

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное автономное образовательное учреждение  
высшего образования  
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития  
Кафедра инфокоммуникаций

**ОТЧЕТ**  
**ПО ПРАКТИЧЕСКОЙ РАБОТЕ №1**  
**дисциплины «Программирование на Python»**  
**Вариант \_\_\_\_**

Выполнил:  
Степанов Леонид Викторович  
2 курс, группа ИВТ-б-о-22-1,  
09.03.01 «Информатика и  
вычислительная техника»,  
направленность (профиль)  
«Программное обеспечение  
средств вычислительной  
техники и автоматизирование  
систем», очная форма обучения

---

(подпись)

Руководитель практики:  
Воронкин Р.А., канд. техн. наук,  
доцент, доцент кафедры  
инфокоммуникаций

---

(подпись)

Отчет защищен с оценкой \_\_\_\_\_ Дата защиты \_\_\_\_\_

Ставрополь, 2023 г.

Тема: Исследование основных возможностей Git и GitHub

Цель: исследовать базовые возможности системы контроля версий Git и веб-сервиса для хостинга IT-проектов GitHub.

Порядок выполнения работы:

1. Создал новый репозиторий: общедоступный репозиторий, лицензия MIT, язык программирования Python.

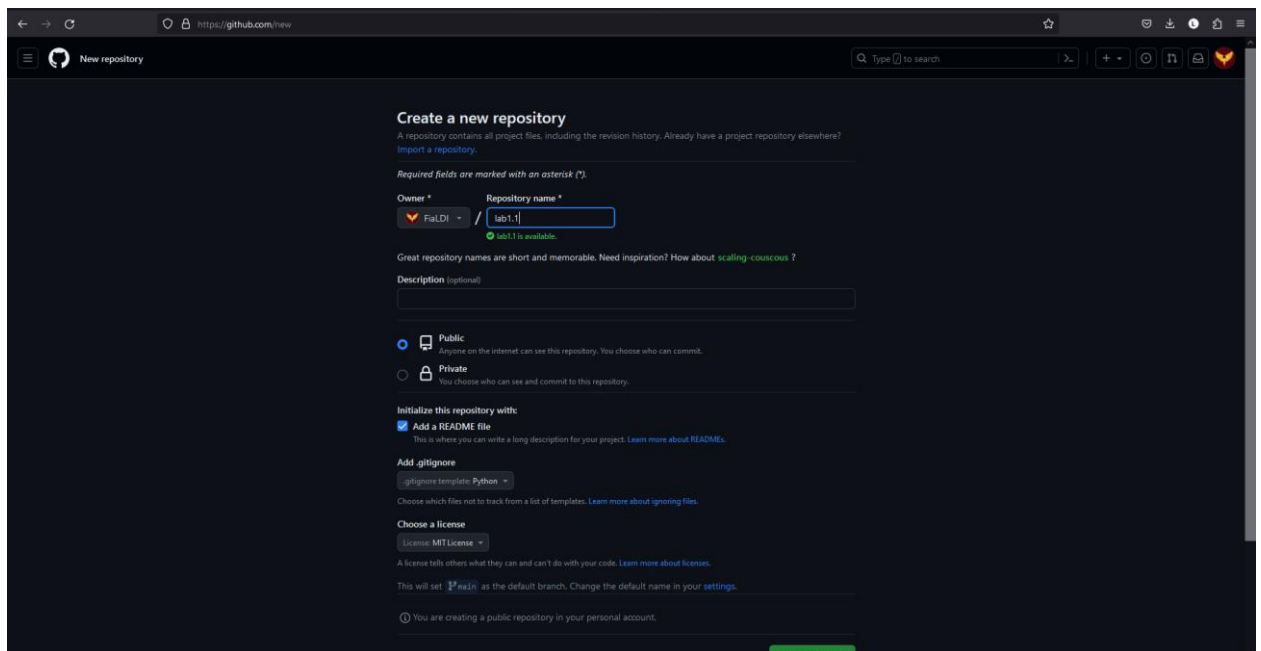


Рисунок 1. Создание репозитория

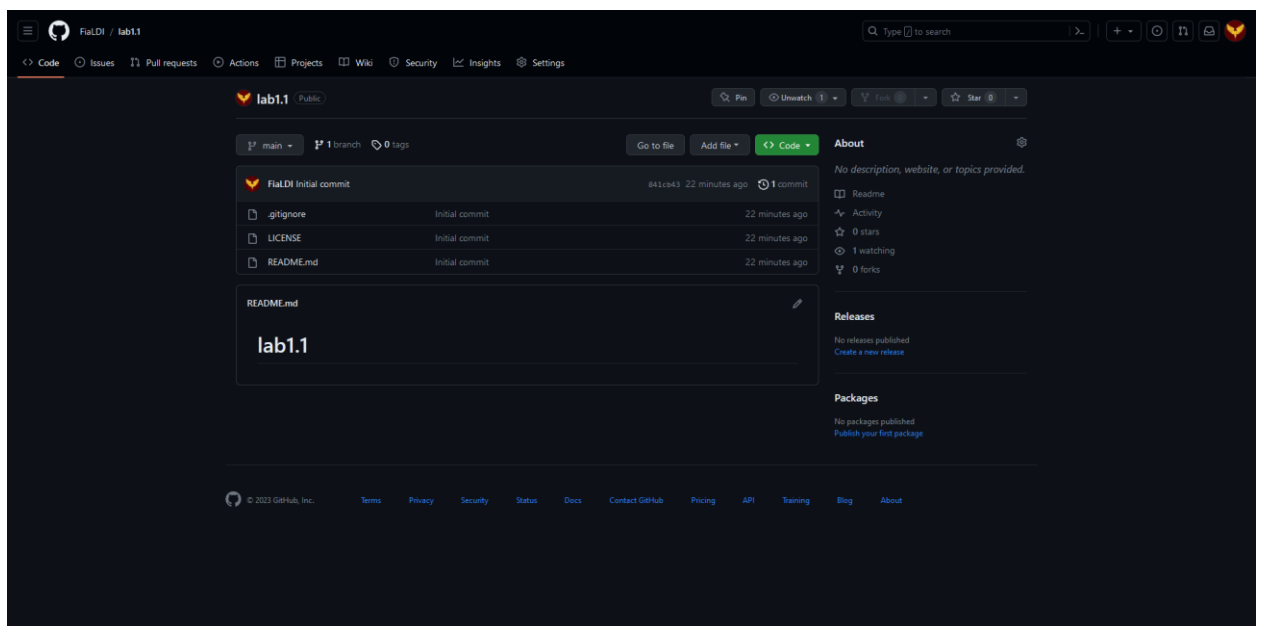
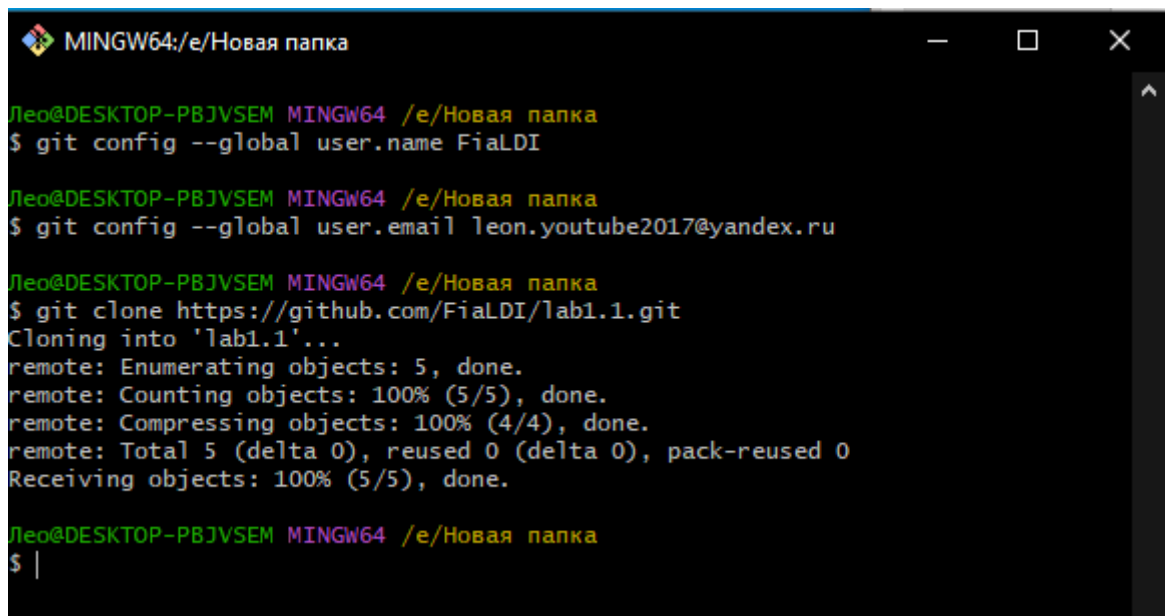


Рисунок 2. Репозиторий

2. Установил имя пользователя и почту и клонировал репозиторий



```
MINGW64:/е/Новая папка

Leo@DESKTOP-PBJVSEM MINGW64 /е/Новая папка
$ git config --global user.name FiaLDI

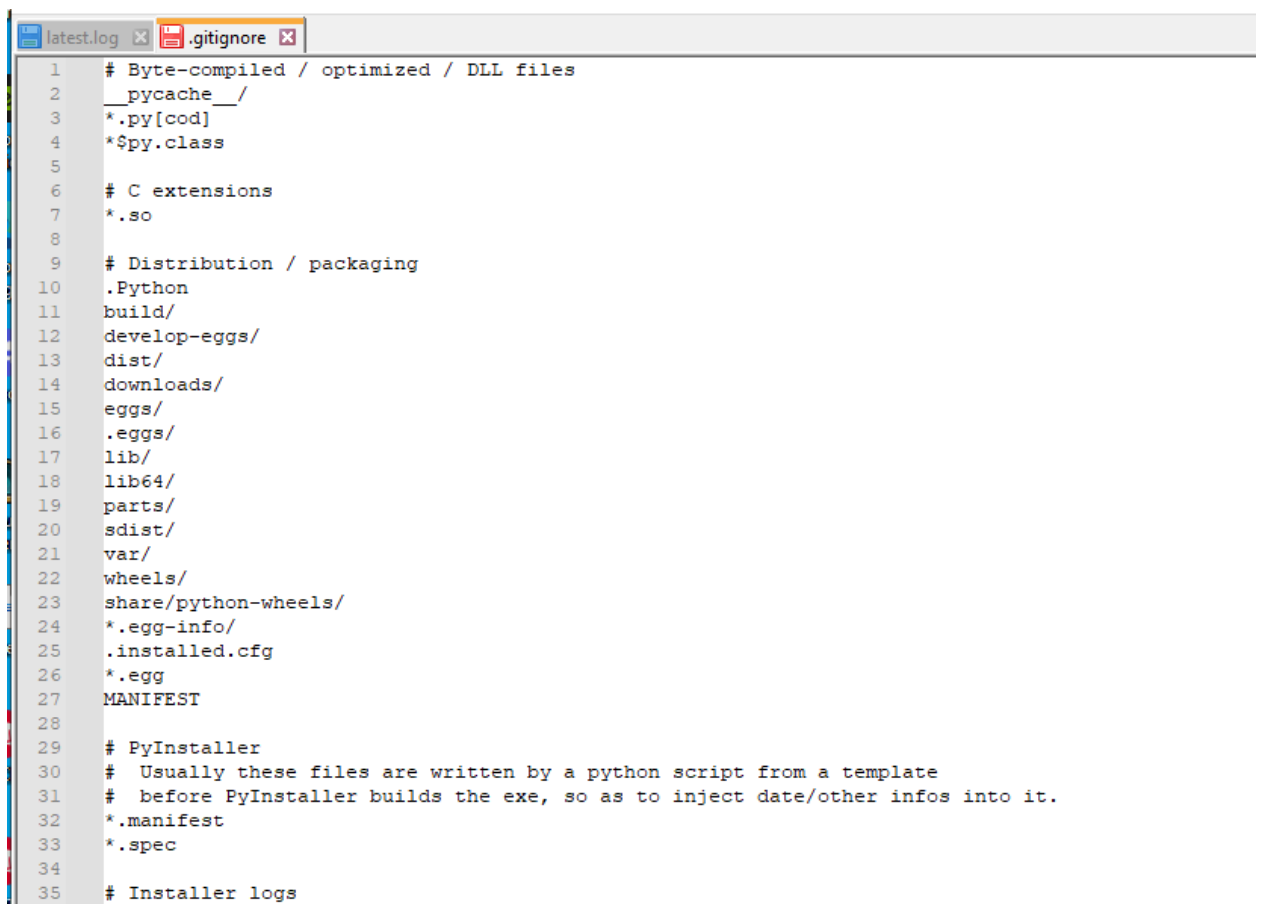
Leo@DESKTOP-PBJVSEM MINGW64 /е/Новая папка
$ git config --global user.email leon.youtube2017@yandex.ru

Leo@DESKTOP-PBJVSEM MINGW64 /е/Новая папка
$ git clone https://github.com/FiaLDI/lab1.1.git
Cloning into 'lab1.1'...
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (5/5), done.

Leo@DESKTOP-PBJVSEM MINGW64 /е/Новая папка
$ |
```

Рисунок 3. Клонирование репозитория

### 3. Дополнил файл .gitignore



```
latest.log x .gitignore x
1 # Byte-compiled / optimized / DLL files
2 __pycache__ /
3 *.py[co]
4 *.py.class
5
6 # C extensions
7 *.so
8
9 # Distribution / packaging
10 .Python
11 build/
12 develop-eggs/
13 dist/
14 downloads/
15 eggs/
16 .eggs/
17 lib/
18 lib64/
19 parts/
20 sdist/
21 var/
22 wheels/
23 share/python-wheels/
24 *.egg-info/
25 .installed.cfg
26 *.egg
27 MANIFEST
28
29 # PyInstaller
30 # Usually these files are written by a python script from a template
31 # before PyInstaller builds the exe, so as to inject date/other infos into it.
32 *.manifest
33 *.spec
34
35 # Installer logs
```

Рисунок 4. Дополнение файла .gitignore

### 4. Добавил в файл README.md информацию о себе

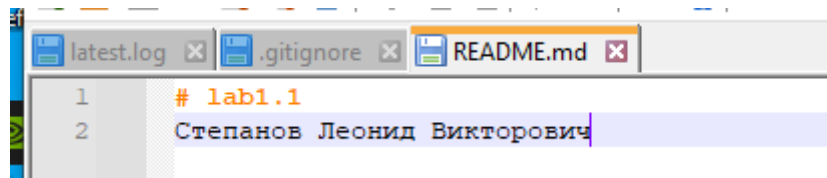


Рисунок 5. Изменения файла README.md

5. Написал программу на Python и сделал 7 commit

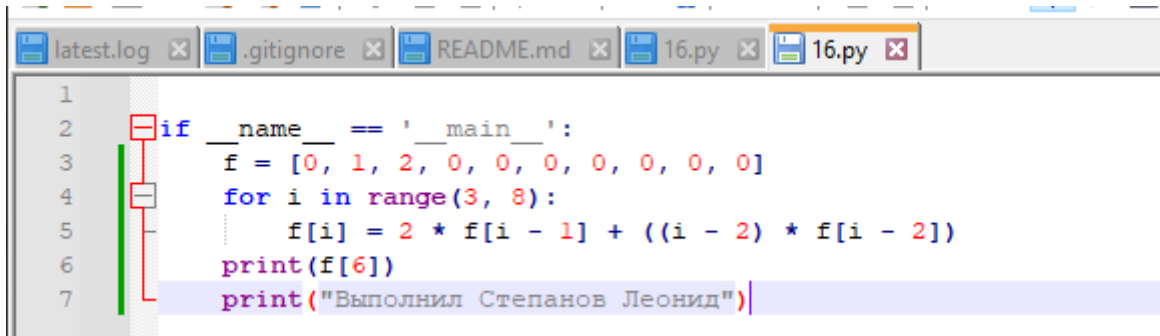


Рисунок 6. Программа на Python

6. Добавил README и зафиксировал

```
Leo@DESKTOP-PBJVSEM MINGW64 ~/Desktop/git/lab1.1 (main)
$ git add README

Leo@DESKTOP-PBJVSEM MINGW64 ~/Desktop/git/lab1.1 (main)
$ git commit -m 'Добавил README'
[main 00ebd2b] Добавил README
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 README

Leo@DESKTOP-PBJVSEM MINGW64 ~/Desktop/git/lab1.1 (main)
```

Рисунок 7. Добавление README

7. Отправил все файлы на github

```
Leo@DESKTOP-PBJVSEM MINGW64 ~/Desktop/git/lab1.1 (main)
$ git push
Enumerating objects: 28, done.
Counting objects: 100% (28/28), done.
Delta compression using up to 12 threads
Compressing objects: 100% (25/25), done.
Writing objects: 100% (27/27), 3.24 KiB | 662.00 KiB/s, done.
Total 27 (delta 7), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (7/7), done.
To https://github.com/FialDI/lab1.1.git
841cb43..e0cc81a main -> main

Leo@DESKTOP-PBJVSEM MINGW64 ~/Desktop/git/lab1.1 (main)
```

Рисунок 8. Отправка файлов на GitHub

Выводы: исследовал базовые возможности системы контроля версий Git и веб-сервиса для хостинга IT-проектов GitHub.

### Ответы на контрольные вопросы:

1. Система контроля версий (СКВ) — это система, регистрирующая изменения в одном или нескольких файлах с тем, чтобы в дальнейшем была возможность вернуться к определённым старым версиям этих файлов.

2. Локальные: можно наделать много ошибок, случайно изменить не тот файл или копировать не те файлы.

Центральные: единая точка отказа.

3. К распределённым СКВ.

4. Подход Git к хранению данных отличается от остальных тем, что коммит сохраняет каждый файл в данный момент.

5. В Git для всего вычисляется хеш-сумма, и только потом происходит сохранение.

6. Зафиксированное, изменённое и подготовленное.

Зафиксированный значит, что файл уже сохранён в вашей локальной базе.

К изменённым относятся файлы, которые поменялись, но ещё не были зафиксированы.

Подготовленные файлы — это изменённые файлы, отмеченные для включения в следующий коммит.

7. Профиль пользователя – публичная страница GitHub

8. Репозитории бывают публичные, приватные.

9. Upseteam - репозиторий на GitHub

Origin – копия на GitHub

Local – это добавленные gitbush в commit

Working copy – это на локальном компьютере

10. Проверка версии: git version

Ввод данных, имени и почты:

1) git config --global user.name <YOUR\_NAME>

2) git config --global user.email <EMAIL>

11. В правом углу нажать: new repositories.

Затем ввод: Имени Репозитория, Описания, Public/private., .git ignore, LICENSE

## 12. Типы поддерживаемых лицензий

Таблица 1. Лицензии GitHub

License	License keyword
Academic Free License v3.0	afl-3.0
Apache license 2.0	apache-2.0
Artistic license 2.0	artistic-2.0
Boost Software License 1.0	bsl-1.0
BSD 2-clause "Simplified" license	bsd-2-clause
BSD 3-clause "New" or "Revised" license	bsd-3-clause
BSD 3-clause Clear license	bsd-3-clause-clear
BSD 4-clause "Original" or "Old" license	bsd-4-clause
BSD Zero-Clause license	0bsd
Creative Commons license family	cc
Creative Commons Zero v1.0 Universal	cc0-1.0
Creative Commons Attribution 4.0	cc-by-4.0
Creative Commons Attribution ShareAlike 4.0	cc-by-sa-4.0
Do What The F*ck You Want To Public License	wtfpl
Educational Community License v2.0	ecl-2.0
Eclipse Public License 1.0	epl-1.0
Eclipse Public License 2.0	epl-2.0
European Union Public License 1.1	eupl-1.1

GNU Affero General Public License v3.0	agpl-3.0
GNU General Public License family	gpl
GNU General Public License v2.0	gpl-2.0
GNU General Public License v3.0	gpl-3.0
GNU Lesser General Public License family	lgpl
GNU Lesser General Public License v2.1	lgpl-2.1
GNU Lesser General Public License v3.0	lgpl-3.0
ISC	isc
LaTeX Project Public License v1.3c	lppl-1.3c
Microsoft Public License	ms-pl
MIT	mit
Mozilla Public License 2.0	mpl-2.0
Open Software License 3.0	osl-3.0
PostgreSQL License	postgresql
SIL Open Font License 1.1	ofl-1.1
University of Illinois/NCSA Open Source License	ncsa
The Unlicense	unlicense
zLib License	zlib

13. Осуществляется клонирование при помощи команды: `git clone`. Оно осуществляется для создания репозитория на локальном ПК.

14. `Git status`

15. Добавление/изменение файла в локальный репозиторий Git:

1) После добавления/изменения файла в локальный репозиторий Git, файл будет находиться в рабочем каталоге.

2) Файл не будет автоматически добавлен в индекс (staging area) для фиксации.

Добавление нового/измененного файла под версионный контроль с помощью команды `git add`:

1) Команда `git add` используется для добавления файлов в индекс (staging area) для фиксации.

2) После выполнения команды `git add`, файлы будут добавлены в индекс и будут готовы для фиксации (коммита).

Фиксация (коммит) изменений с помощью команды `git commit`:

1) Команда `git commit` используется для фиксации изменений в локальном репозитории Git.

2) После выполнения команды `git commit`, изменения будут зафиксированы и сохранены в истории репозитория.

Отправка изменений на сервер с помощью команды `git push`:

1) Команда `git push` используется для отправки изменений из локального репозитория на удаленный сервер Git.

2) После выполнения команды `git push`, изменения будут отправлены на сервер и будут доступны для других пользователей.

16. `git clone <URL репозитория> <путь к локальной папке>`

17. GitLab, BitBucket

GitHub обладает широкой пользовательской базой и интеграцией с другими инструментами разработки, в то время как GitLab предлагает больше возможностей для управления жизненным циклом приложений.

18. GitHub Desktop: GitHub Desktop - это официальное приложение от GitHub, которое предоставляет простой и интуитивно понятный интерфейс для работы с Git. Оно позволяет клонировать репозитории, создавать ветки, вносить изменения, делать коммиты и пушить изменения на удаленный сервер



GitHub. Операции Git, такие как создание веток и коммиты, реализуются через соответствующие кнопки и меню в приложении.

Приведу пример реализации некоторых операций Git с помощью GitHub Desktop для всего есть отдельная кнопка:

1. Клонирование репозитория:
2. Создание ветки
3. Внесение изменений и коммит: