

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ ФГАОУ ВО
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития
Кафедра инфокоммуникаций

Отчет по лабораторной работе №1

По дисциплине основы кроссплатформенного программирования
«Исследования основных возможностей Git и GitHub»

Выполнила:

студентк группы ИТС-б-о-21-1

Абдикодиров Жахонгир Хуснитдин
угли

(подпись)

Проверил: Доцент, к.т.н, доцент
кафедры

инфокоммуникаций

Воронкин Р. А.


Работа защищена с оценкой:


(подпись)

Ставрополь, 2022

```
C:\Users\Admin> git config --global user.name Jahongir
C:\Users\Admin> git config --global user.email jahongirabdikadirov@gmail.com
C:\Users\Admin> git config --global user.password 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
C:\Users\Admin>
```

Для работы с гитом я внес данные своего гита на консоль, воспользовался командами **git config --global** и в зависимости от данных, добавил свои пользовательские данные

☒  **Public**
Anyone on the internet can see this repository. You choose who can commit.


☐  **Private**
You choose who can see and commit to this repository.


Initialize this repository with:
Skip this step if you're importing an existing repository.

☒ **Add a README file**
This is where you can write a long description for your project. [Learn more.](#)

Add .gitignore
Choose which files not to track from a list of templates. [Learn more.](#)

Choose a license
A license tells others what they can and can't do with your code. [Learn more.](#)


This will set  **main** as the default branch. Change the default name in your [settings](#).

 You are creating a public repository in your personal account.

Create repository

Для выполнения лабораторной работы, создал новый репозиторий и в него добавил

README file;
gitignore (python);
и **MIT license.**


 **Clone**

HTTPS

SSH

GitHub CLI

https://github.com/dzsesakq/1_LR.git



Use Git or checkout with SVN using the web URL.

После создания нового репозитория скопировал ссылку чтобы клонировать его

```
C:\Users\Admin> cd C:\Users\Admin\Desktop\1LR  
C:\Users\Admin\Desktop\1LR> git clone https://github.com/dzsesakq/1_LR.git  
Cloning into '1_LR'...  
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.  
C:\Users\Admin\Desktop\1LR>
```

Также создал папку, где будут сохраняться коды. Скопировал его путь и с помощью консоли открыл. Клонировал свой репозиторий с помощью команды **git clone**

```
1_LR > 1.py > ...  
You, 1 секунду назад | 1 author (You)  
1 s = input("write your name")  
2 print(s)  
3 b = input("write your surname")  
4  
ПРОБЛЕМЫ Выходные данные ТЕРМИНАЛ ... powershell + - [ ] [ ] [ ] [ ] [ ] [ ]  
Compressing objects: 100% (2/2), done.  
Writing objects: 100% (3/3), 380 bytes | 190.00 KiB/s, done.  
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0  
To https://github.com/dzsesakq/1_LR.git  
8fa437c..6255ff6 main -> main  
PS C:\Users\Admin\Desktop\1\1_LR> git add .  
PS C:\Users\Admin\Desktop\1\1_LR> git commit -m "2"  
[main 57aea23] 2  
1 file changed, 1 insertion(+)  
PS C:\Users\Admin\Desktop\1\1_LR> git push  
Enumerating objects: 5, done.  
Counting objects: 100% (5/5), done.  
Delta compression using up to 4 threads  
Compressing objects: 100% (3/3), done.  
Writing objects: 100% (3/3), 388 bytes | 194.00 KiB/s, done.  
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0  
To https://github.com/dzsesakq/1_LR.git  
6255ff6..57aea23 main -> main  
PS C:\Users\Admin\Desktop\1\1_LR>
```

Написала маленький код, как было сказано в методических указаниях, и с помощью команды **git add .** сохранил. Потом с помощью команды **git commit** добавил комментарий и в итоге с помощью команды **git push** отправил свой написанный код в гит

The screenshot shows the Visual Studio Code interface. On the left is the Explorer sidebar with a file tree containing `1_LR`, `.gitignore`, `1.py`, `LICENSE`, and `README.md`. The main editor displays the `1.py` file with the following Python code:

```
1_LR > 1.py > ...
You, 1 секунду назад | 1 author (You)
1 a = input("write your name")
2 print(a)
3 b = input("write your surname")
4 print(b)
5 c = input("write your age")
6 print(a)
7 print(("my friend " + a (" , or dear ") + b (" , you are very
8 print("you are only " + c) You, 32 секунды назад * end ...
```

At the bottom, the integrated terminal shows the execution of Git commands:

```
PS C:\Users\Admin\Desktop\1\1_LR> git commit -m "correct"
[main d8608b6] correct
1 file changed, 1 insertion(+), 1 deletion(-)
PS C:\Users\Admin\Desktop\1\1_LR> git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 382 bytes | 191.00 KiB/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/dzsesakq/1_LR.git
172f1d6..d8608b6 main -> main
PS C:\Users\Admin\Desktop\1\1_LR>
```

Такую же процедуры по методическим указаниям, необходимо было сделать 7 раз, я свой написанный код разбил на семь частей и сделал необходимое количество коммитов

The screenshot shows the GitHub repository page for `dzsesakq/1_LR`. The repository is public and has 1 branch and 0 tags. The commit history shows a recent commit by `dzsesakq` with the message `correct` (commit hash `d8608b6`), which includes changes to `.gitignore`, `1.py`, `LICENSE`, and `README.md`. Below the commit list, the `README.md` file is displayed with the content:

```
1_LR
1
```

Все сделано

1. Что такое СКВ и каково ее назначение?

Системы контроля версий — это программные инструменты, помогающие командам разработчиков управлять изменениями в исходном коде с течением времени.

2. В чем недостатки локальных и централизованных СКВ?

Недостатки:

- сложности с поиском необходимой версии в обширной и плохо структурированной базе данных;
- возможность потери данных вследствие возникновения физических поломок оборудования;
- отсутствие возможности совместной разработки.

3. К какой СКВ относится Git?

Git — распределённая система контроля версий, которая даёт возможность разработчикам отслеживать изменения в файлах и работать над одним проектом совместно с коллегами.

4. В чем концептуальное отличие Git от других СКВ?

В основу Git закладывались концепции, призванные создать более быструю распределенную систему контроля версий, в противовес правилам и решениям, использованным в CVS.

5. Как обеспечивается целостность хранимых данных в Git?

При разработке в Git прежде всего обеспечивается целостность исходного кода под управлением системы. Содержимое файлов, а также объекты репозитория, фиксирующие взаимосвязи между файлами, каталогами, версиями, тегами и коммитами, защищены при помощи криптографически стойкого алгоритма хеширования SHA1.

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

Файлы могут быть не отслеживаемые (Untracked) и отслеживаемые. Отслеживаемые файлы могут находиться в 3 состояниях: Не изменено (Unmodified), изменено (Modified), подготовленное (Staged).

7. Что такое профиль пользователя в GitHub?

Профиль - это ваша публичная страница на GitHub, как и в социальных сетях. Когда вы ищете работу в качестве программиста, работодатели могут посмотреть ваш профиль GitHub и принять его во внимание, когда будут решать, брать вас на работу или нет

8. Какие бывают репозитории в GitHub?

Репозитории бывают как основные, т. е. официально поддерживаемые, так и дополнительные, которые можно подключить в случае возникновения необходимости (например, программы, которую Вы искали, нет в официальном репозитории).

10. Как осуществляется первоначальная настройка Git после установки?

В состав Git входит утилита `git config`, которая позволяет просматривать и настраивать параметры, контролирующие все аспекты работы Git, а также его внешний вид.

11. Опишите этапы создания репозитория в GitHub.

Перейдите на <https://github.com> и войдите в свой аккаунт. Нажмите кнопку New repository (Новый репозиторий). На открывшейся странице введите имя репозитория (Repository name) и нажмите кнопку Create repository. Данная команда добавит удаленный репозиторий с именем origin, который указывает на ваш Github-репозиторий.

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

Клонирование репозитория с GitHub через Git на Windows

1. Перейдите в папку «Мои документы»;
2. Создайте папку с названием «Git». ...
3. Кликните правой кнопкой мыши в папке «Git» и выберите «Git Bash Here». ...

4. Пойдёт процесс клонирования, он не долгий, несколько секунд;

Второй вариант создания директории для контроля версий – копирование существующего проекта с другого сервера. Это актуально, когда осуществляется доработка готового проекта или вы желаете внедрить его компоненты в свой.

14. Как проверить состояние локального репозитория Git? 1

Проверьте состояние репозитория

Используйте команду `git status`, чтобы проверить текущее состояние репозитория.

ВЫПОЛНИТЕ:

```
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` ?

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.

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.