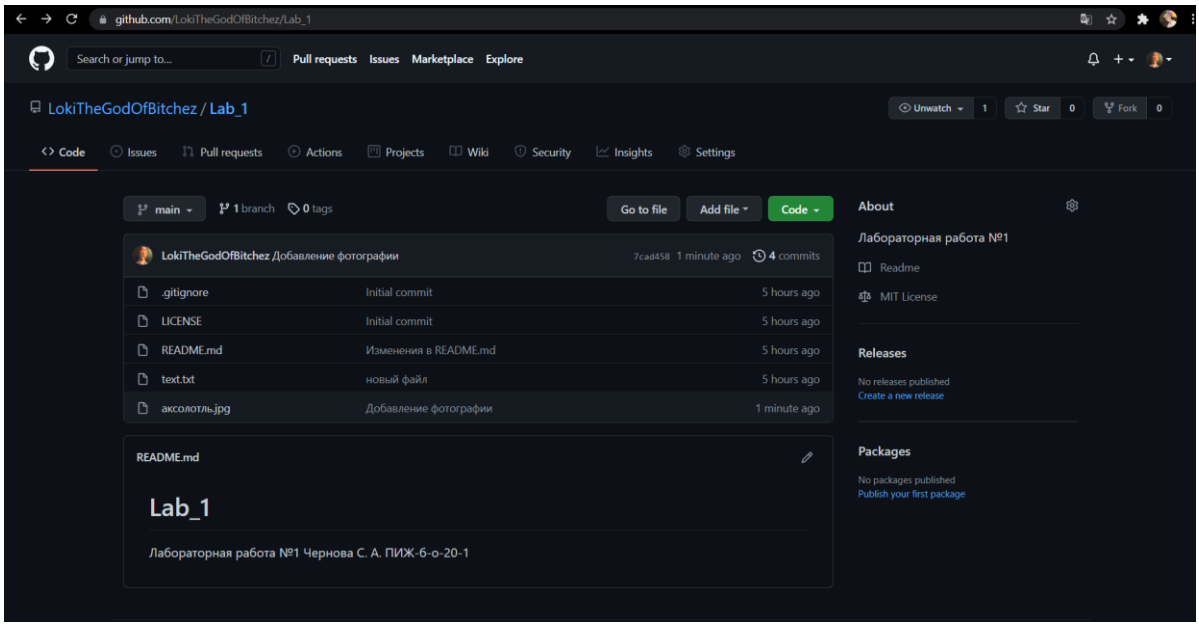
Лабораторная работа №1
Исследование основных возможностей Git и GitHub
Чернова С. А.
ПИЖ-б-о-20-1

Скриншоты команд и их результат:

```
C:\Users\User> D:

D:\> cd lab\Lab_1

D:\lab\Lab_1>git status
On branch main
Your branch is up to date with 'origin/main'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        "\320\260\320\272\321\201\320\276\320\273\320\276\321\202\320\273\321\214.jpg"

nothing added to commit but untracked files present (use "git add" to track)

D:\lab\Lab_1>git add .

D:\lab\Lab_1>git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   "\320\260\320\272\321\201\320\276\320\273\320\276\321\202\320\273\321\214.jpg"


D:\lab\Lab_1>git commit -m"Добавление фотографии"
[main 7cad458] Добавление фотографии
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 "\320\260\320\272\321\201\320\276\320\273\320\276\321\202\320\273\321\214.jpg"

D:\lab\Lab_1>git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean

D:\lab\Lab_1>git push
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 6.10 KiB | 6.10 MiB/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 https://github.com/LokiTheGodOfBitchez/Lab_1.git
   ca98bc5..7cad458  main -> main
```



Ответы на вопросы:

1. Что такое СКВ и каково ее назначение?

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. В чем недостатки локальных и централизованных СКВ?

Недостаток локального в отсутствии связи программистов между собой, а централизованного – в ненадежности физических носителей и элементов системы (серверов, дисков и прочего).

3. К какой СКВ относится Git?

К распределенной системе контроля версий

4. В чем концептуальное отличие Git от других СКВ?

В методе хранения и обработке данных, которые больше похожи на снимки, чем на обычное хранилище

5. Как обеспечивается целостность хранимых данных в Git?

Через хеш-сумму

6. В каких состояниях могут находиться файлы в Git? Как связаны эти состояния?

У Git есть три основных состояния, в которых могут находиться файлы: 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. Что такое профиль пользователя в GitHub?

Публичная страница пользователя

8. Какие бывают репозитории в GitHub?

Общие и частные

9. Укажите основные этапы модели работы с GitHub.

Стандартный подход к работе с проектом состоит в том, чтобы иметь локальную копию репозитория и фиксировать изменения в этой копии, а не в удаленном репозитории, размещенном на GitHub. Этот локальный репозиторий имеет полную историю версий проекта, которая может быть полезна при разработке без подключения к интернету.

10. Как осуществляется первоначальная настройка Git после установки?

Необходимо сначала проверить, был ли Git установлен успешно, с помощью команды git version в терминале, а затем следует указать имя и адрес электронной почты, связанный с аккаунтом GitHub с помощью следующих команд:

Git config --global user.name “your.name”

Git config –global user.email “your.email”

11. Опишите этапы создания репозитория в GitHub.

После нажатие кнопки “New repository” необходимо указать некоторые важный поля:

- Имя репозитория. Оно необязательно должно быть уникальным во всем github, но уникальным в рамках репозиториев одного аккаунта
- Описание, которое можно оставить пустым
- Public\private если необходимо, чтобы репозиторий был публичным либо закрытым
- .gitignor и license для игнорируемых файлов и для лицензирования репозитория

12. Какие типы лицензий поддерживаются GitHub при создании репозитория?

Apache License 2.0
GNU General Public License v3.0
MIT License
BSD 2-Clause "Simplified" License
BSD 3-Clause "New" or "Revised" License
Boost Software License 1.0
Creative Commons Zero v1.0 Universal
Eclipse Public License 2.0
GNU Affero General Public License v3.0
GNU General Public License v2.0
GNU Lesser General Public License v2.1
Mozilla Public License 2.0
The Unlicense

13. Как осуществляется клонирование репозитория GitHub? Зачем нужно клонировать репозиторий?

С помощью команды “git clone”. Клонирование необходимо для работы с файлами репозитория на разных устройствах – изменения файлов, их добавления и удаления и т. д.

14. Как проверить состояние локального репозитория Git?

Командой “git status”

15. Как изменяется состояние локального репозитория Git после выполнения следующих операций: добавления/изменения файла в локальный репозиторий Git; добавления нового/измененного файла под версионный контроль с помощью команды git add ; фиксации (коммита) изменений с помощью команды git commit и отправки изменений на сервер с помощью команды git push ?

Появятся незафиксированные файлы (выделены красным шрифтом); новые файлы будут зафиксированы в локальном репозитории (выделены зеленым шрифтом); изменения станут зафиксированными в хранилище, репозиторий обновится

16. У Вас имеется репозиторий на GitHub и два рабочих компьютера, с помощью которых Вы можете осуществлять работу над некоторым проектом с использованием этого репозитория. Опишите последовательность команд, с помощью которых оба локальных репозитория, связанных с репозиторием GitHub будут находиться в синхронизированном состоянии. Примечание: описание необходимо начать с команды git clone.

17. GitHub является не единственным сервисом, работающим с Git. Какие сервисы еще Вам известны? Приведите сравнительный анализ одного из таких сервисов с GitHub.

Bitbucket, GUI Clients, GitKraken и пр.

18. Интерфейс командной строки является не единственным и далеко не самым удобным способом работы с Git. Какие Вам известны программные средства с графическим интерфейсом пользователя для работы с Git? Приведите как реализуются описанные в лабораторной работе операции Git с помощью одного из таких программных средств

- **GitHub** Desktop.
- Fork.
- Tower.
- Sourcetree.
- SmartGit.
- Sublime Merge.
- GitKraken.
- GitUp.