

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ № 3

дисциплина: Архитектура компьютера

Студент: Шошина Евгения Александровна

Группа: НКАбд-03-22

МОСКВА

2022 г.

Содержание

1. Цель работы.....	3
2. Теоретическое введение.....	4
3. Выполнение лабораторной работы.....	8
4. Выводы.....	13
5. Список литературы.....	14

Цель работы

Целью работы является изучить идеологию и применение средств контроля версий.
Приобрести практические навыки по работе с системой git.

Теоретическое введение

3.2.1. 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 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.

3.2.2. 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 распределённой системой контроля версий, резервную копию локального хранилища можно сделать простым копированием или архивацией.

3.2.3. Основные команды git

Наиболее часто используемые команды `git` представлены в таблице 3.1.

Команда	Описание
<code>git init</code>	создание основного дерева репозитория
<code>git pull</code>	получение обновлений (изменений) текущего дерева из центрального репозитория
<code>git push</code>	отправка всех произведённых изменений локального дерева в центральный репозиторий
<code>git status</code>	просмотр списка изменённых файлов в текущей директории
<code>git diff</code>	просмотр текущих изменения
<code>git add .</code>	добавить все изменённые и/или созданные файлы и/или каталоги
<code>git add имена_файлов</code>	добавить конкретные изменённые и/или созданные файлы и/или каталоги
<code>git rm имена_файлов</code>	удалить файл и/или каталог из индекса репозитория (при этом файл и/или каталог остаётся в локальной директории)
<code>git commit -am 'Описание коммита'</code>	сохранить все добавленные изменения и все изменённые файлы
<code>git checkout -b имя_ветки</code>	создание новой ветки, базирующейся на текущей
<code>git checkout имя_ветки</code>	переключение на некоторую ветку (при переключении на ветку, которой ещё нет в локальном репозитории, она будет создана и связана с удалённой)
<code>git push origin имя_ветки</code>	отправка изменений конкретной ветки в центральный репозиторий
<code>git merge --no-ff имя_ветки</code>	слияние ветки с текущим деревом
<code>git branch -d имя_ветки</code>	удаление локальной уже слитой с основным деревом ветки
<code>git branch -D имя_ветки</code>	принудительное удаление локальной ветки
<code>git push origin :имя_ветки</code>	удаление ветки с центрального репозитория

3.2.4. 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 центральной 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 push

Лабораторная работа

Настройка github

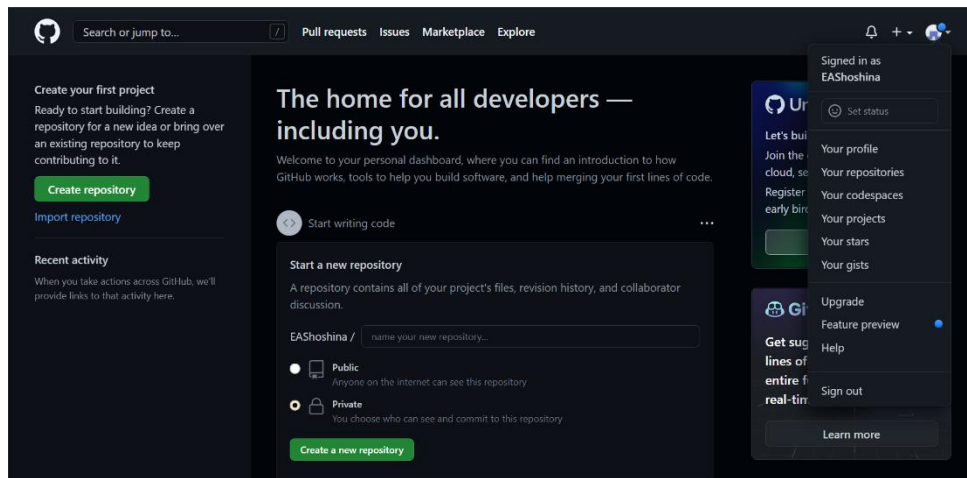


Рис 3.4.1.1 Создание учетной записи на github

Базовая настройка git

```
[eashoshina@fedora ~]$ git config --global user.name "EAShoshina"
[eashoshina@fedora ~]$ git config --global user.email "<lady.shoshina2017@yandex.ru>"
[eashoshina@fedora ~]$ git config --global core.quotepath false
[eashoshina@fedora ~]$ git config --global init.defaultBranch master
[eashoshina@fedora ~]$ git config --global core.autocrlf input
[eashoshina@fedora ~]$ git config --global core.safecrlf warn
```

Рис 3.4.2.1 Проводим предварительную конфигурацию git

Создание SSH ключа

```
[eashoshina@fedora ~]$ ssh-keygen -C"EAShoshina <lady.shoshina2017@yandex.ru>"
Generating public/private rsa key pair.
Enter file in which to save the key (/home/eashoshina/.ssh/id_rsa):
/home/eashoshina/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/eashoshina/.ssh/id_rsa
Your public key has been saved in /home/eashoshina/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:rMHv7RFarmunEvFZrcHX/xuXNIC9TUqHjTfU5BNZX7Y EAShoshina <lady.shoshina2017@yandex.ru>
The key's randomart image is:
+----[RSA 3072]-----+
|      .      +O|
|      o   o  =+*|
|     .o. o  * Eo|
|    .. o+.o o 0 o|
|   o+oS+.  + + |
|  . o=.  .  o o|
|  .+ o .  +. |
| .. + . .  + |
| .. ..o .. |
+----[SHA256]-----+
[eashoshina@fedora ~]$ cat ~/.ssh/id_rsa.pub | xclip -sel clip
[eashoshina@fedora ~]$
```

Рис 3.4.3.1 Для последующей идентификации пользователя на сервере репозитория сгенерировала пару ключей (приватный и открытый).

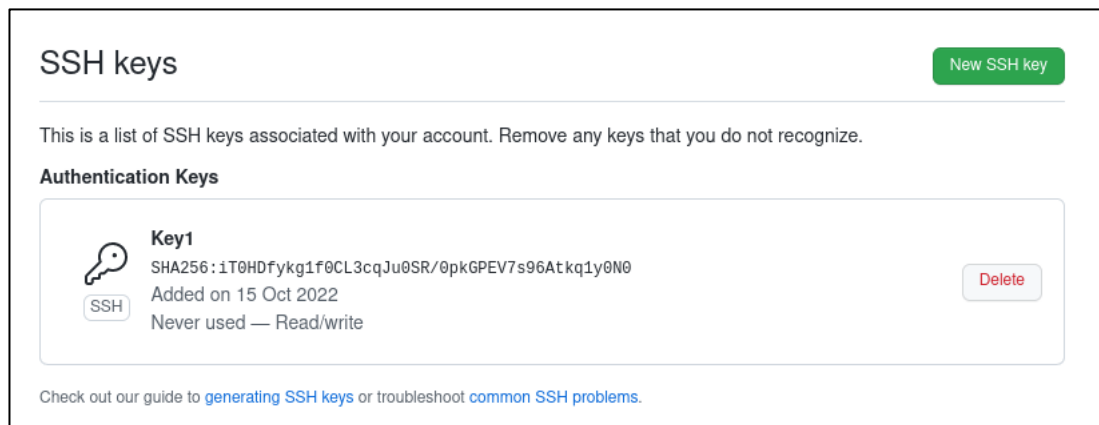


Рис 3.4.3.2 Загрузка сгенерённого ключа

```
[eashoshina@fedora ~]$ cat ~/.ssh.id_rsa.pub | xclip -sel clip
```

Рис 3.4.3.3 Скопировав из локальной консоли ключ в буфер обмена, вставила ключ в появившееся на сайте поле и указала для ключа имя (Title).

Создание рабочего пространства и репозитория курса на основе шаблона

```
[eashoshina@fedora ~]$ mkdir -p ~/work/study/2022-2023/"Архитектура компьютера"
[eashoshina@fedora ~]$
```

Рис 3.4.4.1 Создание каталога для предмета «Архитектура компьютера»

Создание репозитория курса на основе шаблона

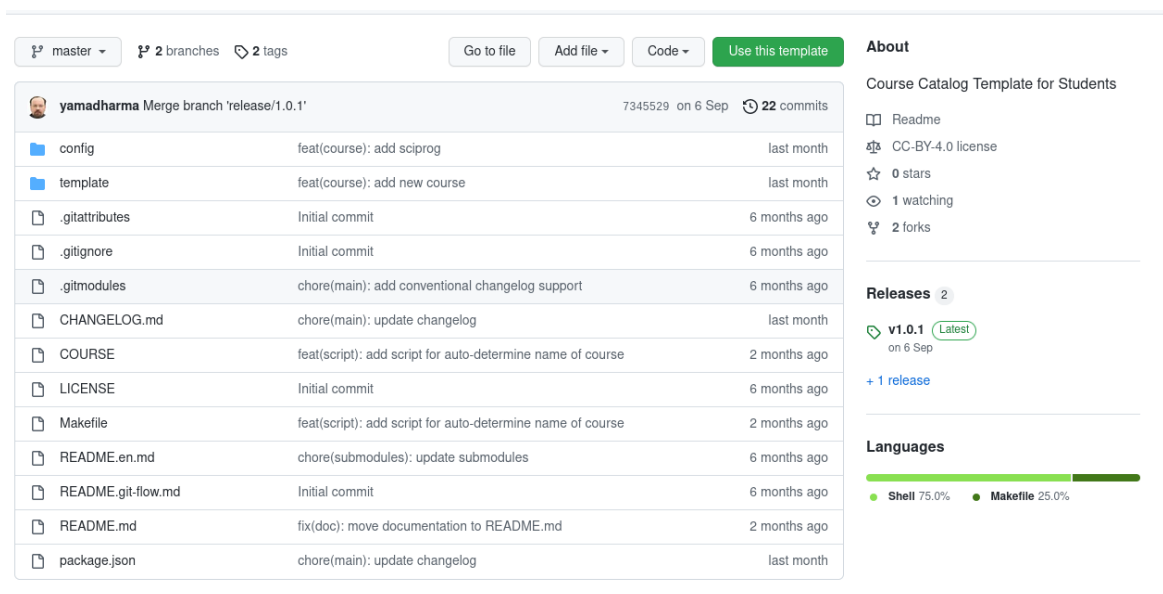



Рис 3.4.5.1 Выбор шаблона

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere?
[Import a repository.](#)

Owner *

 EAShoshina ▾

Repository name *

study_2022-2023_arch-pc ✓

Great repository names are short and memorable. Need inspiration? How about **ubiquitous-engine**?

Description (optional)

Рис 3.4.5.2 Создание репозитория

```
[eashoshina@fedora Архитектура компьютера]$ git clone --recursive git@github.com:EAShoshina/study_2022-2023_arch-pc.git
Клонирование в «study_2022-2023_arch-pc»...
The authenticity of host 'github.com (140.82.121.3)' can't be established.
ED25519 key fingerprint is SHA256:+DiY3wvV6TuJJhbpZisF/zLDA0zPMSvHdkr4UvCOqU.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added 'github.com' (ED25519) to the list of known hosts.
remote: Enumerating objects: 26, done.
remote: Counting objects: 100% (26/26), done.
remote: Compressing objects: 100% (25/25), done.
remote: Total 26 (delta 0), reused 17 (delta 0), pack-reused 0
Получение объектов: 100% (26/26), 16.39 КиБ | 839.00 КиБ/с, готово.
Подмодуль «template/presentation» (https://github.com/yamadharma/academic-presentation-markdown-template.git) зарегистрирован по пути «template/presentation»
Подмодуль «template/report» (https://github.com/yamadharma/academic-laboratory-report-template.git) зарегистрирован по пути «template/report»
Клонирование в «/home/eashoshina/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arch-pc/template/presentation»...
remote: Enumerating objects: 71, done.
remote: Counting objects: 100% (71/71), done.
remote: Compressing objects: 100% (49/49), done.
remote: Total 71 (delta 23), reused 68 (delta 20), pack-reused 0
Получение объектов: 100% (71/71), 88.89 КиБ | 716.00 КиБ/с, готово.
Определение изменений: 100% (23/23), готово.
Клонирование в «/home/eashoshina/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arch-pc/template/report»...
remote: Enumerating objects: 78, done.
remote: Counting objects: 100% (78/78), done.
remote: Compressing objects: 100% (52/52), done.
remote: Total 78 (delta 31), reused 69 (delta 22), pack-reused 0
Получение объектов: 100% (78/78), 292.27 КиБ | 1.25 МиБ/с, готово.
Определение изменений: 100% (31/31), готово.
Submodule path 'template/presentation': checked out '2703b47423792d472694aaf7555a5626dce51a25'
Submodule path 'template/report': checked out 'df7b2ef80f8def3b9a496f8695277469a1a7842a'
[eashoshina@fedora Архитектура компьютера]$
```

Рис 3.4.5.3 Клонирование репозитория

Настройка каталога курса

```
[eashoshina@fedora arch-pc]$ rm package.json
[eashoshina@fedora arch-pc]$ echo arch-pc > COURSE
[eashoshina@fedora arch-pc]$ make
[eashoshina@fedora arch-pc]$ git add .
[eashoshina@fedora arch-pc]$ git commit -am 'feat(main): make course structure'
[master 33e1b09] feat(main): make course structure
91 files changed, 8229 insertions(+), 14 deletions(-)
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/report/Makefile
create mode 100644 labs/lab01/report/bib/cite.bib
create mode 100644 labs/lab01/report/image/placeimg_800_600_tech.jpg
create mode 100644 labs/lab01/report/pandoc/csl/gost-r-7-0-5-2008-numeric.csl
create mode 100644 labs/lab01/report/report.md
create mode 100644 labs/lab02/presentation/Makefile
create mode 100644 labs/lab02/presentation/image/kulyabov.jpg
create mode 100644 labs/lab02/presentation/presentation.md
create mode 100644 labs/lab02/report/Makefile
create mode 100644 labs/lab02/report/bib/cite.bib
create mode 100644 labs/lab02/report/image/placeimg_800_600_tech.jpg
create mode 100644 labs/lab02/report/pandoc/csl/gost-r-7-0-5-2008-numeric.csl
create mode 100644 labs/lab02/report/report.md
create mode 100644 labs/lab03/presentation/Makefile
create mode 100644 labs/lab03/presentation/image/kulyabov.jpg
create mode 100644 labs/lab03/presentation/presentation.md
create mode 100644 labs/lab03/report/Makefile
create mode 100644 labs/lab03/report/bib/cite.bib
create mode 100644 labs/lab03/report/image/placeimg_800_600_tech.jpg
create mode 100644 labs/lab03/report/pandoc/csl/gost-r-7-0-5-2008-numeric.csl
create mode 100644 labs/lab03/report/report.md
```

<...>

```
[eashoshina@fedora arch-pc]$ git push
Перечисление объектов: 22, готово.
Подсчет объектов: 100% (22/22), готово.
Сжатие объектов: 100% (16/16), готово.
Запись объектов: 100% (20/20), 310.95 КиБ | 1.78 МиБ/с, готово.
Всего 20 (изменений 1), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:EAShoshina/study_2022-2023_arh-pc.git
 f47d0b6..33e1b09 master -> master
[eashoshina@fedora arch-pc]$
```

Рис 3.4.6.1 Переход в каталог курса, удаление лишних и создание необходимых файлов, отправка файлов на сервер.

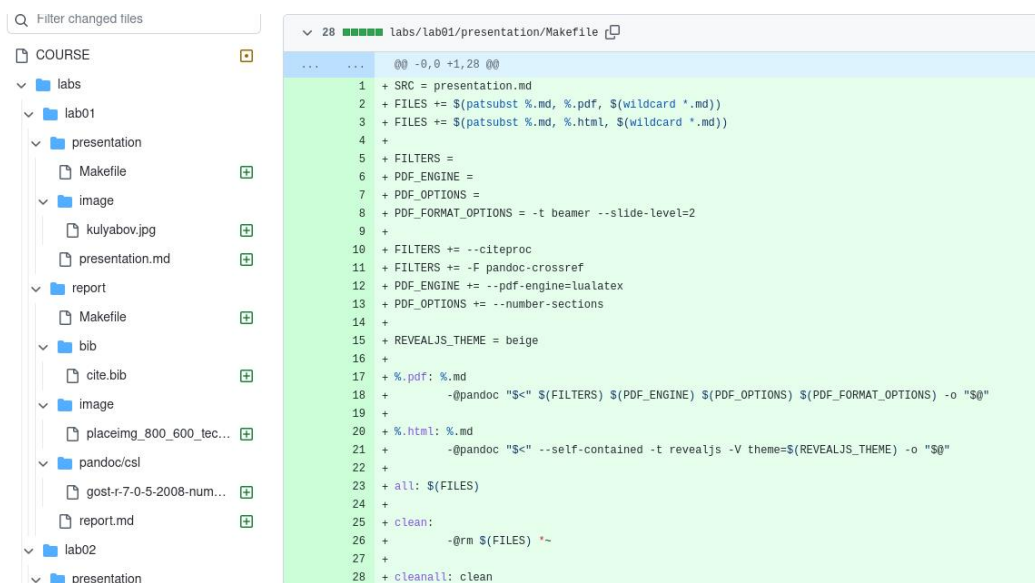
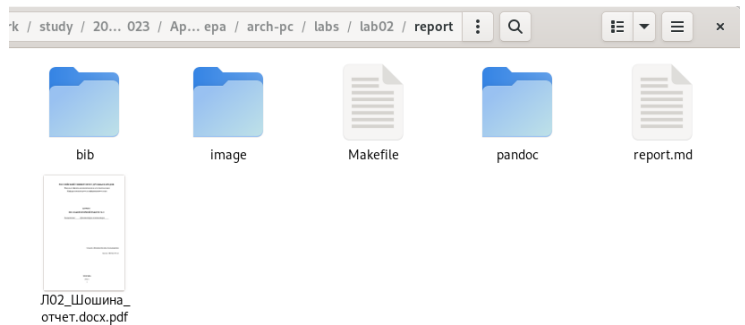
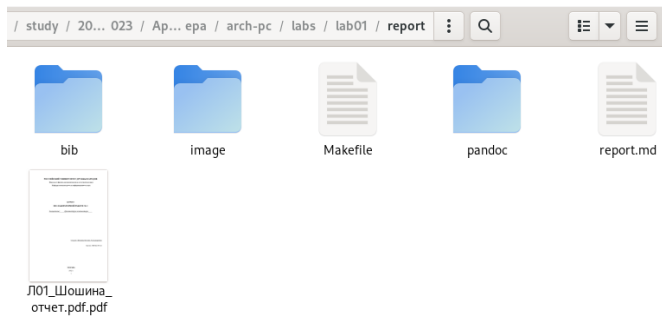


Рис 3.4.6.2 Проверка правильности создания иерархии рабочего пространства в локальном репозитории и на странице github.

Задание для самостоятельной работы

1. Создайте отчет по выполнению лабораторной работы в соответствующем каталоге рабочего пространства (labs>lab03>report).
2. Скопируйте отчеты по выполнению предыдущих лабораторных работ в соответствующие каталоги созданного рабочего пространства.



Полноценный отчет №3 загрузила по аналогии по завершении выполнения лабораторной работы.

3. Загрузила файлы на github.

```
[eashoshina@fedora arch-pc]$ git add .
[eashoshina@fedora arch-pc]$ git commit -am 'report'
[master 48671f2] report
 2 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 labs/lab01/report/Л01_Шошина_отчет.pdf.pdf
 create mode 100644 labs/lab02/report/Л02_Шошина_отчет.docx.pdf
[eashoshina@fedora arch-pc]$ git push
Перечисление объектов: 15, готово.
Подсчет объектов: 100% (13/13), готово.
Сжатие объектов: 100% (9/9), готово.
Запись объектов: 100% (9/9), 2.31 МиБ | 802.00 КиБ/с, готово.
Всего 9 (изменений 3), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
remote: Resolving deltas: 100% (3/3), completed with 2 local objects.
To github.com:EAShoshina/study_2022-2023_arh-pc.git
```

Вывод

Изучила идеологию и применение средств контроля версий, а также приобрела практические навыки по работе с системой git.

Список литературы

https://www.researchgate.net/publication/331545809_Arhitektura_vycislitelnyh_sistem_laboratornye_raboty (Архитектура вычислительных систем: лабораторные работы, Январь 2019 г., Издатель: Российский университет дружбы народов, ISBN: 978-5-209-08880-6, Проект: Системная и сетевая инженерия)