work@mail"`, указывая в ней мою электронную почту. Настраиваю utf-8 в выводе сообщений git для корректного отображения символов. Задаю имя «master» для начальной ветки. Задаю параметр `autocrlf` со значением `input`, так как я работаю в системе Linux, чтобы конвертировать CRLF в LF только при коммитах. Задаю параметр `safecrlf` со значением `warn`, так Git будет проверять преобразование на обратимость.

```
ekpestova@dk6n35 ~ $ git config --global user.name "<Eva Pestova>"
ekpestova@dk6n35 ~ $ git config --global user.email "<1132236053@pfur.ru>"
ekpestova@dk6n35 ~ $ git config --global core.quotepath false
ekpestova@dk6n35 ~ $ git config --global init.defaultBranch master
ekpestova@dk6n35 ~ $ git config --global core.autocrlf input
ekpestova@dk6n35 ~ $ git config --global core.safecrlf warn
```

## 3. Создание SSH ключа

Для последующей идентификации пользователя на сервере репозиториев необходимо сгенерировать пару ключей (приватный и открытый). Для этого ввожу команду `ssh-keygen -C "Имя Фамилия, work@email"`, указывая имя владельца и электронную почту владельца. Ключ автоматически сохранится в каталоге `~/.ssh/`.

```
ekpestova@dk6n35 ~ $ ssh-keygen -C "Eva Pestova <1132236053@pfur.ru>"
Generating public/private rsa key pair.
Enter file in which to save the key (/afs/.dk.sci.pfu.edu.ru/home/e/k/ekpestova/.ssh/id_rsa):
Created directory '/afs/.dk.sci.pfu.edu.ru/home/e/k/ekpestova/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /afs/.dk.sci.pfu.edu.ru/home/e/k/ekpestova/.ssh/id_rsa
Your public key has been saved in /afs/.dk.sci.pfu.edu.ru/home/e/k/ekpestova/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:F9W63heOhtI3W/NdAh8BWPmyUTSda3ARRzKXtF0wou4 Eva Pestova <1132236053@pfur.ru>
The key's randomart image is:
+----[RSA 3072]-----+
|
| o=+OBX|
| .o+oX*|
| o ++.o|
| .+.+.|
| S o .o |
| o oo o |
| E...=oo|
| . o.+oo*|
| ooo.o |
+----[SHA256]-----+
```

Копирую открытый ключ из директории, в которой он был сохранен, с помощью утилиты xclip. `ekpestova@dk6n35 ~ $ cat ~/.ssh/id_rsa.pub | xclip -sel clip`

Открываю свой профиль на GitHub и выбираю страницу «SSH and GPG keys». Нажимаю кнопку «New SSH key». Вставляю скопированный ключ в поле «Key». В поле Title указываю имя для ключа. Нажимаю «Add SSH-key», чтобы завершить

Add new SSH Key

Title

evapestova

Key type

Authentication Key

Key

Uc4yCAH1y5sxhAX6qNYQ+CVsTqGQBnSyddQKJ2ncxj6/iHplAZvrLU/jTAm8  
VI8F+ixS5Qg8LXcTS8EsXaRJXt+M9THRMYp2uGKk1T7mxBk13TAu5cbPB1Q  
CkhmRahrKqL+2QD0mj4HkMDNbideoZYnpp3XQthYa2wA7oorWXg6PmjF  
f3Rfli4qmw5zTorzcvVzRKqkR5bPvXhCDMiwvAEw0lqUJ79Qlgk0a74Wchi8  
jQwk6hE0gC9b2K8Z/MBSrNsie+P+AHj34rwtRUBizu1Cno2teRIQabcL/D0fq  
GJVTCr+Wo9nwcBasZJu2g1bWGkcpvAj9q3Hb/DpRptnPFQrS2QhfZ6ajR  
QkekVfLEH0cg5HmOqdvqOo0UKH6TDdZBOI5kRC5+ALLuBZZ+MKP0mdV  
EkZvdDOdifys66Uwl9mNY0= Eva Pestova <1132236053@pfur.ru>

Add SSH key

добавление ключа.

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

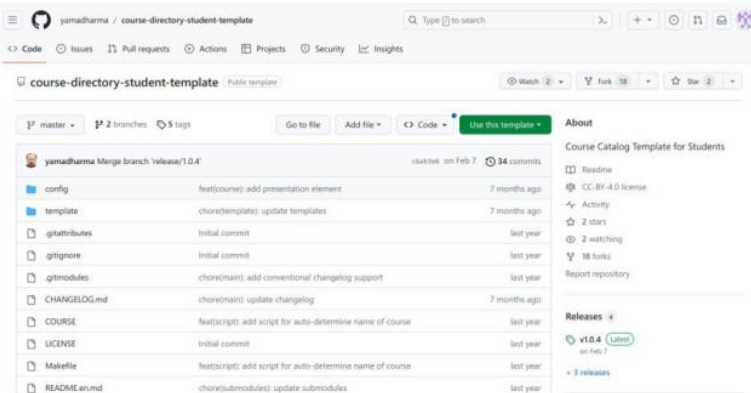
Создаю директорию, рабочее пространство, с помощью утилиты mkdir, благодаря ключу -p создаю все директории после домашней ~/work/study/2023-2024/“Архитектура компьютера” рекурсивно. Далее проверяю с помощью ls, действительно ли были созданы необходимые мне каталоги.

```
ekpestova@dk6n35 ~ $ mkdir -p work/study/2023-2024/"Архитектура компьютеров"
ekpestova@dk6n35 ~ $ ls
files      public    test      Видео      Загрузки   Музыка     'Рабочий стол'
newfiles   public_html work      Документы  Изображения  Общиедступные  Шаблоны
```

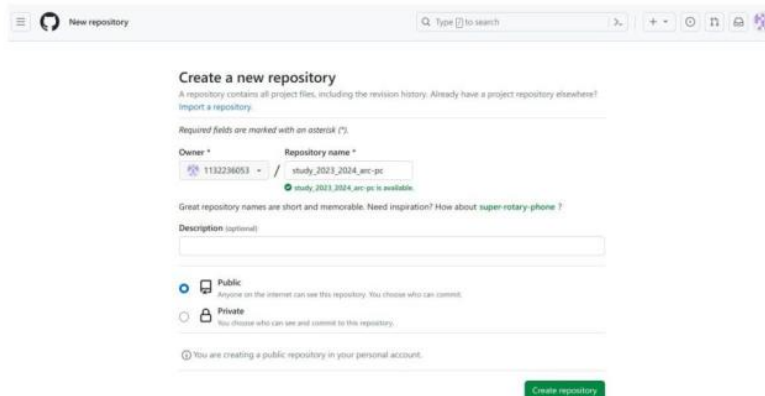
#### 5. Создание репозитория курса на основе шаблона

В браузере перехожу на страницу репозитория с шаблоном курса по адресу <https://github.com/yamadharma/course-directory-student-template>. Далее выбираю

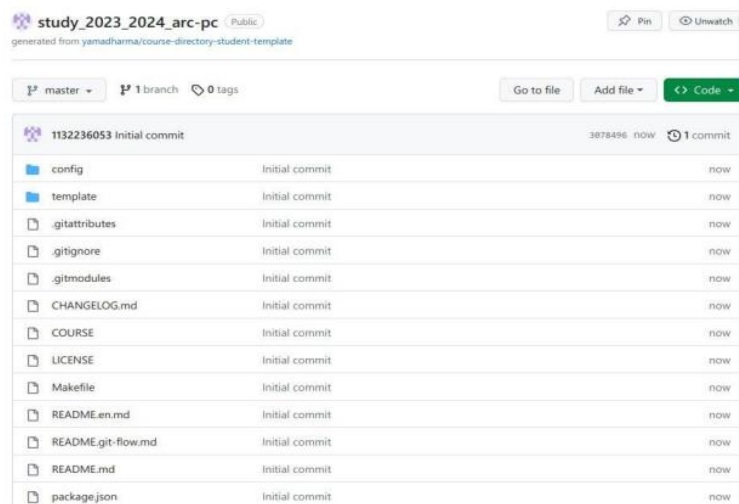
«Use this template», чтобы использовать этот шаблон для своего репозитория.



В открывшемся окне задаю имя репозитория (Repository name): study\_2023–2024\_arc-pc и создаю репозиторий, нажимаю на кнопку «Create repository from



template».



Репозиторий создан.

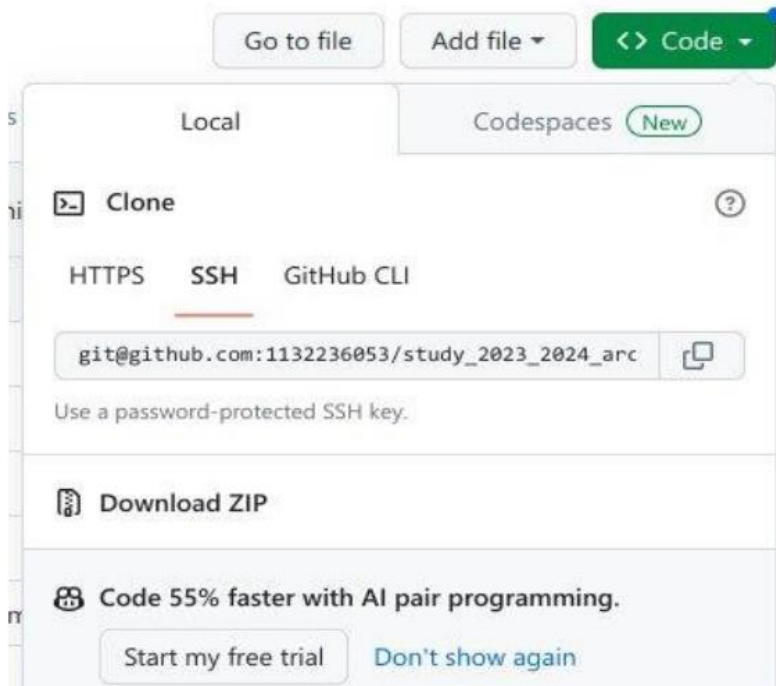
Через терминал перехожу в созданный каталог курса с помощью утилиты cd.

```
ekpestova@dk6n35 ~ $ cd ~/work/study/2023-2024/'Архитектура компьютеров'
ekpestova@dk6n35 ~/work/study/2023-2024/Архитектура компьютеров $
```

Клонирую созданный репозиторий с помощью команды `git clone --recursive git@github.com:/study_2023_2024_arc-pc.git arch-pc`

```
ekpestova@dk6n35 ~/work/study/2023-2024/Архитектура компьютеров $ git clone --recursive git@github.com:1132236053/study_2023_2024_arc-pc.git arch-pc
Клонирование в «arch-pc»...
```

Копирую ссылку для клонирования на странице созданного репозитория, сначала перейдя в окно «code», далее выбрав в окне вкладку «SSH».



## 6. Настройка каталога курса.

Перехожу в каталог `arch-pc` с помощью утилиты `cd`.

```
ekpestova@dk6n35 ~/work/study/2023-2024/Архитектура компьютеров $ cd ~/work/study/2023-2024/Архитектура компьютеров/arch-pc
ekpestova@dk6n35 ~/work/study/2023-2024/Архитектура компьютеров/arch-pc $
```

Удаляю лишние файлы с помощью утилиты `rm` и создаю необходимые каталоги.

```
ekpestova@dk6n35 ~/work/study/2023-2024/Архитектура компьютеров/arch-pc $ rm package.json
ekpestova@dk6n35 ~/work/study/2023-2024/Архитектура компьютеров/arch-pc $ echo arch-pc > COURSE
ekpestova@dk6n35 ~/work/study/2023-2024/Архитектура компьютеров/arch-pc $ make
```

Отправляю созданные каталоги с локального репозитория на сервер: добавляю все созданные каталоги с помощью `git add`, комментирую и сохраняю изменения на сервере как добавление курса с помощью `git commit`.

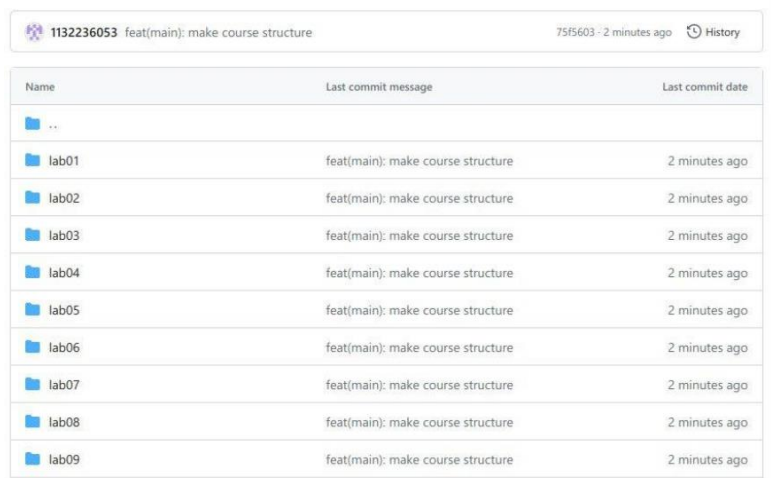
```
ekpestova@dk6n35 ~/work/study/2023-2024/Архитектура компьютеров/arch-pc $ git add .
ekpestova@dk6n35 ~/work/study/2023-2024/Архитектура компьютеров/arch-pc $ git commit -am 'feat(main): make course structure'
[master 75f5603] feat(main): make course structure
199 files changed, 54725 insertions(+), 14 deletions(-)
create mode 100644 labs/README.md
create mode 100644 labs/README.ru.md
create mode 100644 labs/lab01/presentation/Makefile
create mode 100644 labs/lab01/presentation/image/kulyabov.jpg
create mode 100644 labs/lab01/presentation/presentation.md
create mode 100644 labs/lab01/presentation/Makefile
```



Отправляю все на сервер с помощью push.

```
ekpestova@dk6n35 ~/work/study/2023-2024/Архитектура компьютеров/arch-pc $ git push
Перечисление объектов: 37, готово.
Подсчет объектов: 100% (37/37), готово.
При сжатии изменений используется до 4 потоков
Сжатие объектов: 100% (29/29), готово.
Запись объектов: 100% (35/35), 342.13 КиБ | 2.85 МиБ/с, готово.
Всего 35 (изменений 4), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
remote: Resolving deltas: 100% (4/4), completed with 1 local object.
To github.com:1132236053/study_2023_2024_arch-pc.git
 3078496..75f5603 master -> master
```

Проверяю правильность выполнения работы на самом сайте GitHub.



The screenshot shows a GitHub repository page for user 1132236053. The commit history table is as follows:

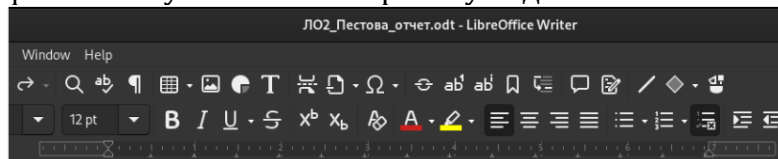
Name	Last commit message	Last commit date
..		
lab01	feat(main): make course structure	2 minutes ago
lab02	feat(main): make course structure	2 minutes ago
lab03	feat(main): make course structure	2 minutes ago
lab04	feat(main): make course structure	2 minutes ago
lab05	feat(main): make course structure	2 minutes ago
lab06	feat(main): make course structure	2 minutes ago
lab07	feat(main): make course structure	2 minutes ago
lab08	feat(main): make course structure	2 minutes ago
lab09	feat(main): make course structure	2 minutes ago

## 7. Выполнение заданий для самостоятельной работы.

Перехожу в директорию labs/lab03/report с помощью утилиты cd. Создаю в каталоге файл для отчета по второй лабораторной работе с помощью утилиты touch.

```
ekpestova@dk6n35 ~/work/study/2023-2024/Архитектура компьютеров/arch-pc $ cd labs/lab03/report
ekpestova@dk6n35 ~/work/study/2023-2024/Архитектура компьютеров/arch-pc/labs/lab03/report $ touch ЛО2_Пестова_отчет
```

Оформить отчет я смогу в текстовом процессоре LibreOffice Writer, найдя его в меню приложений. После открытия текстового процессора открываю в нем созданный файл и могу начать в нем работу над отчетом.



Перехожу из подкаталога lab03/report в подкаталог lab01/report с помощью

```
ekpestova@dk6n35 ~/work/study/2023-2024/Архитектура компьютеров/arch-pc/labs/lab03/report $ cd ..
ekpestova@dk6n35 ~/work/study/2023-2024/Архитектура компьютеров/arch-pc/labs/lab03 $ cd ..
ekpestova@dk6n35 ~/work/study/2023-2024/Архитектура компьютеров/arch-pc/labs $ cd lab01/
ekpestova@dk6n35 ~/work/study/2023-2024/Архитектура компьютеров/arch-pc/labs/lab01 $ cd ..
ekpestova@dk6n35 ~/work/study/2023-2024/Архитектура компьютеров/arch-pc/labs $ cd lab01/report/
```

утилиты cd.

Проверяю местонахождение файлов в подкаталоге домашней директории «Загрузки», для проверки использую команду ls.

```
ekpestova@dk6n35 ~/work/study/2023-2024/Архитектура компьютеров/arch-pc/labs/lab01/report $ ls ~/Загрузки
```

Копирую первую лабораторную с помощью утилиты `cp`.

```
ekpestova@dk6n35 ~/work/study/2023-2024/Архитектура компьютеров/arch-pc/labs/lab01/report $ cp ~/Задания/Л01_Пестова_отчет.pdf /home/evapestova/work/study/2023-2024/Архитектура компьютеров/arch-pc/labs/lab01/report
```

Перехожу из подкаталога `lab01/report` в подкаталог `lab02/report` с помощью

утилиты `cd`.

```
ekpestova@dk6n35 ~/work/study/2023-2024/Архитектура компьютеров/arch-pc/labs/lab01/report $ cd ..
ekpestova@dk6n35 ~/work/study/2023-2024/Архитектура компьютеров/arch-pc/labs/lab01 $ cd ..
ekpestova@dk6n35 ~/work/study/2023-2024/Архитектура компьютеров/arch-pc/labs $ cd lab02/report
ekpestova@dk6n35 ~/work/study/2023-2024/Архитектура компьютеров/arch-pc/labs/lab02/report $
```

Копирую вторую начатую лабораторную с помощью утилиты `cp`.

```
ekpestova@dk6n35 ~/work/study/2023-2024/Архитектура компьютеров/arch-pc/labs/lab02/report $ cp ~/Задания/Л02_Пестова_отчет.pdf /home/evapestova/work/study/2023-2024/Архитектура компьютеров/arch-pc/labs/lab02/report
```

Добавляю с помощью команды `git add` в коммит созданные файлы:

`Л02_Пестова_отчет.pdf`.

```
ekpestova@dk6n35 ~/work/study/2023-2024/Архитектура компьютеров/arch-pc/labs/lab02/report $ git add Л02_Пестова_отчет.pdf
```

Перехожу в директорию, в которой находится отчет по первой лабораторной работе

с помощью `cd`.

```
ekpestova@dk6n35 ~/work/study/2023-2024/Архитектура компьютеров/arch-pc/labs/lab02/report $ cd ..;
cd ..
ekpestova@dk6n35 ~/work/study/2023-2024/Архитектура компьютеров/arch-pc/labs $ cd lab01/report/
```

Добавляю файл `Л01_Пестова_отчет.pdf`.

```
ekpestova@dk6n35 ~/work/study/2023-2024/Архитектура компьютеров/arch-pc/labs/lab01/report $ git add Л01_Пестова_отчет.pdf
```

Сохраняю изменения на сервере командой `git commit -m "..."`, поясняя, что добавила файлы.

```
ekpestova@dk6n35 ~/work/study/2023-2024/Архитектура компьютеров/arch-pc/labs/lab01/report $ git commit -m "Add existing file"
```

Отправляю в центральный репозиторий сохраненные изменения командой `git push -f origin master`.

```
ekpestova@dk6n35 ~/work/study/2023-2024/Архитектура компьютеров/arch-pc/labs/lab01/report $ git push -f origin master
```

Проверяю на сайте GitHub правильность выполнения заданий и вижу, что были добавлены файлы с отчетами по лабораторным работам.

## Выводы

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

## Список литературы

1. Архитектура ЭВМ
2. Git - gitattributes Документация