

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ №2

Дисциплина : Архитектура компьютера

Студент : Зоригоо Номун
Группа : НКАбд-04-23

МОСКВА
2023 г.

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	9
4.1	Настройка GitHub	9
4.2	Базовая настройка Git.....	10
4.3	Создание SSH-ключа	11
4.4	Создание рабочего пространства и репозитория курса на основе шаблона	14
4.5	Создание репозитория курса на основе шаблона	15
4.6	Настройка каталога курса.....	17
4.7	Выполнение заданий для самостоятельной работы	20
5	Выводы	27
6	Список литературы	28

1 Цель работы

Целью данной работы является изучить идеологию и применение средств контроля версий, а также приобрести практические навыки по работе с системой `git`.

2 Задание

1. Настройка GitHub.
2. Базовая настройка Git.
3. Создание SSH-ключа.
4. Создание рабочего пространства и репозитория 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 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

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 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.

4 Выполнение лабораторной работы

4.1 Настройка GitHub

Создаю учетную запись на сайте GitHub (рис 1.1)

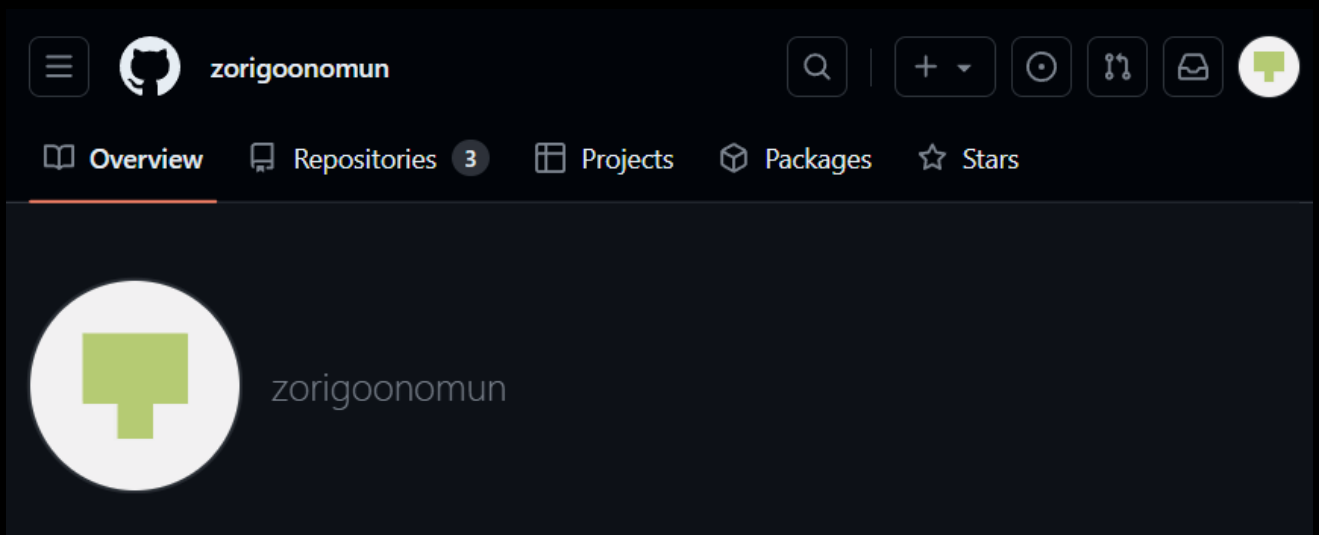


Рис 1.1: GitHub account

4.2 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 электронную почту владельца. (рис 1.2).

```
nzorigoo@Linux:~$ git config --global user.name "<nzorigoo>"
nzorigoo@Linux:~$ git config --global user.email "<1032225436@pfur.ru>"
```

Рис1.2: Предварительная конфигурация git

Настраиваю utf-8 в выводе сообщений git для корректного отображения символов (рис. 1.3).

```
nzorigoo@Linux:~$ git config --global core.quotePath false
```

Рис. 1.3: Настройка кодировки

Задаю имя «master» для начальной ветки (рис 1.4)

```
nzorigoo@Linux:~$ git config --global core.quotePath false
nzorigoo@Linux:~$ git config --global init.defaultBranch master
```

Рис. 1.4: Создание имени для начальной ветки

Задаю параметр `autocrlf` с значением `input`, так как я работаю в системе Linux, чтобы конвертировать CRLF в LF только при коммитах (рис. 1.5). CR и LF – это символы, которые можно использовать для обозначения разрыва строки в текстовых файлах

```
nzorigoo@Linux:~$ git config --global core.autocrlf input
```

Рис 1.5

Задаю параметр `safecrlf` со значением `warn`, так Git будет проверять преобразование на обратимость (рис. 1.6). При значении `warn` Git только выведет предупреждение, но будет принимать необратимые конвертации.

```
nzorigoo@Linux:~$ git config --global core.safecrlf warn
```

Рис 1.6

4.3 Создание SSH-ключа

Для последующей идентификации пользователя на сервере репозитория 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 (рис. 1.7). Ключ автоматически сохранится в каталоге `~/.ssh/`

```
nzorigoo@Linux: ~$ ssh-keygen -C "Nzorigoo <1032225436@pfur.ru>"
Generating public/private rsa key pair.
Enter file in which to save the key (/home/nzorigoo/.ssh/id_rsa): documents
documents already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in documents
Your public key has been saved in documents.pub
The key fingerprint is:
SHA256:2A1IOA0uHRE3A/Yl8Cby+aG2arQy1H5owwYkWVYLDzA Nzorigoo <1032225436@pfur.ru>
The key's randomart image is:
+----[RSA 3072]-----+
|  Eo%XB .          |
|  o=oX.*           |
|  +o +.* .         |
|o. + + o o         |
|o . o o S .        |
| o.. o .           |
|..+..+ .           |
|o oO o             |
| ++.+              |
+-----[SHA256]-----+
```

Рис 1.7

Xclip – утилита, позволяющая скопировать любой текст через терминал. Оказывается, в дистрибутиве Linux Kali ее сначала надо установить. Устанавливаю xclip с помощью команды apt-get install с ключом -y от имени суперпользователя, введя в начале команды sudo (рис. 1.8).

```
nzorigoo@Linux:~$ sudo apt-get install -y xclip
[sudo] password for nzorigoo:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  xclip
0 upgraded, 1 newly installed, 0 to remove and 3 not upgraded.
Need to get 18.3 kB of archives.
After this operation, 60.4 kB of additional disk space will be used.
Get:1 http://ru.archive.ubuntu.com/ubuntu jammy/universe amd64 xclip amd64 0.13-2 [18.3 kB]
Fetched 18.3 kB in 1s (25.8 kB/s)
Selecting previously unselected package xclip.
(Reading database ... 199716 files and directories currently installed.)
Preparing to unpack .../xclip_0.13-2_amd64.deb ...
Unpacking xclip (0.13-2) ...
Setting up xclip (0.13-2) ...
Processing triggers for man-db (2.10.2-1) ...
nzorigoo@Linux:~$
```

Рис. 1.8: Установка утилиты xclip

Копирую открытый ключ из директории, в которой он был сохранен, с помощью утилиты xclip (рис. 1.9).

```
nzorigoo@Linux:~$ cat ~/.ssh/id_rsa.pub | xclip -sel clip
```

Рис 1.9 Копирование содержимого файла

Открываю браузер, захожу на сайт GitHub. Открываю свой профиль и выбираю страницу «SSH and GPG keys». Нажимаю кнопку «New SSH key» (рис. 1.10).

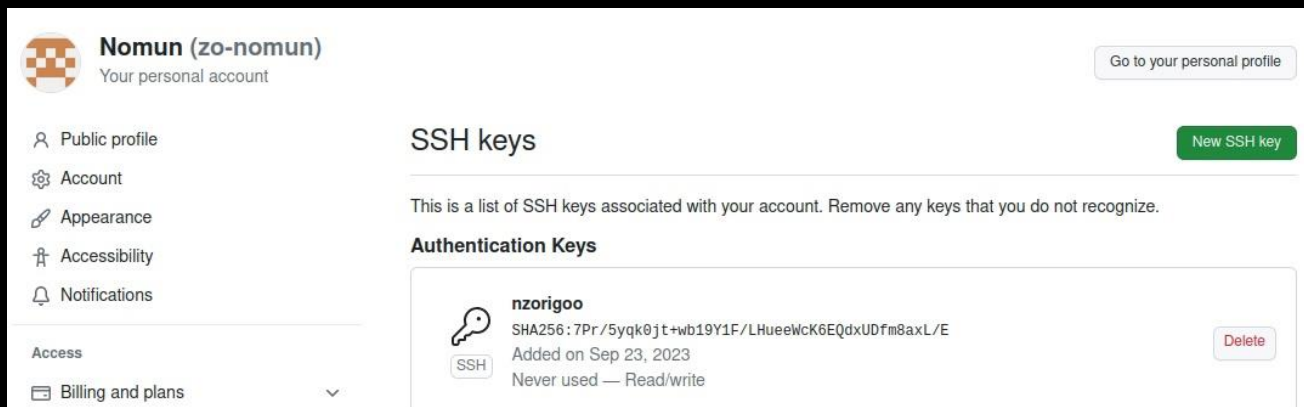


Рис. 1.10: Окно SSH

Вставляю скопированный ключ в поле «Key». В поле Title указываю имя для ключа. (рис 1.11)

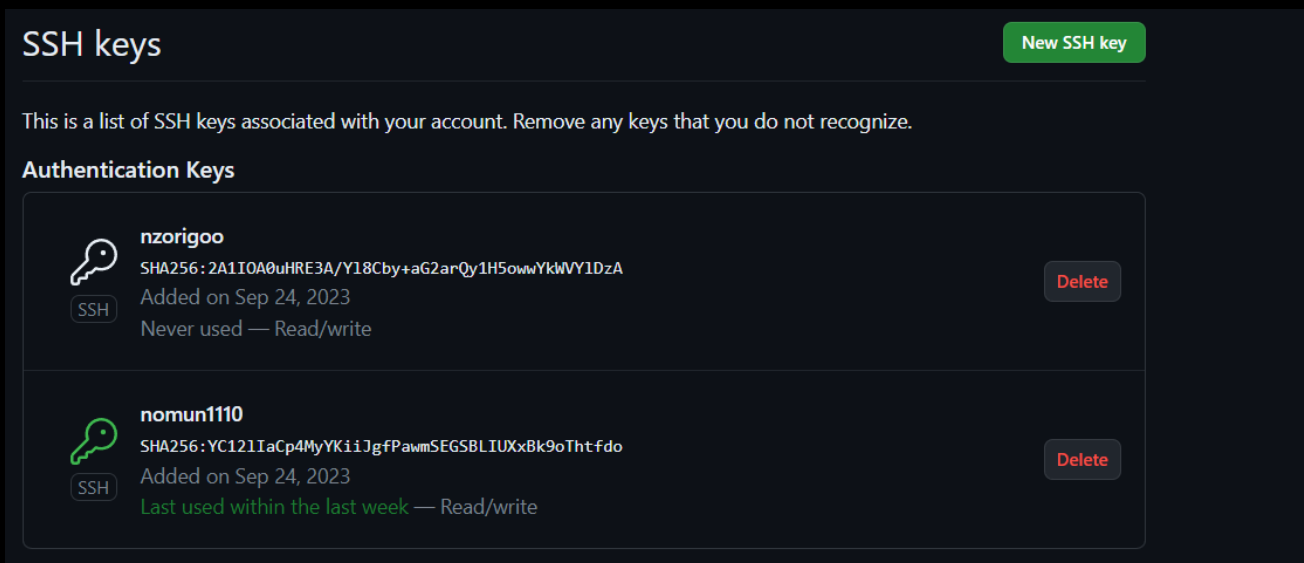


Рис. 1.11: Добавление ключа

4.4 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/2022-2023/“Архитектура компьютера”` рекурсивно. Далее проверяю с помощью `ls`, действительно ли были созданы каталоги (рис. 1.12).

```
nzorigoo@Linux:~$ mkdir -p work/study/2023-2024/"Computer Architecture"
nzorigoo@Linux:~$ ls
Desktop      Downloads  parentdir1  Pictures     Videos
documents    git        parentdir2  Public       work
Documents    Music      parentdir3  snap         zorigoo.txt
documents.pub parentdir   perentdir   Templates
```

Рис. 1.12: Создание рабочего пространства

Создание репозитория курса на основе шаблона

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

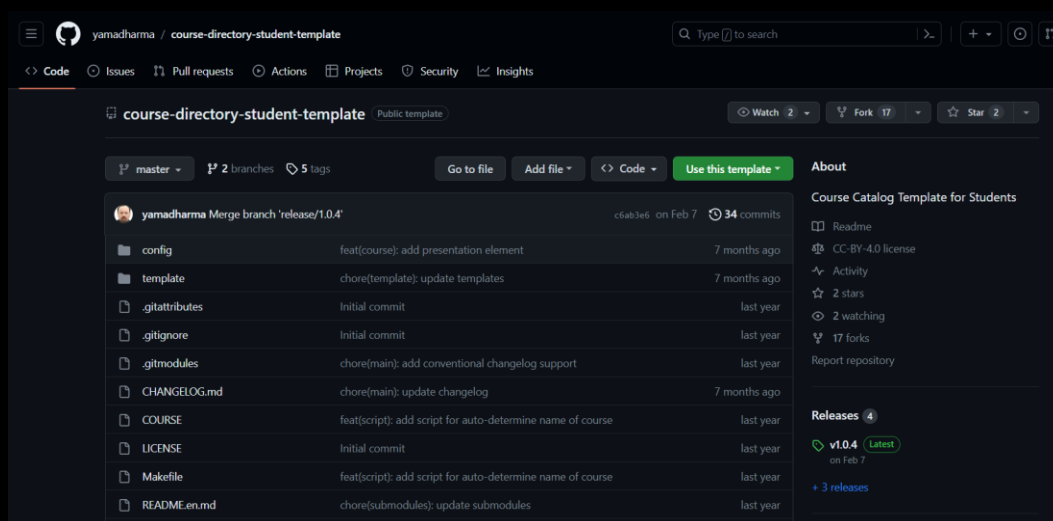



Рис 1.13

В открывшемся окне задаю имя репозитория (Repository name): study_2022–2023_arh-pc и создаю репозиторий, нажимая на кнопку «Create repository» (рис. 1.14).

Owner *

 zorigoonomun


/

study_2023-2024_arh-pc

study_2023-2024_arh-pc is available.

Great repository names are short and memorable. Need inspiration? How about **symmetrical-octo-meme** ?

Description (optional)

☒  **Public**
Anyone on the internet can see this repository. You choose who can commit.


☐  **Private**
You choose who can see and commit to this repository.

Рис 1.14

Репозиторий создан (рис.1.15)(забыла сделать скриншот)

study_2023-2024_arh-pc

Public template

generated from yamadharma/course-directory-student-template

master

1 branch

0 tags

Go to file

Add file

Code

Use this template

zorigoonomun feat(main): make course structure

3acfa5 3 days ago 3 commits

config	Initial commit	3 days ago
labs	feat(main): make course structure	3 days ago
presentation	feat(main): make course structure	3 days ago
template	Initial commit	3 days ago
.gitattributes	Initial commit	3 days ago
.gitignore	Initial commit	3 days ago
.gitmodules	Initial commit	3 days ago
CHANGELOG.md	Initial commit	3 days ago
COURSE	feat(main): make course structure	3 days ago
LICENSE	Initial commit	3 days ago
Makefile	Initial commit	3 days ago
README	Initial commit	3 days ago

Рис 1.15

Через терминал перехожу в созданный каталог курса с помощью утилиты `cd` (рис. 1.16).

```
nzorigoo@Linux:~$ cd ~/work/study/2023-2024/'Computer Architecture'  
nzorigoo@Linux:~/work/study/2023-2024/Computer Architecture$
```

Рис 1.16

Клонирую созданный репозиторий с помощью команды `git clone --recursive git@github.com:/study_2023-2024_arh-pc.git arch-pc` (рис. 1.17).

```
nzorigoo@Linux:~$ git clone --recursive git@github.com:zorigoonomun/study_2023-2  
024_arh-pc.git  
Cloning into 'study_2023-2024_arh-pc'...
```

Рис 1.17

Копирую ссылку для клонирования на странице созданного репозитория, сначала перейдя в окно «code», далее выбрав в окне вкладку «SSH» (рис. 1.18).

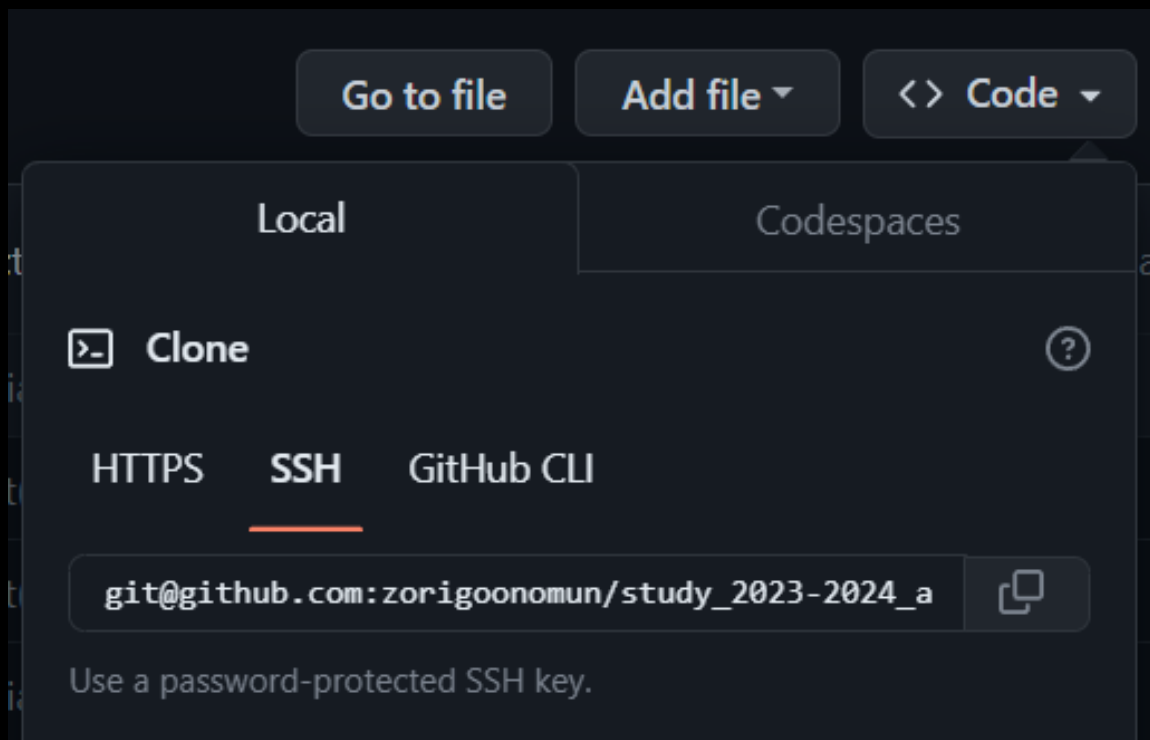


Рис 1.18

4.5 Настройка каталога курса

Перехожу в каталог arch-pc с помощью утилиты cd (рис. 1.18).

```
nzorigoo@Linux:~/work/study/2023-2024/Архитектура компьютера$ cd ~/work/study/2023-2024/Архитектура\ компьютера/arch-pc
nzorigoo@Linux:~/work/study/2023-2024/Архитектура компьютера/arch-pc$
```

Рис 1.18

Удаляю лишние файлы с помощью утилиты rm (рис. 1.19).

```
nzorigoo@Linux:~/work/study/2023-2024/Архитектура компьютера/arch-pc$ rm package.json
nzorigoo@Linux:~/work/study/2023-2024/Архитектура компьютера/arch-pc$
```

Рис 1.19

Создаю необходимые каталоги (рис. 1.20).

```
nzorigoo@Linux:~/work/study/2023-2024/Архитектура компьютера$ echo arch-pc > COURSE
RSE
nzorigoo@Linux:~/work/study/2023-2024/Архитектура компьютера$ make
```

Рис 1.20

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 (рис. 1.21).

```
nzorigoo@Linux:~/work/study/2023-2024/Архитектура компьютера/arch-pc$ git commit -am 'feat(main): make course structure'
[master 3acfaf5] feat(main): make course structure
197 files changed, 54724 insertions(+)
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/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 100755 labs/lab01/report/pandoc/filters/pandoc_eqnos.py
create mode 100755 labs/lab01/report/pandoc/filters/pandoc_fignos.py
create mode 100755 labs/lab01/report/pandoc/filters/pandoc_secnos.py
create mode 100755 labs/lab01/report/pandoc/filters/pandoc_tablenos.py
create mode 100644 labs/lab01/report/pandoc/filters/pandocxnos/__init__.py
create mode 100644 labs/lab01/report/pandoc/filters/pandocxnos/core.py
create mode 100644 labs/lab01/report/pandoc/filters/pandocxnos/main.py
create mode 100644 labs/lab01/report/pandoc/filters/pandocxnos/pandocattributes.py
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 100755 labs/lab02/report/pandoc/filters/pandoc_eqnos.py
create mode 100755 labs/lab02/report/pandoc/filters/pandoc_fignos.py
create mode 100755 labs/lab02/report/pandoc/filters/pandoc_secnos.py
create mode 100755 labs/lab02/report/pandoc/filters/pandoc_tablenos.py
create mode 100644 labs/lab02/report/pandoc/filters/pandocxnos/__init__.py
create mode 100644 labs/lab02/report/pandoc/filters/pandocxnos/core.py
create mode 100644 labs/lab02/report/pandoc/filters/pandocxnos/main.py
create mode 100644 labs/lab02/report/pandoc/filters/pandocxnos/pandocattributes.py
create mode 100644 labs/lab02/report/report.md
```

Рис 1.21


Отправляю все на сервер с помощью push (рис. 1.22).

```
nothing to commit, working tree clean
nзоригуоо@Linux:~/work/study/2023-2024/Архитектура компьютера/arch-pc$ git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 2 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 285 bytes | 285.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:zorigoonomun/study_2023-2024_arh-pc.git
   c2561d2..9dc7bfd  master -> master
```

Рис 1.22

Проверяю правильность выполнения работы сайте GitHub(рис. 1.23).

study_2023-2024_arh-pc / labs / Add file ...

zorigoonomun

feat(main): make course structure

3acfa5 · 3 days ago History

Name	Last commit message	Last commit date
..		
lab01	feat(main): make course structure	3 days ago
lab02	feat(main): make course structure	3 days ago
lab03	feat(main): make course structure	3 days ago
lab04	feat(main): make course structure	3 days ago
lab05	feat(main): make course structure	3 days ago
lab06	feat(main): make course structure	3 days ago
lab07	feat(main): make course structure	3 days ago
lab08	feat(main): make course structure	3 days ago
lab09	feat(main): make course structure	3 days ago
lab10	feat(main): make course structure	3 days ago
lab11	feat(main): make course structure	3 days ago

Рис 1.23

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