

Отчет по лабораторной работе №2

Операционные системы

БЕМБО ЖОЗЕ ЛУМИНГУ

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
3.1	Установка программного обеспечения	7
3.2	Базовая настройка git	8
3.3	Создание ключа SSH	8
3.4	Создание ключа GPG	9
3.5	Регистрация на Github	11
3.6	Добавление ключа GPG в Github	11
3.7	Настроить подписи Git	13
3.8	Настройка gh	14
3.9	Создание репозитория курса на основе шаблона	15
4	Выводы	18
5	Ответы на контрольные вопросы.	19
	Список литературы	22

Список иллюстраций

3.1	Установка git и gh	7
3.2	Задаю имя и email владельца репозитория	8
3.3	Настройка utf-8 в выводе сообщений git	8
3.4	Задаю имя начальной ветки	8
3.5	Задаю параметры autocrlf и safecrlf	8
3.6	Генерация ssh ключа по алгоритму rsa	9
3.7	Генерация ssh ключа по алгоритму ed25519	9
3.8	Генерация ключа	10
3.9	Защита ключа GPG	10
3.10	Аккаунт на Github	11
3.11	Вывод списка ключей	12
3.12	Копирование ключа в буфер обмена	12
3.13	Настройки GitHub	12
3.14	Добавление нового PGP ключа	13
3.15	Добавленный ключ GPG	13
3.16	Настройка подписей Git	14
3.17	Авторизация в gh	14
3.18	Завершение авторизации через браузер	14
3.19	Завершение авторизации	15
3.20	Создание репозитория	15
3.21	Перемещение между директориями	16
3.22	Удаление файлов и создание каталогов	16
3.23	Отправка файлов на сервер	17
3.24	Отправка файлов на сервер	17

Список таблиц

1 Цель работы

Цель данной лабораторной работы – изучение идеологии и применения средств контроля версий, освоение умения по работе с git.

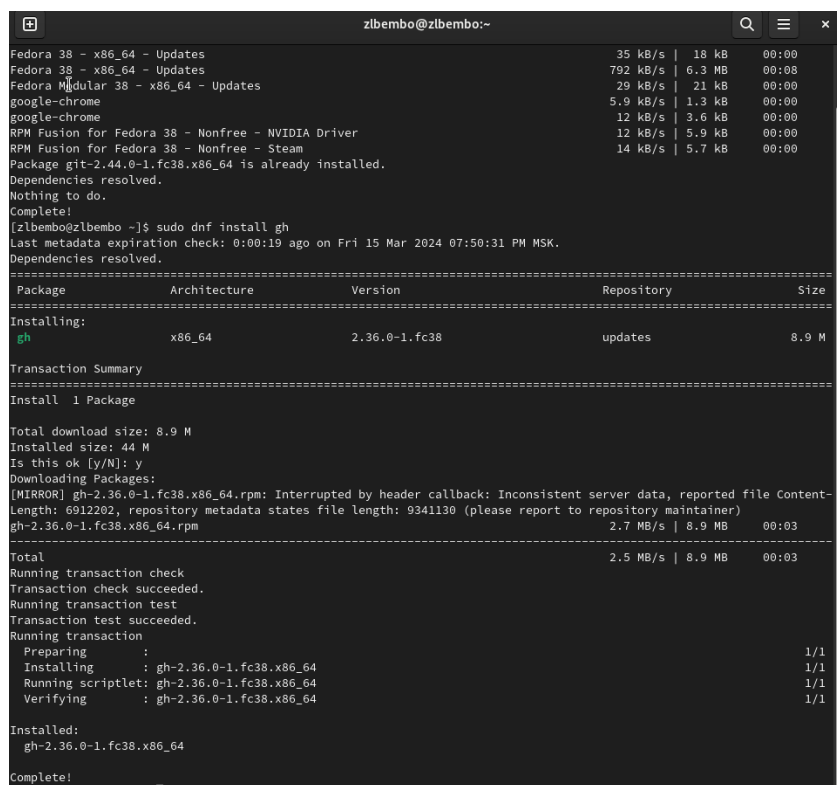
2 Задание

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

3 Выполнение лабораторной работы

3.1 Установка программного обеспечения

Устанавливаю необходимое программное обеспечение git и gh через терминал с помощью команд: `dnf install git` и `dnf install gh` (рис. fig. 3.1).



```
zlbembo@zlbembo:~  
Fedora 38 - x86_64 - Updates 35 kB/s | 18 kB 00:00  
Fedora 38 - x86_64 - Updates 792 kB/s | 6.3 MB 00:08  
Fedora Modular 38 - x86_64 - Updates 29 kB/s | 21 kB 00:00  
google-chrome 5.9 kB/s | 1.3 kB 00:00  
google-chrome 12 kB/s | 3.6 kB 00:00  
RPM Fusion for Fedora 38 - Nonfree - NVIDIA Driver 12 kB/s | 5.9 kB 00:00  
RPM Fusion for Fedora 38 - Nonfree - Steam 14 kB/s | 5.7 kB 00:00  
Package git-2.44.0-1.fc38.x86_64 is already installed.  
Dependencies resolved.  
Nothing to do.  
Complete!  
[zlbembo@zlbembo ~]$ sudo dnf install gh  
Last metadata expiration check: 0:00:19 ago on Fri 15 Mar 2024 07:50:31 PM MSK.  
Dependencies resolved.  
=====
```

Package	Architecture	Version	Repository	Size
gh	x86_64	2.36.0-1.fc38	updates	8.9 M

```
=====
```

Transaction Summary

=====

Install 1 Package

Total download size: 8.9 M
Installed size: 44 M
Is this ok [y/N]: y
Downloading Packages:
[MIRROR] gh-2.36.0-1.fc38.x86_64.rpm: Interrupted by header callback: Inconsistent server data, reported file Content-
Length: 6912202, repository metadata states file length: 9341130 (please report to repository maintainer)
gh-2.36.0-1.fc38.x86_64.rpm 2.7 MB/s | 8.9 MB 00:03

Total	2.5 MB/s 8.9 MB	00:03
-------	-------------------	-------

```
-----  
Running transaction check  
Transaction check succeeded.  
Running transaction test  
Transaction test succeeded.  
Running transaction  
  Preparing      : 1/1  
  Installing     : gh-2.36.0-1.fc38.x86_64 1/1  
  Running scriptlet: gh-2.36.0-1.fc38.x86_64 1/1  
  Verifying      : gh-2.36.0-1.fc38.x86_64 1/1  
Installed:  
  gh-2.36.0-1.fc38.x86_64  
Complete!
```

Рис. 3.1: Установка git и gh

3.2 Базовая настройка git

Задаю в качестве имени и email владельца репозитория свои имя, фамилию и электронную почту (рис. fig. 3.2).

```
[zlbembo@zlbembo ~]$ git config --global user.name "zlbembo1"  
[zlbembo@zlbembo ~]$ git config --global user.email "bembolumingujose@gmail.com"
```

Рис. 3.2: Задаю имя и email владельца репозитория

Настраиваю utf-8 в выводе сообщений git для их корректного отображения (рис. fig. 3.3).

```
[zlbembo@zlbembo ~]$ git config --global core.quotePath false
```

Рис. 3.3: Настройка utf-8 в выводе сообщений git

Начальной ветке задаю имя master (рис. fig. 3.4).

```
[zlbembo@zlbembo ~]$ git config --global init.defaultBranch master
```

Рис. 3.4: Задаю имя начальной ветки

Задаю параметры autocrlf и safecrlf для корректного отображения конца строки (рис. fig. 3.5).

```
[zlbembo@zlbembo ~]$ git config --global core.autocrlf input  
[zlbembo@zlbembo ~]$ git config --global core.safecrlf warn
```

Рис. 3.5: Задаю параметры autocrlf и safecrlf

3.3 Создание ключа SSH

Создаю ключ ssh размером 4096 бит по алгоритму rsa (рис. fig. 3.6).


```
[zlbembo@zlbembo ~]$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/zlbembo/.ssh/id_rsa):
Created directory '/home/zlbembo/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/zlbembo/.ssh/id_rsa
Your public key has been saved in /home/zlbembo/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:Bg0W+v6CQFLMEg9S99vUfPSw9vMkdVdH1SP+k1rZris zlbembo@zlbembo
The key's randomart image is:
+---[RSA 4096]-----+
|+.. .+.  o .B|
|oo+. + o o . = .+|
| o. . o o o = o =|
|. . . = o o .o|
| o  o S  = =|
| . . . @. |
| . ..  o.o|
| . .. E. .|
| .. .oo |
+---[SHA256]-----+
```

Рис. 3.6: Генерация ssh ключа по алгоритму rsa

Создаю ключ ssh по алгоритму ed25519 (рис. fig. 3.7).

```
[zlbembo@zlbembo ~]$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/zlbembo/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/zlbembo/.ssh/id_ed25519
Your public key has been saved in /home/zlbembo/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:D0CqhXwiqC0xjb8CWYx5YQPZjZ4dnGwjbV7BqEkWtPU zlbembo@zlbembo
The key's randomart image is:
+--[ED25519 256]--+
|.oBo+.. |
|. X+%oo |
| 0.%.+E |
|=.0.o o |
|=*o. S |
|=Bo |
|0 . |
|oo |
|.o. |
+---[SHA256]-----+
```

Рис. 3.7: Генерация ssh ключа по алгоритму ed25519

3.4 Создание ключа GPG

Генерирую ключ GPG, затем выбираю тип ключа RSA and RSA, задаю максимальную длину ключа: 4096, оставляю неограниченный срок действия ключа. Далее отвечаю на вопросы программы о личной информации (рис. fig. 3.8).

```
[zlbembo@zlbembo ~]$ gpg --full-generate-key
gpg (GnuPG) 2.4.0; Copyright (C) 2021 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

gpg: directory '/home/zlbembo/.gnupg' created
gpg: keybox '/home/zlbembo/.gnupg/pubring.kbx' created
Please select what kind of key you want:
  (1) RSA and RSA
  (2) DSA and Elgamal
  (3) DSA (sign only)
  (4) RSA (sign only)
  (9) ECC (sign and encrypt) *default*
  (10) ECC (sign only)
  (14) Existing key from card
Your selection? 1
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (3072) 4096
Requested keysize is 4096 bits
Please specify how long the key should be valid.
  0 = key does not expire
  <n> = key expires in n days
  <n>w = key expires in n weeks
  <n>m = key expires in n months
  <n>y = key expires in n years
Key is valid for? (0) 0
Key does not expire at all
Is this correct? (y/N) y

GnuPG needs to construct a user ID to identify your key.

Real name: zlbembo1
Email address: bembolumingujose@gmail.com
Comment:
```

Рис. 3.8: Генерация ключа

Ввожу фразу-пароль для защиты нового ключа (рис. fig. 3.9).

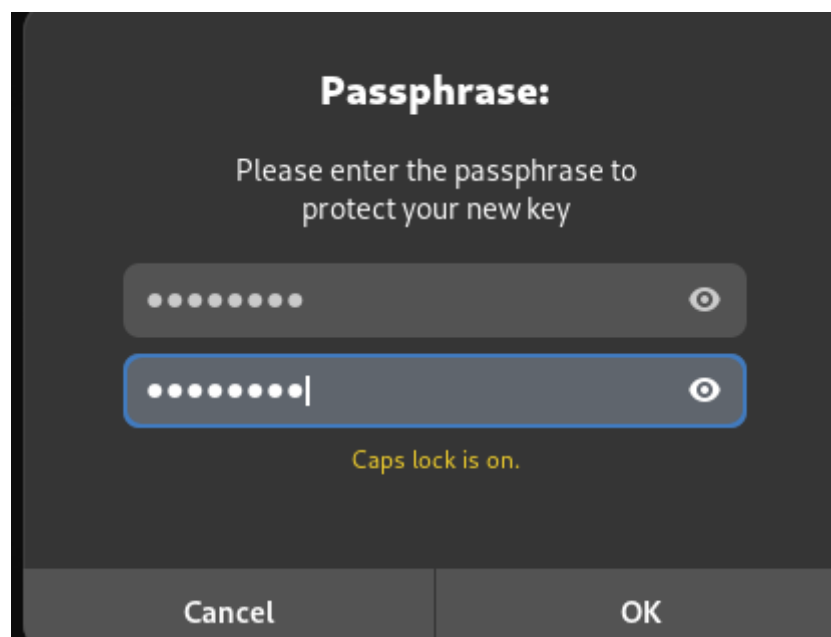


Рис. 3.9: Защита ключа GPG

3.5 Регистрация на Github

У меня уже был создан аккаунт на Github, соответственно, основные данные аккаунта я так же заполняла и проводила его настройку, поэтому просто вхожу в свой аккаунт (рис. fig. 3.10).

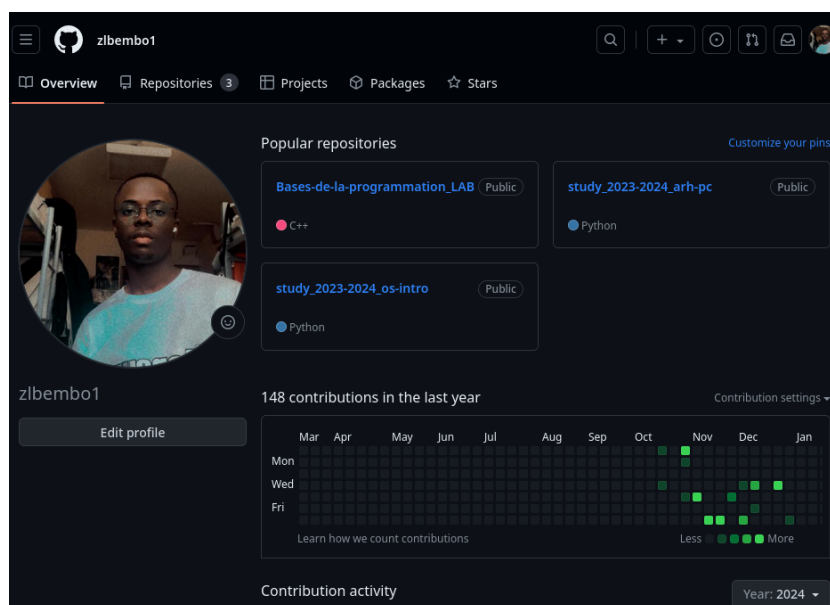


Рис. 3.10: Аккаунт на Github

3.6 Добавление ключа GPG в Github

Вывожу список созданных ключей в терминал, ищу в результате запроса отпечаток ключа (последовательность байтов для идентификации более длинного, по сравнению с самим отпечатком, ключа), он стоит после знака слеша, копирую его в буфер обмена (рис. fig. 3.11).

```
[zlbembo@zlbembo ~]$ gpg --list-secret-keys --keyid-format LONG
gpg: checking the trustdb
gpg: marginals needed: 3 completes needed: 1 trust model: pgp
gpg: depth: 0 valid: 1 signed: 0 trust: 0-, 0q, 0n, 0m, 0f, 1u
/home/zlbembo/.gnupg/pubring.kbx
-----
sec   rsa4096/D26FEF2A4EC71D11 2024-03-15 [SC]
      7B5525C1E7F4F7357B86E3A2D26FEF2A4EC71D11
uid           [ultimate] zlbembo1 <bembolumingujose@gmail.com>
ssb   rsa4096/8B1868F632258481 2024-03-15 [E]
```

Рис. 3.11: Вывод списка ключей

Ввожу в терминале команду, с помощью которой копирую сам ключ GPG в буфер обмена, за это отвечает утилита xclip (рис. fig. 3.12).

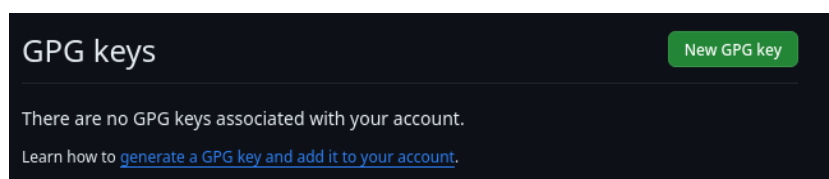


Рис. 3.12: Копирование ключа в буфер обмена

Открываю настройки GitHub, ищу среди них добавление GPG ключа (рис. fig. 3.13).

```
[zlbembo@zlbembo ~]$ gpg --armor --export D26FEF2A4EC71D11 | xclip -sel clip
```

Рис. 3.13: Настройки GitHub

Нажимаю на “New GPG key” и вставляю в поле ключ из буфера обмена (рис. fig. 3.14).

Add new GPG key

Title

zlbembo1

Key

```
-----BEGIN PGP PUBLIC KEY BLOCK-----

mQINBGX0fugBEADSW7aPoN1N4OHjZGtYbkPrNZpzHH4jv9VgwVVIqIWFpleEo2I
1SatM7PLBMZhHT5ZmS1W2dq6q6ITLMjhCu/sRa7kQUd3vwzFBWz4IbWu07jNSLVa
hZDQ33zNLjgXMBVPG7ObldYtIBtpzapLTkzZ2ciKryz8wEbsnKL0sSI/CH3WCV7J
zIECeJ7q3MciFTzTS3GnxUavoc+z4diRwEouQEMuXfG4f6Sufe4LVAr7ntdjBaPL
1sHB41f0tflKwelH8eKF/zZGXcksEei0nzEHwyosajJtwCS9CqjuHdFcCwcfLYKc
OTNdMgw4kPY0OrCc0HTUTlXrZA0JLmoeV9ZXnecMAwfG6wXz2ntpkk0xlyPCDURF
HO26XFjbWC7qMX/Ki59AR7OGzcQr9U3u0ZMtOxYG/qOt1kYO0jEuU6OppmfOVnEQ

-----
```

Add GPG key

Рис. 3.14: Добавление нового PGP ключа

Я добавил ключ GPG на GitHub (рис. fig. 3.15).

GPG keys

New GPG key

This is a list of GPG keys associated with your account. Remove any keys that you do not recognize.


	zlbembo1 Email address: bembolumingujose@gmail.com Key ID: D26FEF2A4EC71D11 Subkeys: 8B1868F632258481 Added on Mar 15, 2024	Delete
---	--	--------

Рис. 3.15: Добавленный ключ GPG

3.7 Настроить подписи Git

Настраиваю автоматические подписи коммитов git: используя введенный ранее email, указываю git использовать его при создании подписей коммитов (рис. fig. 3.16).

```
[zlbembo@zlbembo ~]$ git config --global user.signingkey D26FEF2A4EC71D11
[zlbembo@zlbembo ~]$ git config --global commit.gpgsign true
[zlbembo@zlbembo ~]$ git config --global gpg.program $(which gpg2)
```

Рис. 3.16: Настройка подписей Git

3.8 Настройка gh

Начинаю авторизацию в gh, отвечаю на наводящие вопросы от утилиты, в конце выбираю авторизоваться через браузер (рис. fig. 3.17).

```
[zlbembo@zlbembo ~]$ gh auth login
? What account do you want to log into? GitHub.com
? What is your preferred protocol for Git operations? SSH
? Upload your SSH public key to your GitHub account? /home/zlbembo/.ssh/id_ed25519.pub
? Title for your SSH key: zlbembo1
? How would you like to authenticate GitHub CLI? Login with a web browser
```

Рис. 3.17: Авторизация в gh

Завершаю авторизацию на сайте (рис. fig. 3.18).

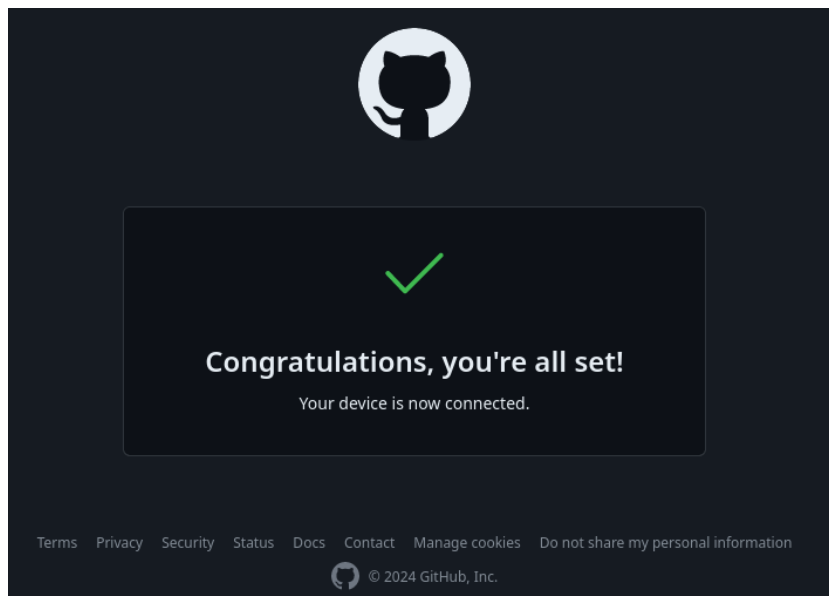


Рис. 3.18: Завершение авторизации через браузер

Вижу сообщение о завершении авторизации под именем zlbembo1 (рис. fig. 3.19).

```

! First copy your one-time code: C1E6-1741
Press Enter to open github.com in your browser...
gh auth login/ Authentication complete.
- gh config set -h github.com git_protocol ssh
/ Configured git protocol
/ Uploaded the SSH key to your GitHub account: /home/zlbembo/.ssh/id_ed25519.pub
/ Logged in as zlbembo1

```

Рис. 3.19: Завершение авторизации

3.9 Создание репозитория курса на основе шаблона

Сначала создаю директорию с помощью утилиты `mkdir` и флага `-p`, который позволяет установить каталоги на всем указанном пути. После этого с помощью утилиты `cd` перехожу в только что созданную директорию “Операционные системы”. Далее в терминале ввожу команду `gh repo create study_2022-2023_os-intro --template yamadharm/course-directory-student-trmplate --public`, чтобы создать репозиторий на основе шаблона репозитория. После этого клонирую репозиторий к себе в директорию, я указываю ссылку с протоколом `https`, а не `ssh`, потому что при авторизации в `gh` выбрала протокол `https` (рис. fig. 3.20).

```

[zlbembo@zlbembo Операционные системы]$ git clone --recursive git@github.com:zlbembo1/study_2023-2024_os-i
intro
Cloning into 'os-intro'...
The authenticity of host 'github.com (140.82.121.3)' can't be established.
ED25519 key fingerprint is SHA256:4DiY3wvvV6TuJJhbpZisF/zLDA0zPMSvHdkr4UvC0qU.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'github.com' (ED25519) to the list of known hosts.
remote: Enumerating objects: 32, done.
remote: Counting objects: 100% (32/32), done.
remote: Compressing objects: 100% (31/31), done.
remote: Total 32 (delta 1), reused 18 (delta 0), pack-reused 0
Receiving objects: 100% (32/32), 18.59 KiB | 4.65 MiB/s, done.
Resolving deltas: 100% (1/1), done.
Submodule 'template/presentation' (https://github.com/yamadharm/academic-presentation-markdown-template.g
ed for path 'template/presentation'
Submodule 'template/report' (https://github.com/yamadharm/academic-laboratory-report-template.git) regist
h 'template/report'
Cloning into '/home/zlbembo/work/study/2023-2024/Операционные системы/os-intro/template/presentation'...
remote: Enumerating objects: 95, done.
remote: Counting objects: 100% (95/95), done.
remote: Compressing objects: 100% (67/67), done.
remote: Total 95 (delta 34), reused 87 (delta 26), pack-reused 0
Receiving objects: 100% (95/95), 96.99 KiB | 871.00 KiB/s, done.
Resolving deltas: 100% (34/34), done.
Cloning into '/home/zlbembo/work/study/2023-2024/Операционные системы/os-intro/template/report'...
remote: Enumerating objects: 126, done.
remote: Counting objects: 100% (126/126), done.
remote: Compressing objects: 100% (87/87), done.
remote: Total 126 (delta 52), reused 108 (delta 34), pack-reused 0
Receiving objects: 100% (126/126), 335.80 KiB | 1.01 MiB/s, done.
Resolving deltas: 100% (52/52), done.
Submodule path 'template/presentation': checked out '40a1761813e197d00e8443ffica72c60a304f24c'
Submodule path 'template/report': checked out '7c31ab8e5dfa8c2b2d67caeb8a19ef8028ced88e'

```

Рис. 3.20: Создание репозитория

Перехожу в каталог курса с помощью утилиты `cd`, проверяю содержание каталога

лога с помощью утилиты `ls` (рис. fig. 3.21).

```
[zlbembo@zlbembo Операционные системы]$ cd ~/work/study/2023-2024/"Операционные системы"/os-intro
```

Рис. 3.21: Перемещение между директориями

Удаляю лишние файлы с помощью утилиты `rm`, далее создаю необходимые каталоги используя `makefile` (рис. fig. 3.22).

```
[zlbembo@zlbembo os-intro]$ rm package.json  
[zlbembo@zlbembo os-intro]$ echo os-intro > COURSE  
[zlbembo@zlbembo os-intro]$ make
```

Рис. 3.22: Удаление файлов и создание каталогов

Добавляю все новые файлы для отправки на сервер (сохраняю добавленные изменения) с помощью команды `git add` и комментирую их с помощью `git commit` (рис. fig. 3.23).


```

[zlbebo@zlbebo os-intro]$ git add .
[zlbebo@zlbebo os-intro]$ git commit -am 'feat(main): make course structure'
[master 5392b4b] feat(main): make course structure
361 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/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
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 100755 labs/lab03/report/pandoc/filters/pandoc_eqnos.py
create mode 100755 labs/lab03/report/pandoc/filters/pandoc_fignos.py

```

Рис. 3.23: Отправка файлов на сервер

Отправляю файлы на сервер с помощью git push (рис. fig. 3.24).

```

[zlbebo@zlbebo os-intro]$ git push
Enumerating objects: 40, done.
Counting objects: 100% (40/40), done.
Delta compression using up to 3 threads
Compressing objects: 100% (30/30), done.
Writing objects: 100% (38/38), 342.11 KiB | 1.68 MiB/s, done.
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:zlbebo1/study_2023-2024_os-intro.git
 243e820..5392b4b master -> master
[zlbebo@zlbebo os-intro]$

```

Рис. 3.24: Отправка файлов на сервер

4 Выводы

При выполнении данной лабораторной работы я изучила идеологию и применение средств контроля версий, освоила умение по работе с git.

5 Ответы на контрольные вопросы.

1. Системы контроля версий (VCS) - программное обеспечение для облегчения работы с изменяющейся информацией. Они позволяют хранить несколько версий изменяющейся информации, одного и того же документа, может предоставить доступ к более ранним версиям документа. Используется для работы нескольких человек над проектом, позволяет посмотреть, кто и когда внес какое-либо изменение и т. д. VCS применяются для: Хранения полной истории изменений, сохранения причин всех изменений, поиска причин изменений и совершивших изменение, совместной работы над проектами.
2. Хранилище – репозиторий, хранилище версий, в нем хранятся все документы, включая историю их изменения и прочей служебной информацией. commit – отслеживание изменений, сохраняет разницу в изменениях. История – хранит все изменения в проекте и позволяет при необходимости вернуться/обратиться к нужным данным. Рабочая копия – копия проекта, основанная на версии из хранилища, чаще всего последней версии.
3. Централизованные VCS (например: CVS, TFS, AccuRev) – одно основное хранилище всего проекта. Каждый пользователь копирует себе необходимые ему файлы из этого репозитория, изменяет, затем добавляет изменения обратно в хранилище. Децентрализованные VCS (например: Git, Bazaar) – у каждого пользователя свой вариант репозитория (возможно несколько вариантов), есть возможность добавлять и забирать изменения из любого

репозитория. В отличие от классических, в распределенных (децентрализованных) системах контроля версий центральный репозиторий не является обязательным.

4. Сначала создается и подключается удаленный репозиторий, затем по мере изменения проекта эти изменения отправляются на сервер.
5. Участник проекта перед началом работы получает нужную ему версию проекта в хранилище, с помощью определенных команд, после внесения изменений пользователь размещает новую версию в хранилище. При этом предыдущие версии не удаляются. К ним можно вернуться в любой момент.
6. Хранение информации о всех изменениях в вашем коде, обеспечение удобства командной работы над кодом.
7. Создание основного дерева репозитория: `git init`

Получение обновлений (изменений) текущего дерева из центрального репозитория: `git pull`

Отправка всех произведённых изменений локального дерева в центральный репозиторий: `git push`

Просмотр списка изменённых файлов в текущей директории: `git status`

Просмотр текущих изменений: `git diff`

Сохранение текущих изменений: добавить все изменённые и/или созданные файлы и/или каталоги: `git add` .

добавить конкретные изменённые и/или созданные файлы и/или каталоги: `git add имена_файлов`

удалить файл и/или каталог из индекса репозитория (при этом файл и/или каталог остаётся в локальной директории): `git rm имена_файлов`

Сохранение добавленных изменений:

сохранить все добавленные изменения и все изменённые файлы: `git commit -am 'Описание коммита'`

сохранить добавленные изменения с внесением комментария через встроенный редактор: `git commit`

создание новой ветки, базирующейся на текущей: `git checkout -b имя_ветки`

переключение на некоторую ветку: `git checkout имя_ветки` (при переключении на ветку, которой ещё нет в локальном репозитории, она будет создана и связана с удалённой)

отправка изменений конкретной ветки в центральный репозиторий: `git push origin имя_ветки`

слияние ветки с текущим деревом: `git merge --no-ff имя_ветки`

Удаление ветки:

удаление локальной уже слитой с основным деревом ветки: `git branch -d имя_ветки`

принудительное удаление локальной ветки: `git branch -D имя_ветки`

удаление ветки с центрального репозитория: `git push origin :имя_ветки`

8. `git push -all` отправляем из локального репозитория все сохраненные изменения в центральный репозиторий, предварительно создав локальный репозиторий и сделав предварительную конфигурацию.
9. Ветвление - один из параллельных участков в одном хранилище, исходящих из одной версии, обычно есть главная ветка. Между ветками, т. е. их концами возможно их слияние. Используются для разработки новых функций.
10. Во время работы над проектом могут создаваться файлы, которые не следуют добавлять в репозиторий. Например, временные файлы. Можно прописать шаблоны игнорируемых при добавлении в репозиторий типов файлов в файл `.gitignore` с помощью сервисов.

Список литературы