

Лабораторная работа №2

Дисциплина: операционные системы

Пронякова Ольга Максимовна

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Ответы на контрольные вопросы:	14
4	Выводы	18
	Список литературы	19

Список иллюстраций

2.1	Регистрация на сайте	6
2.2	Конфигурация git	7
2.3	Генерация ключей	8
2.4	Копирование ключа	9
2.5	Загрузка сгенеренного ключа	10
2.6	Создание репозитория	10
2.7	Создание репозитория	11
2.8	Создание репозитория	11
2.9	Настройка каталога курса	12
2.10	Настройка каталога курса	12
2.11	Настройка каталога курса	12
2.12	Настройка каталога курса	13
2.13	Проверка репозитория	13
2.14	Проверка репозитория	13

Список таблиц

1 Цель работы

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

2 Выполнение лабораторной работы

Создаем учётную запись на сайте <https://github.com/> и заполните основные данные. (рис. 2.1)

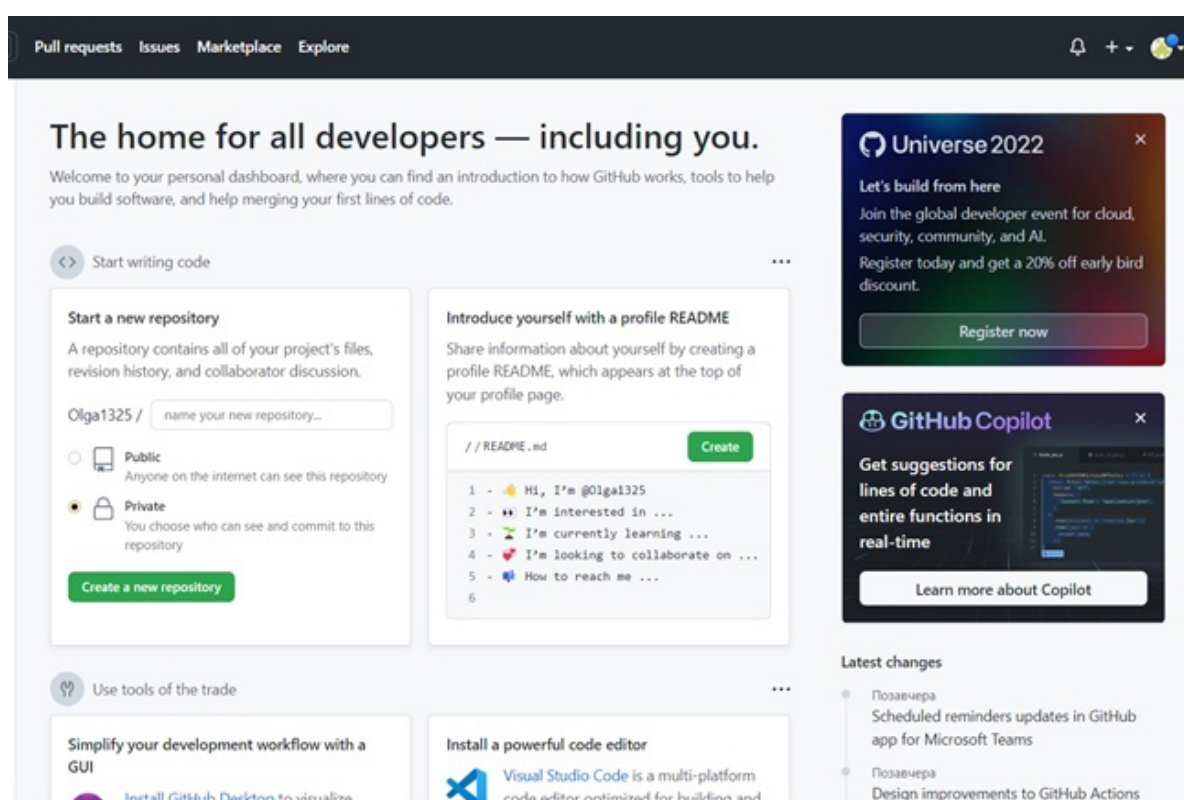


Рис. 2.1: Регистрация на сайте

Сначала сделаем предварительную конфигурацию git. Открываем терминал и вводим следующие команды, указав имя и email владельца репозитория (рис. 2.2).

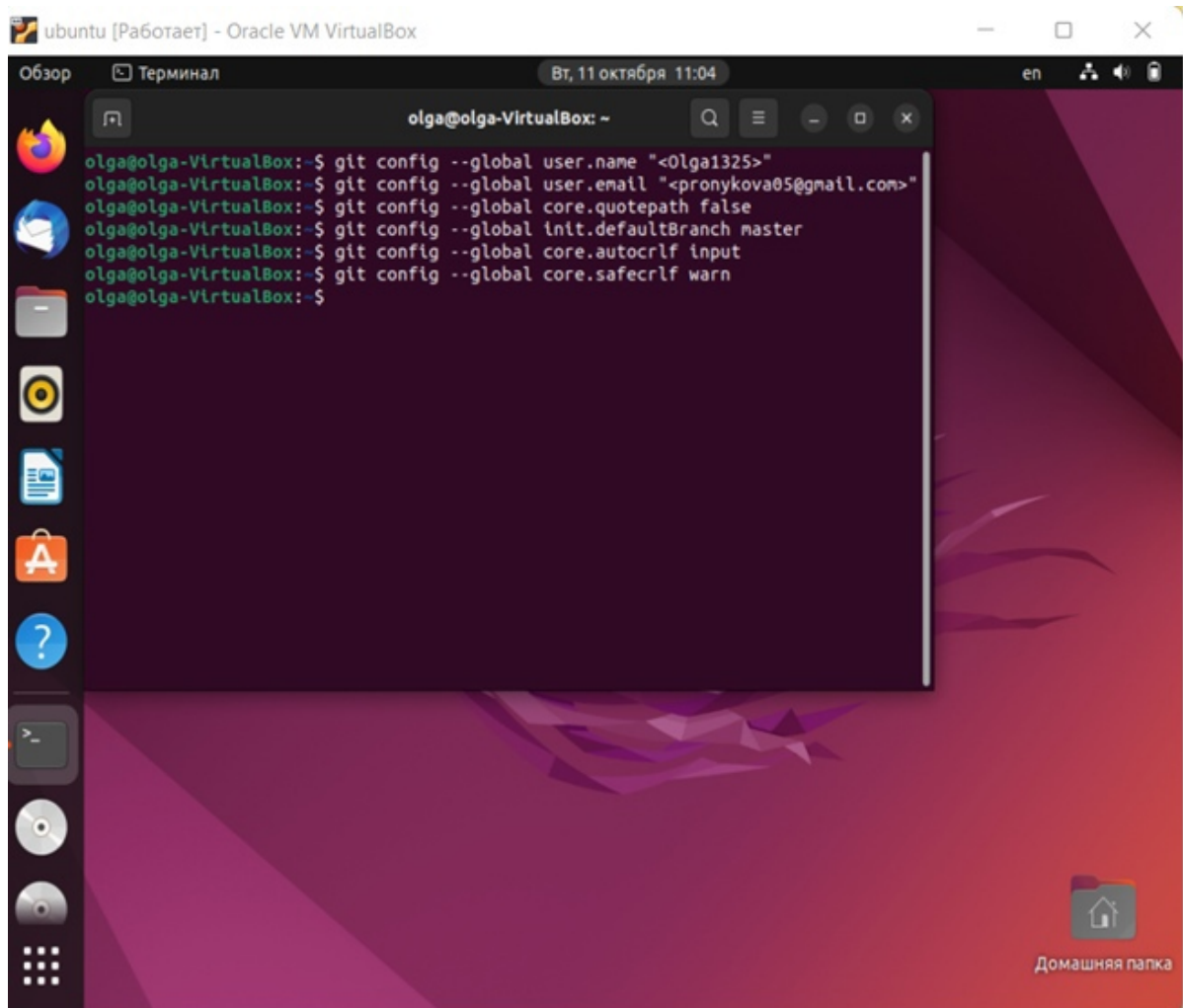


Рис. 2.2: Конфигурация git

Для последующей идентификации пользователя на сервере репозитория необходимо сгенерировать пару ключей (приватный и открытый) (рис. 2.3).

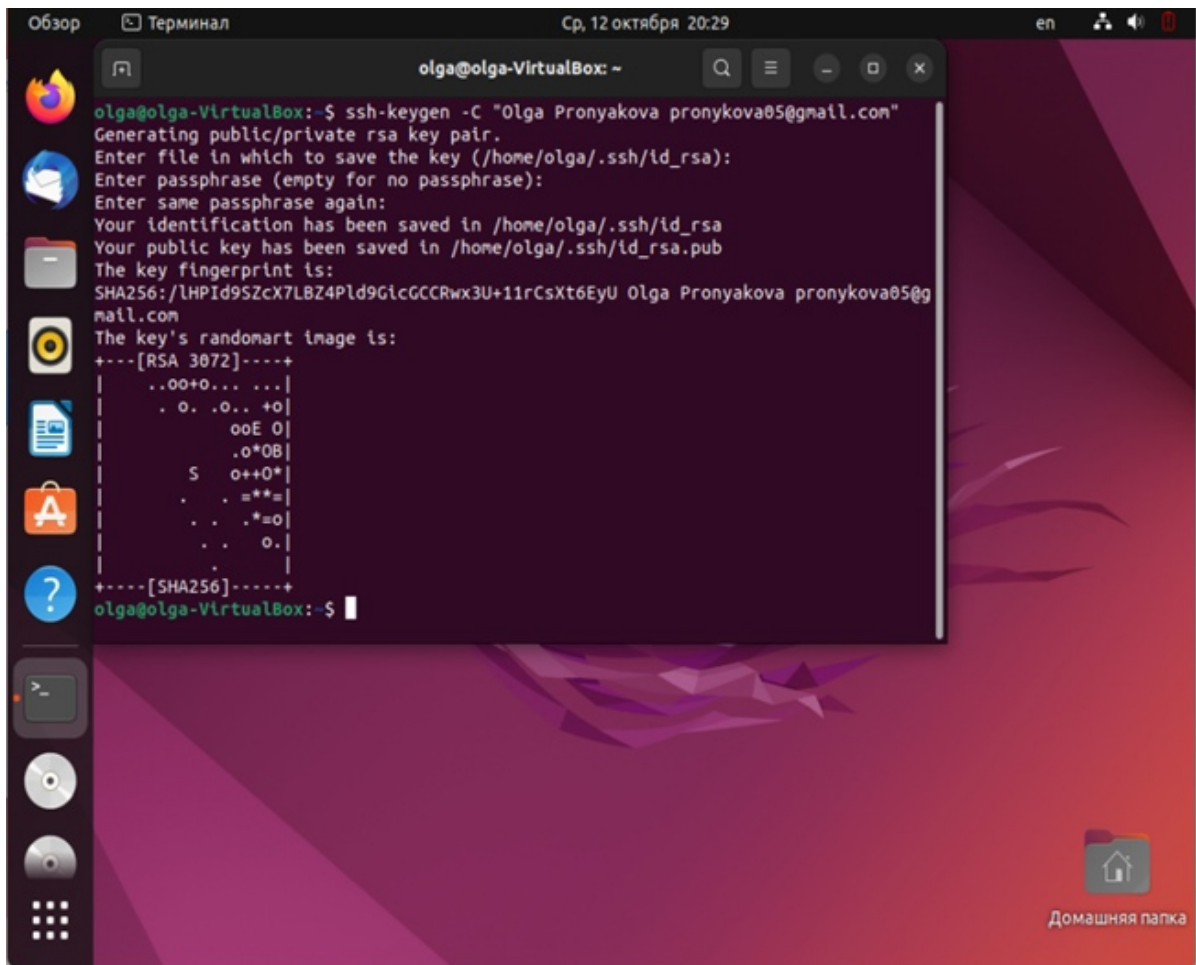


Рис. 2.3: Генерация ключей

Далее необходимо загрузить сгенерённый открытый ключ. Для этого заходим на сайт <http://github.org/> под своей учётной записью и переходим в меню Setting . После этого выберем в боковом меню SSH and GPG keys и нажимаем кнопку New SSH key . Скопировав из локальной консоли ключ в буфер обмена (рис. 2.4) И (рис. 2.5).

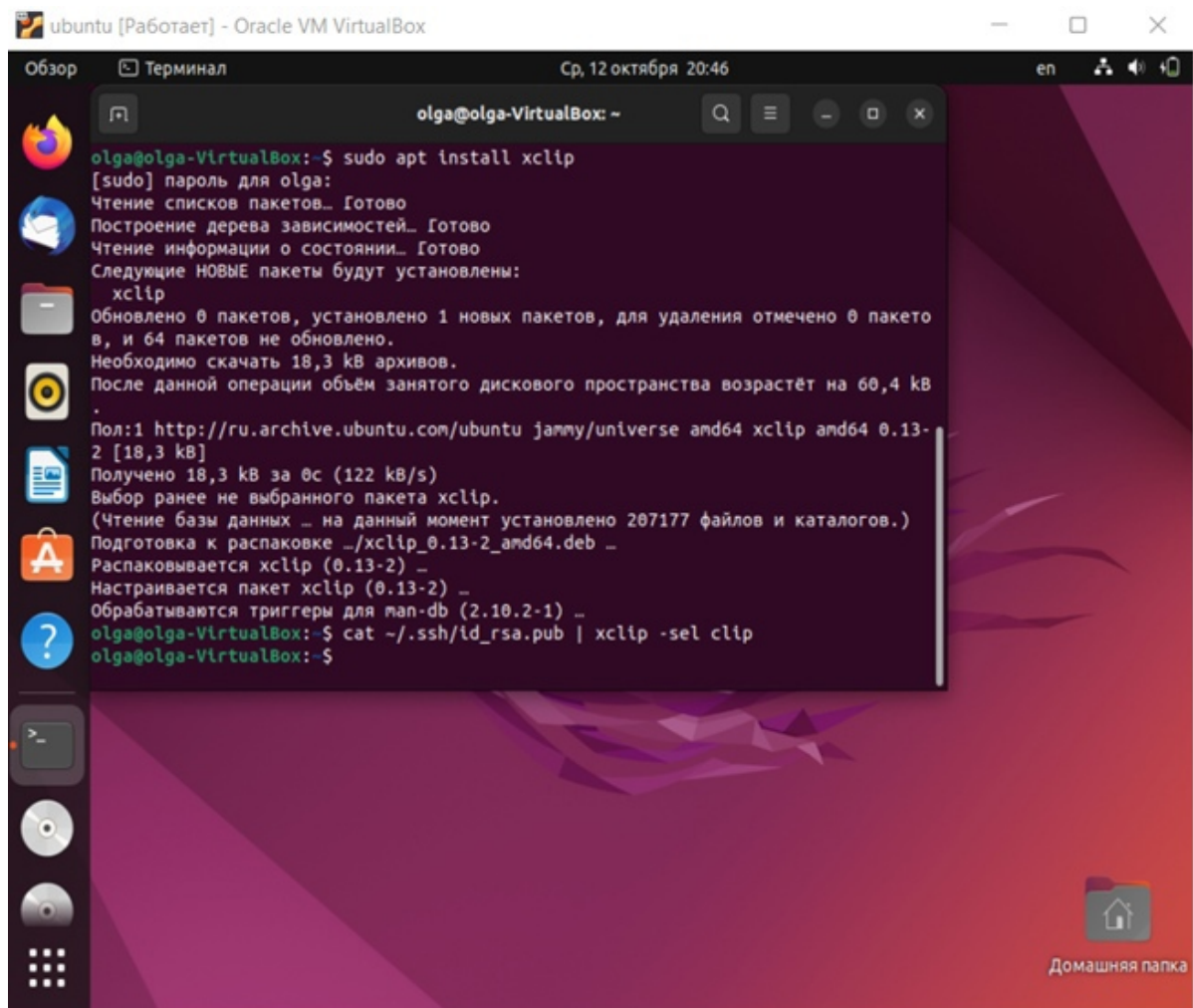


Рис. 2.4: Копирование ключа

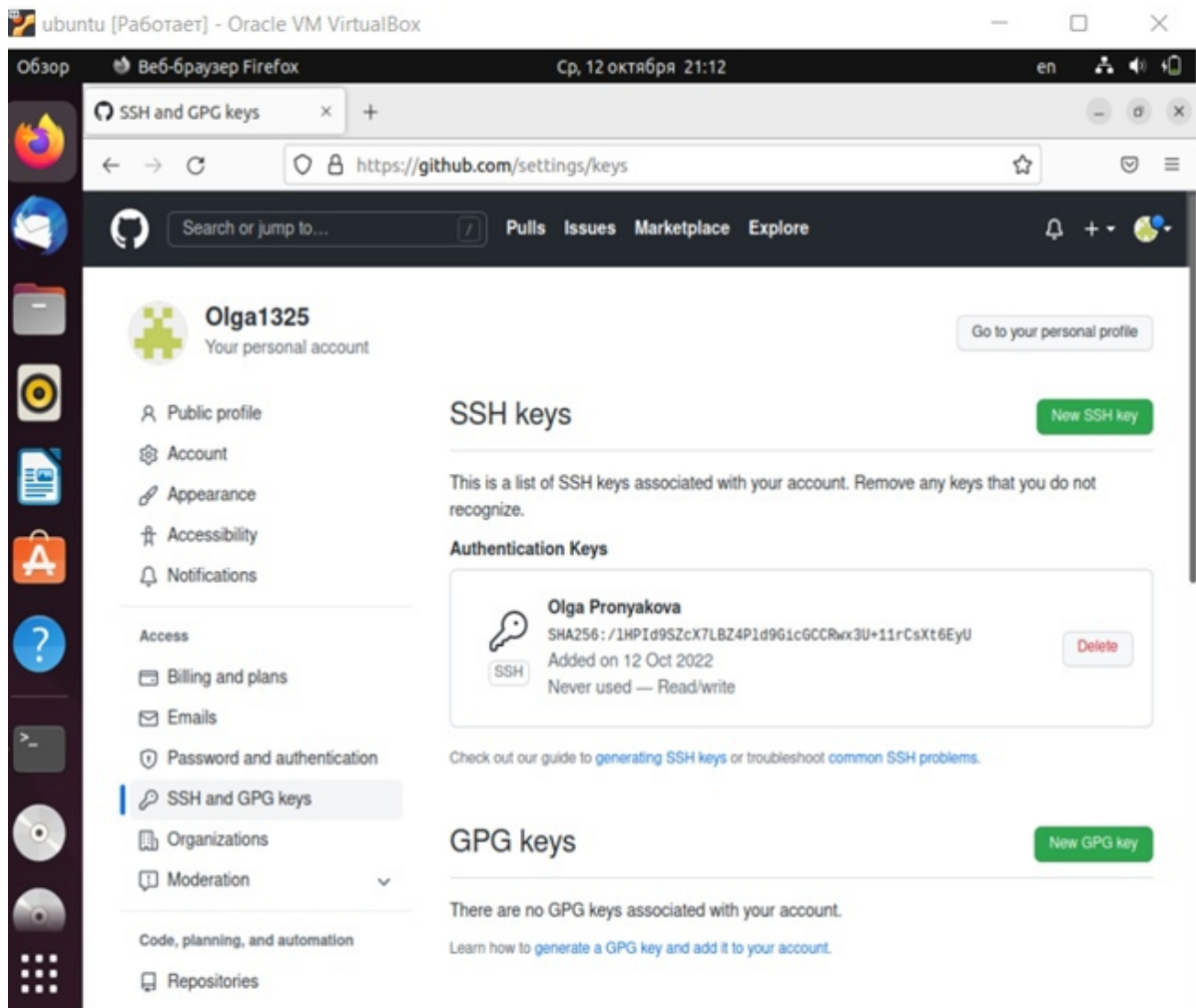


Рис. 2.5: Загрузка сгенеренного ключа

Создаю рабочее пространство(репозиторий) по предмету “Операционные системы”(рис. 2.6) (рис. 2.7) (рис. 2.8).

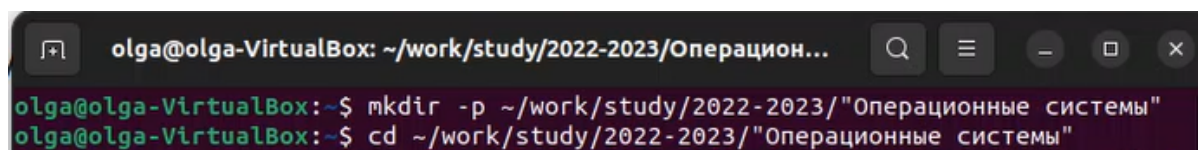


Рис. 2.6: Создание репозитория

```
olga@olga-VirtualBox:~/work/study/2022-2023/Операционные системы$ gh repo create
study_2022-2023_os-intro --template=yamadharm/course-directory-student-templat
e --public
✓ Created repository Olga1325/study_2022-2023_os-intro on GitHub
olga@olga-VirtualBox:~/work/study/2022-2023/Операционные системы$
```

Рис. 2.7: Создание репозитория

```
olga@olga-VirtualBox: ~/work/study/2022-2023/Операцион...
olga@olga-VirtualBox:~/work/study/2022-2023/Операционные системы$ git clone --re
cursive git@github.com:Olga1325/study_2022-2023_os-intro.git os-intro
Клонирование в «os-intro»...
remote: Enumerating objects: 27, done.
remote: Counting objects: 100% (27/27), done.
remote: Compressing objects: 100% (26/26), done.
remote: Total 27 (delta 1), reused 11 (delta 0), pack-reused 0
Получение объектов: 100% (27/27), 16.93 КиБ | 456.00 КиБ/с, готово.
Определение изменений: 100% (1/1), готово.
Подмодуль «template/presentation» (https://github.com/yamadharm/academic-presen
tation-markdown-template.git) зарегистрирован по пути «template/presentation»
Подмодуль «template/report» (https://github.com/yamadharm/academic-laboratory-r
eport-template.git) зарегистрирован по пути «template/report»
Клонирование в «/home/olga/work/study/2022-2023/Операционные системы/os-intro/te
mplate/presentation»...
remote: Enumerating objects: 82, done.
remote: Counting objects: 100% (82/82), done.
remote: Compressing objects: 100% (57/57), done.
remote: Total 82 (delta 28), reused 77 (delta 23), pack-reused 0
Получение объектов: 100% (82/82), 92.90 КиБ | 432.00 КиБ/с, готово.
Определение изменений: 100% (28/28), готово.
Клонирование в «/home/olga/work/study/2022-2023/Операционные системы/os-intro/te
mplate/report»...
```

Рис. 2.8: Создание репозитория

Далее перехожу в каталог курса и удаляю лишние файлы. Затем создаю необ-
ходимые каталоги(рис. 2.9)


```
olga@olga-VirtualBox: ~/work/study/2022-2023/Операцион...
olga@olga-VirtualBox:~$ cd ~/work/study/2022-2023/"Операционные системы"/os-intro
olga@olga-VirtualBox:~/work/study/2022-2023/Операционные системы/os-intro$ rm package.json
olga@olga-VirtualBox:~/work/study/2022-2023/Операционные системы/os-intro$ echo os-intro > COURSE
olga@olga-VirtualBox:~/work/study/2022-2023/Операционные системы/os-intro$ make
olga@olga-VirtualBox:~/work/study/2022-2023/Операционные системы/os-intro$ git add .
```

Рис. 2.9: Настройка каталога курса

Отправляю файлы на сервер(рис. 2.10) (рис. 2.11) (рис. 2.12).

```
olga@olga-VirtualBox:~/work/study/2022-2023/Операционные системы/os-intro$ git commit -am 'feat(main): make course structure'
```

Рис. 2.10: Настройка каталога курса

```
olga@olga-VirtualBox: ~/work/study/2022-2023/Операцион...
create mode 100644 project-personal/stage6/report/Makefile
create mode 100644 project-personal/stage6/report/bib/cite.bib
create mode 100644 project-personal/stage6/report/image/placeimg_800_600_tech.jpg
create mode 100644 project-personal/stage6/report/pandoc/csl/gost-r-7-0-5-2008-numeric.csl
create mode 100755 project-personal/stage6/report/pandoc/filters/pandoc_eqnos.py
create mode 100755 project-personal/stage6/report/pandoc/filters/pandoc_fignos.py
create mode 100755 project-personal/stage6/report/pandoc/filters/pandoc_secnos.py
create mode 100755 project-personal/stage6/report/pandoc/filters/pandoc_tablecontents.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandocxnos/__init__.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandocxnos/core.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandocxnos/main.py
create mode 100644 project-personal/stage6/report/pandoc/filters/pandocxnos/pandocattributes.py
create mode 100644 project-personal/stage6/report/report.md
olga@olga-VirtualBox:~/work/study/2022-2023/Операционные системы/os-intro$
```

Рис. 2.11: Настройка каталога курса

```

olga@olga-VirtualBox:~/work/study/2022-2023/Операционные системы/os-intro$ git push
Перечисление объектов: 40, готово.
Подсчет объектов: 100% (40/40), готово.
При сжатии изменений используется до 2 потоков
Сжатие объектов: 100% (30/30), готово.
Запись объектов: 100% (38/38), 342.40 КиБ | 887.00 КиБ/с, готово.
Всего 38 (изменений 4), повторно использовано 0 (изменений 0), повторно использо
вано пакетов 0
remote: Resolving deltas: 100% (4/4), completed with 1 local object.
To github.com:Olga1325/study_2022-2023_os-intro.git
   db07fe6..1dc5362  master -> master
olga@olga-VirtualBox:~/work/study/2022-2023/Операционные системы/os-intro$

```

Рис. 2.12: Настройка каталога курса

Проверяю наличие данного репозитория и изменения в нем на Github(рис. 2.13) (рис. 2.14).



Рис. 2.13: Проверка репозитория

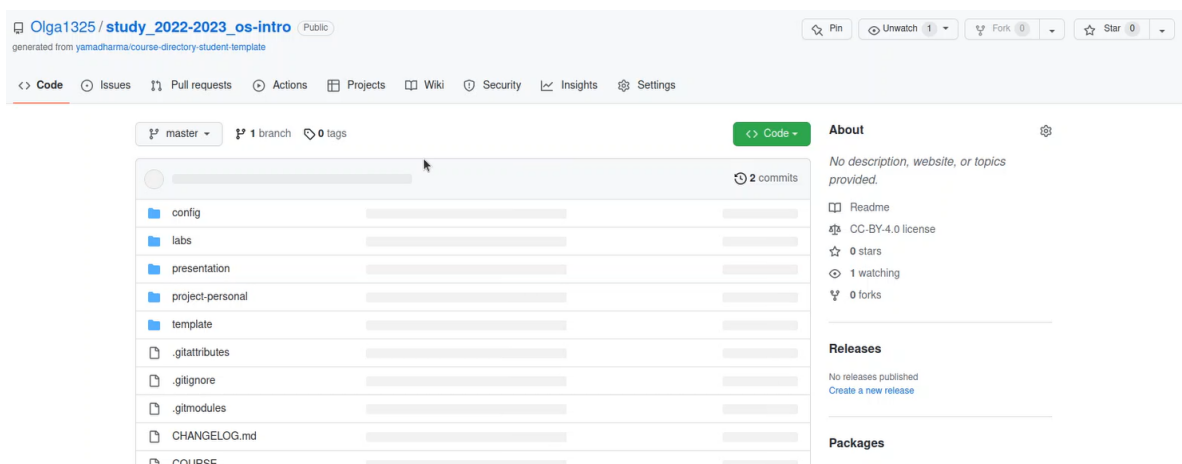


Рис. 2.14: Проверка репозитория

3 Ответы на контрольные вопросы:

1. Что такое системы контроля версий (VCS) и для решения каких задач они предназначены? Система контроля версий — программное обеспечение для облегчения работы с изменяющейся информацией. Система управления версиями позволяет хранить несколько версий одного и того же документа, при необходимости возвращаться к более ранним версиям, определять, кто и когда сделал то или иное изменение, и многое другое. Системы контроля версий (Version Control System, VCS) применяются для:
1)Хранение полной истории изменений 2)причин всех производимых изменений 3)Откат изменений, если что-то пошло не так 4)Поиск причины и ответственного за появления ошибок в программе 5)Совместная работа группы над одним проектом 6)Возможность изменять код, не мешая работе других пользователей
2. Объясните следующие понятия VCS и их отношения: хранилище, commit, история, рабочая копия. Репозиторий - хранилище версий - в нем хранятся все документы вместе с историей их изменения и другой служебной информацией. Commit — отслеживание изменений, сохраняет разницу в изменениях Рабочая копия - копия проекта, связанная с репозиторием (текущее состояние файлов проекта, основанное на версии из хранилища (обычно на последней)) История хранит все изменения в проекте и позволяет при необходимости обратиться к нужным данным.
3. Что представляют собой и чем отличаются централизованные и децентрализованные VCS? Приведите примеры VCS каждого вида. Централизованные

VCS (Subversion; CVS; TFS; VAULT; AccuRev):

Одно основное хранилище всего проекта

Каждый пользователь копирует себе необходимые ему файлы из этого репозитория, изменяет и, затем, добавляет свои изменения обратно

Децентрализованные VCS (Git; Mercurial; Bazaar):

У каждого пользователя свой вариант (возможно не один) репозитория

Присутствует возможность добавлять и забирать изменения из любого репозитория

В классических системах контроля версий используется централизованная модель, предполагающая наличие единого репозитория для хранения файлов. Выполнение большинства функций по управлению версиями осуществляется специальным сервером. В отличие от классических, в распределённых системах контроля версий центральный репозиторий не является обязательным.

4. Опишите действия с VCS при единоличной работе с хранилищем. Сначала создаем и подключаем удаленный репозиторий. Затем по мере изменения проекта отправлять эти изменения на сервер.
5. Опишите порядок работы с общим хранилищем VCS. 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.
6. Каковы основные задачи, решаемые инструментальным средством git? Первая — хранить информацию о всех изменениях в вашем коде, начиная с самой первой строчки, а вторая — обеспечение удобства командной работы над кодом.

7. Назовите и дайте краткую характеристику командам git. Наиболее часто используемые команды git:
- 1) создание основного дерева репозитория: `git init`
 - 2) получение обновлений (изменений) текущего дерева из центрального репозитория: `git pull`
 - 3) отправка всех произведённых изменений локального дерева в центральный репозиторий: `git push`
 - 4) просмотр списка изменённых файлов в текущей директории: `git status`
 - 5) просмотр текущих изменений: `git diff`
 - 6) 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` 1)удалить файл и/или каталог из индекса репозитория (при этом файл и/или каталог остаётся в локальной директории): `git rm имена_файлов` 2)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`
 - 7) создание новой ветки, базирующейся на текущей: `git checkout -b имя_ветки`
 - 8) 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)
 - 9) отправка изменений конкретной ветки в центральный репозиторий: `git push origin имя_ветки`
 - 10) слияние ветки с текущим деревом: `git merge --no-ff имя_ветки`
 - 11) 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. Приведите примеры использования при работе с локальным и удалённым репозиториями. `git push -all` (`push origin master/любой branch`)
9. Что такое и зачем могут быть нужны ветви (branches)? Ветвление («ветка», `branch`) — один из параллельных участков истории в одном хранилище, исходящих из одной версии (точки ветвления).
- Обычно есть главная ветка (`master`), или ствол (`trunk`).
 - Между ветками, то есть их концами, возможно слияние. Используются для разработки новых функций.
10. Как и зачем можно игнорировать некоторые файлы при `commit`? Во время работы над проектом так или иначе могут создаваться файлы, которые не требуется добавлять в последствии в репозиторий. Например, временные файлы, создаваемые редакторами, или объектные файлы, создаваемые компиляторами. Можно прописать шаблоны игнорируемых при добавлении в репозиторий типов файлов в файл `.gitignore` с помощью сервисов.

4 Выводы

Изучила идеологию и применение средств контроля версий. Приобрела практические навыки по работе с системой git.

Список литературы

1. Архитектура ЭВМ
2. Лабораторная работа №2