

Отчёт по лабораторной работе №2

Дисциплина: Архитектура Компьютера

Азарцова Вероника Валерьевна

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	10
4.1	Базовая конфигурация для работы с Git	10
4.2	Ключ SSH	10
4.3	Ключ PGP	12
4.4	Настройка подписи Git	14
4.5	Регистрация на Github	15
4.6	Создание локального каталога для работы по предмету	15
4.7	Контрольные вопросы	16
5	Выводы	21
	Список литературы	22

Список иллюстраций

4.1	Установленный git и gh	10
4.2	Имя и эл. почта владельца	10
4.3	Ключ SSH по алгоритму rsa	11
4.4	Ключ SSH по алгоритму ed25519	11
4.5	Список ключей	13
4.6	Настройка автоматических подписей	14
4.7	Создание рабочего пространства	15
4.8	Создание рабочего пространства	16
4.9	Создание рабочего пространства	16

Список таблиц

1 Цель работы

Цель данной лабораторной работы - изучить идеологию и применение средств контроля версий и освоить умения по работе с git.

2 Задание

1. Создать базовую конфигурацию для работы с git.
2. Создать ключ SSH.
3. Создать ключ PGP.
4. Настроить подписи git.
5. Зарегистрироваться на Github.
6. Создать локальный каталог для выполнения заданий по предмету.

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 (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.

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.

Основные команды `git`:

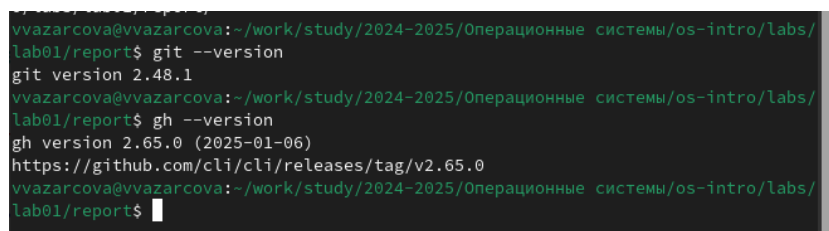
1. `git init` (создать основное древо)
2. `git pull` (получить обновления)
3. `git push` (отправить изменения)
4. `git status` (посмотреть список измененных файлов)

5. `git diff` (посмотреть текущие изменения)
6. `git add {file_name}` (добавить измененные файлы, чтобы добавить все - .)
7. `git rm {file_name}` (удалить измененные файлы)
8. `git commit -am 'Описание коммита'` (сохранить добавленные изменения в коммит)
9. `git checkout {branch_name}` (переключится на ветку, `-b` чтобы создать новую, `-d` чтобы удалить, `-D` чтобы принудительно удалить)

4 Выполнение лабораторной работы

4.1 Базовая конфигурация для работы с Git

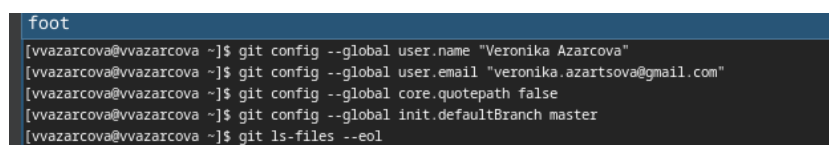
У меня уже установлен git и gh, так что проверяю, что они установлены и что они последней версии (рис. 4.1).



```
vvazarcova@vvazarcova:~/work/study/2024-2025/Операционные системы/os-intro/labs/lab01/report$ git --version
git version 2.48.1
vvazarcova@vvazarcova:~/work/study/2024-2025/Операционные системы/os-intro/labs/lab01/report$ gh --version
gh version 2.65.0 (2025-01-06)
https://github.com/cli/cli/releases/tag/v2.65.0
vvazarcova@vvazarcova:~/work/study/2024-2025/Операционные системы/os-intro/labs/lab01/report$
```

Рис. 4.1: Установленный git и gh

Задаю имя и электронную почту владельца репозитория, настраиваю верификацию и подписание коммитов git и задаю имя начальной ветки - master (рис. 4.2).



```
foot
[vvazarcova@vvazarcova ~]$ git config --global user.name "Veronika Azarcova"
[vvazarcova@vvazarcova ~]$ git config --global user.email "veronika.azartsova@gmail.com"
[vvazarcova@vvazarcova ~]$ git config --global core.quotepath false
[vvazarcova@vvazarcova ~]$ git config --global init.defaultBranch master
[vvazarcova@vvazarcova ~]$ git ls-files --eol
```

Рис. 4.2: Имя и эл. почта владельца

4.2 Ключ SSH

Генерирую ключ SSH.

По алгоритму rsa с ключём размером 4096 бит: (рис. 4.3)

```

[vvazarcova@vvazarcova ~]$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/vvazarcova/.ssh/id_rsa): sshkey
Enter passphrase for "sshkey" (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in sshkey
Your public key has been saved in sshkey.pub
The key fingerprint is:
SHA256:kmrdXDZKSCaZgPMC4evSo6bEyFjHkjsbIU5NAwKrz14 vvazarcova@vvazarcova
The key's randomart image is:
+---[RSA 4096]-----+
|=.                  |
|*... o             |
|o+ o+ o            |
|o =o.+ o           |
|o+=.o + S +        |
|@+ = o = + .       |
|+0*Eo . +          |
|+o.*                |
|=.                  |
+----[SHA256]-----+

```

Рис. 4.3: Ключ SSH по алгоритму rsa

По алгоритму ed25519: (рис. 4.4)

```

bash: /home/vvazarcova/.ssh/id_ed25519.pub: Нет такого файла или каталога
[vvazarcova@vvazarcova ~]$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/vvazarcova/.ssh/id_ed25519): sshkey2
Enter passphrase for "sshkey2" (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in sshkey2
Your public key has been saved in sshkey2.pub
The key fingerprint is:
SHA256:cnxpi2ASDgDn7kLDUafSQ0ebN9I7Bo1c8y3nFBE90sE vvazarcova@vvazarcova
The key's randomart image is:
+--[ED25519 256]--+
|o.o . o=.. |
| +* o ..E |
| oo+.o .. |
|..+oB.o.. |
| +.Bo=+oS++ |
|... +oo+=+ |
|. . + ... |
|. . . |
| |
+-----[SHA256]-----+

```

Рис. 4.4: Ключ SSH по алгоритму ed25519

4.3 Ключ PGP

Генерирую PGP ключ типа RSA and RSA размера 4096 байт с бесконечным сроком действия, введя свою личную информацию, и вывожу на экран список ключей (рис. 4.5).

```
[vvazarcova@vvazarcova ~]$ gpg --list-secret-keys --keyid-format LONG
[keyboard]
-----
sec   rsa4096/ADB00BA77E9094AD 2025-03-02 [SC]
      4E5CC1127C59C1DF24D04AF7ADB00BA77E9094AD
uid   [ a6comotho ] vvazarcova <veronika.azartsova@gmail.com>
ssb   rsa4096/517268EAF0E416D7 2025-03-02 [E]

[vvazarcova@vvazarcova ~]$ gpg --armor --export ADB00BA77E9094AD
-----BEGIN PGP PUBLIC KEY BLOCK-----

mQINBGfEe/MBEAC3wBovV6/HvQfk4uK4Kzm+OKgLSToI6woIAMY9SZUTf3sPuY1Q
kTRjE7Jekd6F6pMk7uZMQWqVmxTfW0ys15mAV5CuoZTheFgG1+Zou5jm1tUSU0ag
I3yUecsEEkt/DzQ/f1RPjlcXqw910ePKAS1FqgKCbLRV00uDFgFE+YJwBNzY2TV0
ZnLz1L1Exuhiz82/hYQUdQYUhsrWf92vodZ4SV7UqY1QFGc5088pgxHwZHLV8Q7d
BFttF5vNBd1ynyBs7f1o9Gt+kN1EfZSIMP4LCS2D90pG+MAGUSSTedmMIGRfJPVV
BHjd+Skt+dGJIBipW1WQ/BCXC5WfYdHz0BeOy71TyNVFmwzwtelFTJvCNhb/oZtf
drrhbX6VeqrqVNBQdb8IZy9KaRfAd5PFxgag2PzwwOya0myTg4YtQA7uYJ3Aumh4E
vIH4B4Hpg2MJqNYbH/We47WYiuSvvd4jHz1jRiihe13/qGjeBkoyHSnZP3jmZvJF
Ye0L/hDt0xaISjth1IdQAWdNUFSBILogALBy55BwYLLsnkV7LZn6xVQzs7afVe2t
7yvYB57y1hqFY0U+tehxR0R70Un0MesvKR+QiUcv/EMz71GDZrse+IZcA/1P/RcB
B9pvdHRqSv68wQGuFnSfUy/ksDpASGX90CMZ8G+/VcZ0aVG8ySrs32wnQARAQAB
tC12dmF5YXJja3ZlbnRlYXN5S5hemFydHNdmdFAZ21haWwvY29tPokCUQQT
AQgAOxYhBE5cwRj8WcHfJNBK962wC6d+kJStBQJnxHvzAhsDBQsJCAcCAIICBUK
CQgLAgQWAGMBAAHAAHEAAAJEK2wC6d+kJStngAP/jnuVNdhs9E7z0PmBY19/teC
vURobz4y74Z0v/4j5y/GJtIPJ5GInMesq7PGbWqM9L3Ue2PhwGnSYi3qCsJMF1K
cWlp+6f6jUwHUIKvsGhU300MS9VRVbdgE1NKwN3uG7+Gz9p8kwa4t0cQLdmfXa
qUsRIdgDM1y7xXg6H5e6Bs0Uqc56jV41BtN5811eQwVmVtECHW0fFvu1YZazRK5s
B22FXn/ZKwn/xP7HZgeSgM4LL/d6JbmEF/iIwpA1WcWzgy14wH2LtJRPnOWME1AF
BnBVq6qW+vBojmXrCdn9chdTbVROo1ota107Kr2/DI+N3aL0cgt0tKP9uE6pKawn
rJ+BQnx505fuq8I7zccblVys/Fnr7+jy8KVnrtaonTF7CXJBDenYt6rc7wF7aVKg
9VD0p0XUSBVGqqhI2k+dWj9dJEJfcgjuetxxXaBgAIJmmGsqj6xbawgYe+NPMUy
015J+UfVvHX93jpfqShmKAnN72r45A1rUfIxttMkw513UPzw4L+JiF75mwqVduV
fyQ4BD6Zf9ytU/KYDv944Zrn/ZnZGazvsSnOV+1lwc3JgbitJbPxbiBtd553g64
BXJHzI1oyewW1DwjJbFk4tZ6QoZmHctQUGXsc12H5RM0awYAs7Et0e9kbegVnBhF
kVf0ld0vif+E6p7YWCsXuQINBGfEe/MBEADJ2ciisK9Qk9GAKvNru2u1EHetwEM6
inIE/8sD0gJybD1Xj9m2GK2szxxXJynWTMAQIq4KuMcsUvuv42azaaYoXs1s12DN
H13KwuseN2J9c15eXKw04JcSwIkuq/xPKxs/AE9LUcERHQ7RkoH+fstnY0Kq/jiM
KwFM09EBSTRBZxtIhQ1/CeU4bHnI3yPIIm7y550+56JAKgma3EABIckTYL5L5en+
TOH086Hs3whj4vILG0gTzS6a1gILtQMxR3N1SZ9IoL1zaQRx6s6bojcw9+WE0/5g
eAPqbk5S970pgyKxuv66RZjRQ+HR4LXm+ffTU3fTkT9FTEYEMJsisrub3HS5Gn8ru
JL6cJ0wFXBJGm5EgwwsvLT3q3bgHvSdtdYHe1H0/HcHfEdIqZKP+cCGujJ+QBwX5
0Jo+mG0abhi7Q+5CJ2pvzv0y3NCzNB1JiHZEexqJ6wnVnOI0ZokD7FkL2wKp+AW6
xLR4ZvM90tdV04px4biykabggLqI/uGwnIotsHguHyD9S4xSzkB1Rx9WYqV749G/
9Zp1wy1X/dtTkzZFNHGHwqJC7s2z0LShQpdkCcIGoPnmGMWQSkuyFUZZWFxofGzr
PVIMsRdFRRUtGAC7sCE0sCFDDb5eL53+jjPpw4pXWZ0psmX174HrC6EIBQxVF7YK
C3z3rLLGy3v+hQARAQABiQI2B8BCAAgFiEET1zBEnx2wd8k0Er3rbALp36Q1K0F
AmfEe/MCGwACgKQrbALp36Q1K1LvQ//XmEavuzNIJUwvZL5D3WG09t/DfXMrng8
h1QzU2myCmb/Odb2MA/F9giroBHHCFtBufzxogLpRkKwpunM1qSfRHwQmAZe6XmP
d5S10Y329I70etbSgShw0bDDeBWBQEEXfgfD5zhqRjK3KPz3wW0/qZv6aIMFimhu
L6qlqzs2Sv1kqI97Th+UgeMiJswXTwsWkPnnJGq/V981BABmWFKhKcQATpphR56r
9qo2/Y3Uuk9v93U1kn0WLSL1Ic30DTgjG7FwDI24fvTrP1PxU/sJvXwQAOoHd9M
EqBn4mT1LmT1Yv2dmEqNwp70jjnH6gczLLuRPN7SL1/Gz4Gz1j1YYXSc6Jvhm86P
LYkkhjQ0/irk7m6h96ZV4PVxWIP1oLK0qm5GJ3tIyaHAFM8pw04sNh8wqW8KwM1h
ceCcXP51mFHxvfZAqWU/hXaUMZd4+RoggyQdP+rNs0K+pvLCIQG55XB51QtDas4i
BPE91mImUdv31Zkvt5iKy9uTcPyH1CacN1HnaZx5m6NYV4cHUfyHHjF0cPsDpP61
5a4Ud1ZM9dJKfM3ZboYUx2ECqCsU7DbqeZRUx2p3cJk5dTw3J1iEs4bLnhIzZxU
uGw1CYh7PNrjW01AB4MYaxwVQVZW0mIs0xT1Qcca+MT9fbgH3p57V7sv+jFu5Rs
oUFbE9TbbpQ=
=ksKj
-----END PGP PUBLIC KEY BLOCK-----
[vvazarcova@vvazarcova ~]$ gpg --armor --export ADB00BA77E9094AD | xclip -sel clip
```

Рис. 4.5: Список ключей

4.4 Настройка подписи Git

Настраиваю автоматические подписи git используя электронную почту, которую я использовала для создания ключа, и авторизуюсь в и настраиваю gh (рис. 4.6).

```
[vvazarcova@vvazarcova ~]$ git config --global user.signingkey ADB00BA77E9094AD
[vvazarcova@vvazarcova ~]$ git config --global commit.gpgsign true
[vvazarcova@vvazarcova ~]$ git config --global gpg.program $(which gpg2)
[vvazarcova@vvazarcova ~]$ gh auth login
bash: gh: команда не найдена
[vvazarcova@vvazarcova ~]$ gh
bash: gh: команда не найдена
[vvazarcova@vvazarcova ~]$ sudo dnf gh
[sudo] пароль для vvazarcova:
Неизвестный аргумент "gh" для команды "dnf5". Добавьте "--help" для получения дополнительной информации об аргументах.
Это может быть команда, предоставляемая плагином, попробуйте: dnf5 install 'dnf5-command(gh)'
[vvazarcova@vvazarcova ~]$ gh
bash: gh: команда не найдена
[vvazarcova@vvazarcova ~]$ gh auth login
bash: gh: команда не найдена
[vvazarcova@vvazarcova ~]$ sudo dnf gh
Неизвестный аргумент "gh" для команды "dnf5". Добавьте "--help" для получения дополнительной информации об аргументах.
Это может быть команда, предоставляемая плагином, попробуйте: dnf5 install 'dnf5-command(gh)'
[vvazarcova@vvazarcova ~]$ sudo dnf install gh
Обновление и загрузка репозитория:
Репозитории загружены.
Пакет
Репозиторий
Арх.
Версия
Разм
gh
updates
x86_64
2.65.0-1.fc41
42.6 M
Сводка транзакции:
Установка: 1 пакета
Общий размер входящих пакетов составляет 10 MiB. Необходимо загрузить 10 MiB.
После этой операции будут использоваться дополнительные 43 MiB (установка 43 MiB, удаление 0 B).
Is this ok [y/N]: y
[1/1] gh-0:2.65.0-1.fc41.x86_64
100% | 1.7 MiB/s | 10.3 MiB | 00m0
6s
[1/1] Total
100% | 1.5 MiB/s | 10.3 MiB | 00m0
7s
Выполнение транзакции
[1/3] Проверить файлы пакета
100% | 28.0 B/s | 1.0 B | 00m00s
[2/3] Подготовить транзакцию
100% | 2.0 B/s | 1.0 B | 00m00s
[3/3] Установка gh-0:2.65.0-1.fc41.x86_64
100% | 25.8 MiB/s | 42.7 MiB | 00m02s
Завершено!
[vvazarcova@vvazarcova ~]$ gh auth login
? Where do you use GitHub? GitHub.com
? What is your preferred protocol for Git operations on this host? SSH
? Generate a new SSH key to add to your GitHub account? No
? How would you like to authenticate GitHub CLI? Login with a web browser
| First copy your one-time code: 2888-E7FE
Press Enter to open https://github.com/login/device in your browser...
restorecon: SELinux: Could not get canonical path for /home/vvazarcova/.mozilla/firefox/*gmp-widevinecdm/* restorecon: No such file
or directory.
```

Рис. 4.6: Настройка автоматических подписей

4.5 Регистрация на Github

Я уже зарегистрирована на github с той электронной почты, которую использовала для двух прошлых шагов. Далее буду использовать свой уже готовый аккаунт.

4.6 Создание локального каталога для работы по предмету

Создаю шаблон рабочего пространства для предмета Операционные системы, перехожу в каталог курса и удаляю лишний файл package.json (рис. 4.7).

```
vvazarcova@vvazarcova:~$ mkdir -p ~/work/study/2022-2023/"Операционные системы"
vvazarcova@vvazarcova:~$ cd ~/work/study/2022-2023/"Операционные системы"
vvazarcova@vvazarcova:~/work/study/2022-2023/Операционные системы$ mkdir -p ~/work/study/20
24-2025/"Операционные системы"
vvazarcova@vvazarcova:~/work/study/2022-2023/Операционные системы$ cd ~/work/study/2024-202
5/"Операционные системы"
vvazarcova@vvazarcova:~/work/study/2024-2025/Операционные системы$ git clone --recursive gi
t@github.com:vvazarcova/study_2024-2025_os-intro.git os-intro
Клонирование в «os-intro»...
remote: Enumerating objects: 36, done.
remote: Counting objects: 100% (36/36), done.
remote: Compressing objects: 100% (35/35), done.
remote: Total 36 (delta 1), reused 21 (delta 0), pack-reused 0 (from 0)
Получение объектов: 100% (36/36), 19.38 КиБ | 413.00 КиБ/с, готово.
Определение изменений: 100% (1/1), готово.
Подмодуль «template/presentation» (https://github.com/yamadharma/academic-presentation-mark
down-template.git) зарегистрирован по пути «template/presentation»
Подмодуль «template/report» (https://github.com/yamadharma/academic-laboratory-report-templ
ate.git) зарегистрирован по пути «template/report»
Клонирование в «/home/vvazarcova/work/study/2024-2025/Операционные системы/os-intro/templat
e/presentation»...
remote: Enumerating objects: 111, done.
remote: Counting objects: 100% (111/111), done.
remote: Compressing objects: 100% (77/77), done.
remote: Total 111 (delta 42), reused 100 (delta 31), pack-reused 0 (from 0)
Получение объектов: 100% (111/111), 102.17 КиБ | 223.00 КиБ/с, готово.
Определение изменений: 100% (42/42), готово.
Клонирование в «/home/vvazarcova/work/study/2024-2025/Операционные системы/os-intro/templat
e/report»...
remote: Enumerating objects: 142, done.
remote: Counting objects: 100% (142/142), done.
remote: Compressing objects: 100% (97/97), done.
remote: Total 142 (delta 60), reused 121 (delta 39), pack-reused 0 (from 0)
Получение объектов: 100% (142/142), 341.09 КиБ | 808.00 КиБ/с, готово.
Определение изменений: 100% (60/60), готово.
Submodule path 'template/presentation': checked out 'c9b2712b4b2d431ad5086c9c72a02bd2fca1d4
a6'
Submodule path 'template/report': checked out 'c26e22effe7b3e0495707d82ef561ab185f5c748'
vvazarcova@vvazarcova:~/work/study/2024-2025/Операционные системы$ cd os-intro
vvazarcova@vvazarcova:~/work/study/2024-2025/Операционные системы/os-intro$
```

Рис. 4.7: Создание рабочего пространства

Создаю нужные каталоги и добавляю их в коммит (рис. 4.8).

```

vvazarcova@vvazarcova:~/work/study/2024-2025/Операционные системы$ cd os-intro
vvazarcova@vvazarcova:~/work/study/2024-2025/Операционные системы/os-intro$ rm package.json
vvazarcova@vvazarcova:~/work/study/2024-2025/Операционные системы/os-intro$ echo os-intro > COURSE
vvazarcova@vvazarcova:~/work/study/2024-2025/Операционные системы/os-intro$ make prepare
vvazarcova@vvazarcova:~/work/study/2024-2025/Операционные системы/os-intro$ git add .
vvazarcova@vvazarcova:~/work/study/2024-2025/Операционные системы/os-intro$ git commit -am 'feat(main):
make course structure'
[master b36e7b7] feat(main): make course structure
405 files changed, 98413 insertions(+), 14 deletions(-)
create mode 100644 labs/README.md
create mode 100644 labs/README.ru.md
create mode 100644 labs/lab01/presentation/.projectile
create mode 100644 labs/lab01/presentation/.texlabroot
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/bib.bib

```

Рис. 4.8: Создание рабочего пространства

Отправляю файлы на сервер (рис. 4.9).

```

vvazarcova@vvazarcova:~/work/study/2024-2025/Операционные системы/os-intro$ git push
Перечисление объектов: 40, готово.
Подсчет объектов: 100% (40/40), готово.
При сжатии изменений используется до 4 потоков
Сжатие объектов: 100% (30/30), готово.
Запись объектов: 100% (38/38), 341.67 КиБ | 1.79 МиБ/с, готово.
Total 38 (delta 4), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (4/4), completed with 1 local object.
To github.com:vvazarcova/study_2024-2025_os-intro.git
69ce19d..b36e7b7 master -> master
vvazarcova@vvazarcova:~/work/study/2024-2025/Операционные системы/os-intro$

```

Рис. 4.9: Создание рабочего пространства

4.7 Контрольные вопросы

1. Что такое системы контроля версий (VCS) и для решения каких задач они предназначены?
VCS (Version Control System) — это система управления изменениями в файлах и коде. Она позволяет отслеживать историю изменений, управлять версиями, работать в команде и откатываться к предыдущим состояниям.
2. Объясните следующие понятия VCS и их отношения: хранилище, commit, история, рабочая копия.
 - хранилище (репозиторий) — место, где хранятся все версии проекта
 - commit — фиксированное изменение в коде с комментарием

- история — последовательность commit'ов, отражающая эволюцию проекта
 - рабочая копия — текущая версия файлов, с которой работает разработчик
3. Что представляют собой и чем отличаются централизованные и децентрализованные VCS? Приведите примеры VCS каждого вида.
- централизованные VCS (CVCS): один сервер содержит репозиторий, разработчики получают и отправляют изменения туда. Пример: SVN
 - децентрализованные VCS (DVCS): каждый разработчик имеет полный репозиторий, можно работать без подключения к серверу. Пример: Git, Mercurial
4. Опишите действия с VCS при единоличной работе с хранилищем.
- Создать репозиторий
 - Добавить файлы и сделать commit
 - Вносить изменения и фиксировать их (commit'ы)
 - Использовать историю для отката при необходимости
 - Создавать ветки для изоляции работы
5. Опишите порядок работы с общим хранилищем VCS.
- Клонировать (скопировать) удалённый репозиторий

- Внести изменения, сделать commit
- Обновить локальные данные (pull)
- Разрешить возможные конфликты
- Отправить изменения в репозиторий (push)
- Использовать ветки для разделения работы

6. Каковы основные задачи, решаемые инструментальным средством Git?

- Хранение версий проекта
- Работа с ветками и параллельной разработкой
- Совместная работа команды
- Откат и восстановление изменений
- Работа с локальными и удалёнными репозиториями

7. Назовите и дайте краткую характеристику командам Git.

- `git init` — создать новый репозиторий
- `git clone` — клонировать репозиторий с сервера
- `git add` — добавить файлы в область подготовки
- `git commit` — сохранить изменения

- `git status` — проверить статус файлов
- `git log` — показать историю коммитов
- `git branch` — работа с ветками
- `git checkout` или `git switch` — переключение между ветками
- `git merge` — объединение веток
- `git pull` — получение изменений из репозитория на сервере
- `git push` — отправка изменений в репозиторий на сервере

8. Приведите примеры использования при работе с локальным и удалённым репозиториями.

- Локальный репозиторий:

```
git init
git add .
git commit -m "Первый коммит"
```

- Удалённый репозиторий:

```
git clone https://github.com/user/repo.git
git pull origin main
git push origin main
```

9. Что такое и зачем могут быть нужны ветви (branches)?

Ветви — это разные линии разработки в репозитории. С ними можно:

- Работать над новыми функциями не меняя основного кода

- Параллельно создавать новые возможности
- Объединять изменения с основной веткой только после тестирования

10. Как и зачем можно игнорировать некоторые файлы при commit?

Для исключения временных, служебных и ненужных файлов. Чтобы это сделать, создаётся файл `.gitignore`, где указываются файлы и каталоги, которые не должны быть в репозитории.

5 Выводы

Подводя итоги выполненной лабораторной работе, я получила практические навыки в работе с системой контроля версий git и создала и настроила рабочее пространство для выполнения заданий по предмету Операционные системы.

Список литературы

1. GDB: The GNU Project Debugger. — URL: <https://www.gnu.org/software/gdb/>.
2. GNU Bash Manual. — 2016. — URL: <https://www.gnu.org/software/bash/manual/>.
3. Midnight Commander Development Center. — 2021. — URL: <https://midnight-commander.org/>.
4. NASM Assembly Language Tutorials. — 2021. — URL: <https://asmtutor.com/>.
5. Newham C. Learning the bash Shell: Unix Shell Programming. — O'Reilly Media, 2005. — 354 с. — (In a Nutshell). — ISBN 0596009658. — URL: <http://www.amazon.com/Learningbash-Shell-Programming-Nutshell/dp/0596009658>.
6. Robbins A. Bash Pocket Reference. — O'Reilly Media, 2016. — 156 с. — ISBN 978-1491941591.
7. The NASM documentation. — 2021. — URL: <https://www.nasm.us/docs.php>.
8. Zarrelli G. Mastering Bash. — Packt Publishing, 2017. — 502 с. — ISBN 9781784396879.
9. Колдаев В. Д., Лупин С. А. Архитектура ЭВМ. — М. : Форум, 2018.