

Федеральное государственное автономное образовательное учреждение
высшего образования «Северо-Кавказский федеральный университет»

Институт цифрового развития

ОТЧЕТ ПО ДИСЦИПЛИНЕ
ОСНОВЫ ПРОГРАММНОЙ ИНЖЕНЕРИИ

Выполнил:

Турклиев Владимир

2 курс, группа ПИЖ-б-о-20-1

Принял:

Воронкин Роман Александрович

Ставрополь, 2021 г.

ВЫПОЛНЕНИЕ


1) Создание репозитория.

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner *

Repository name *


 vegas007gof ▾

 /


lab1 ✓

Great repository names are short and memorable. Need inspiration? How about [symmetrical-winner?](#)

Description (optional)

☒  **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.](#)

.gitignore template: C++ ▾

☒ Choose a license

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

License: Лицензия MIT ▾

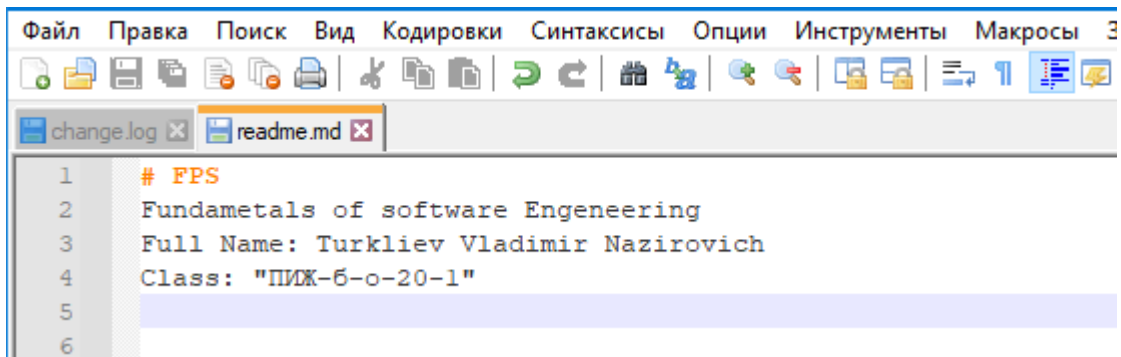
This will set  main as the default branch. Change the default name in your [settings](#).

Create repository

2) Клонирование репозитория на компьютер

```
C:\Users\Vova>cd desktop
C:\Users\Vova\Desktop>cd учеба
C:\Users\Vova\Desktop\учеба>cd основы ии
C:\Users\Vova\Desktop\учеба\основы ии>cd lab1
C:\Users\Vova\Desktop\учеба\основы ии\lab1>git clone https://github.com/vegas007gof/lab1.git
Cloning into 'lab1'...
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (4/4), done.
C:\Users\Vova\Desktop\учеба\основы ии\lab1>
```

3) Добавление информации в «readme.md»



The screenshot shows a code editor with a menu bar (Файл, Правка, Поиск, Вид, Кодировки, Синтаксисы, Опции, Инструменты, Макросы) and a toolbar. The 'readme.md' file is open, showing the following content:

```
1 # FPS
2 Fundamentals of software Engineering
3 Full Name: Turkliiev Vladimir Nazirovich
4 Class: "ПИЖ-б-о-20-1"
5
6
```

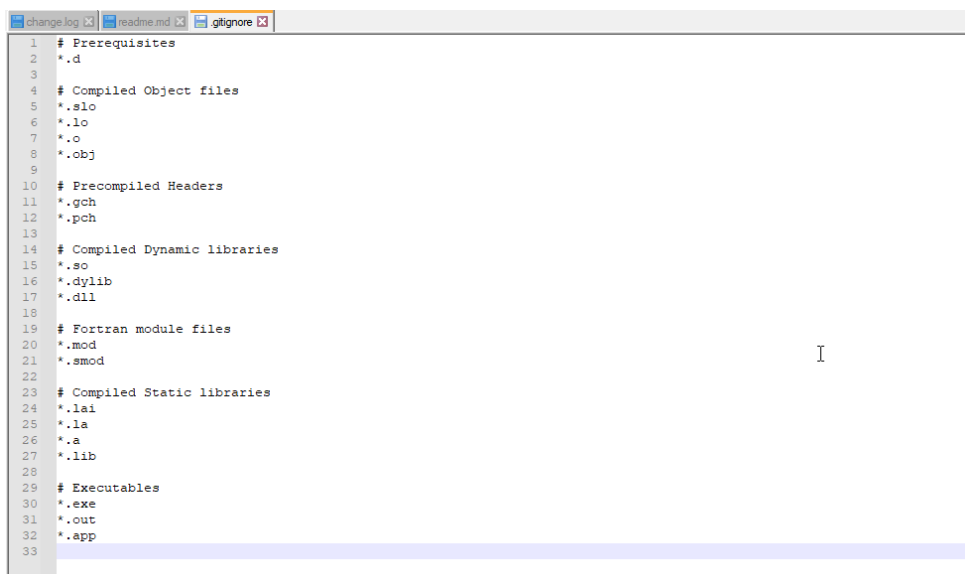
4) Делаем commit

```
C:\Users\Vova\Desktop\учеба\основы ии\lab1\lab1>git add README.md
C:\Users\Vova\Desktop\учеба\основы ии\lab1\lab1>git add readme.md
C:\Users\Vova\Desktop\учеба\основы ии\lab1\lab1> git commit -m "hope"
[main 569e46c] hope
 1 file changed, 4 insertions(+)

C:\Users\Vova\Desktop\учеба\основы ии\lab1\lab1> git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 347 bytes | 347.00 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/vegas007gof/lab1.git
 a30ba4f..569e46c main -> main

C:\Users\Vova\Desktop\учеба\основы ии\lab1\lab1>
```

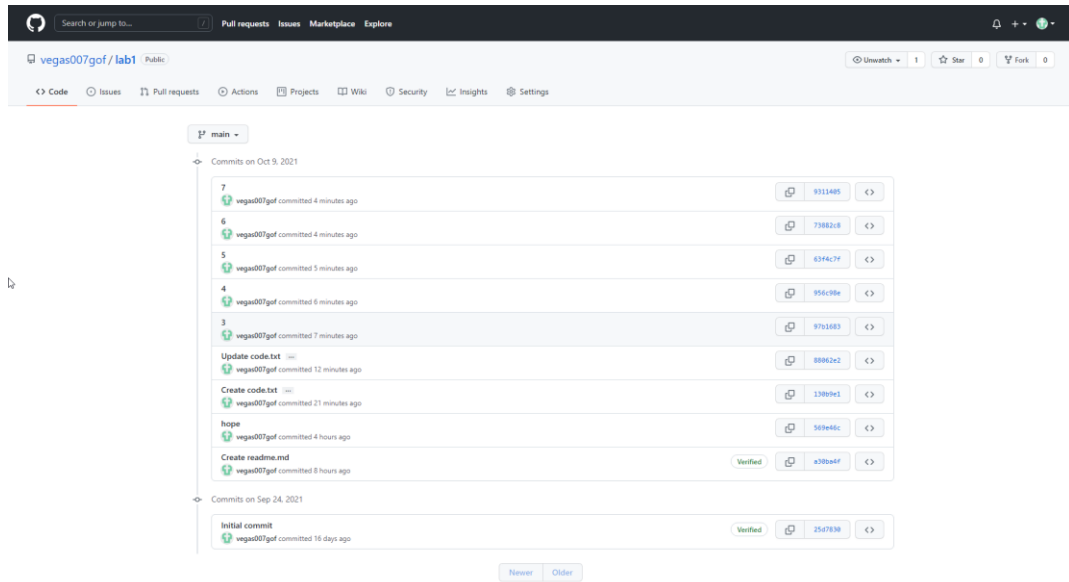
5) Файл .gitignore



The screenshot shows a code editor with a menu bar and a toolbar. The '.gitignore' file is open, showing the following content:

```
1 # Prerequisites
2 *.d
3
4 # Compiled Object files
5 *.slo
6 *.lo
7 *.o
8 *.obj
9
10 # Precompiled Headers
11 *.gch
12 *.pch
13
14 # Compiled Dynamic libraries
15 *.so
16 *.dylib
17 *.dll
18
19 # Fortran module files
20 *.mod
21 *.smod
22
23 # Compiled Static libraries
24 *.lai
25 *.la
26 *.a
27 *.lib
28
29 # Executables
30 *.exe
31 *.out
32 *.app
33
```

6) История коммитов



Ответы на контрольные вопросы:

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?

Проблемой локальной СКВ является основное свойство — локальность. Она совершенно не предназначена для коллективного использования.

Самый очевидный минус централизованной СКВ - это единая точка отказа, представленная централизованным сервером. Если этот сервер выйдет из строя на час, то в течение этого времени никто не сможет использовать контроль версий для сохранения изменений.

3. К какой СКВ относится Git?

Распределенная система контроля версий.

4. В чем концептуальное отличие Git от других СКВ?

Основное отличие Git от любой другой СКВ— это подход к работе со своими данными. Подход Git к хранению данных больше похож на набор снимков миниатюрной файловой системы. Каждый раз, когда вы делаете коммит, то

есть сохраняете состояние своего проекта в Git, система запоминает, как выглядит каждый файл в этот момент, и сохраняет ссылку на этот снимок.

5. Как обеспечивается целостность хранимых данных в Git?

В Git для всего вычисляется хеш-сумма, и только потом происходит сохранение. В дальнейшем обращение к сохранённым объектам происходит по этой хеш-сумме. Это значит, что невозможно изменить содержимое файла или директории так, чтобы Git не узнал об этом.

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 (committed), изменённое (modified) и подготовленное (staged).

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?

Профиль - это 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?

В репозитории можно хранить код, конфигурации, наборы данных, изображения и другие файлы, включенные в проект.

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?

Для этого необходимо перейти на официальный сайт Git <https://git-scm.com/> и скачать версию для операционной системы. После чего необходимо выполнить установку на свой компьютер. Чтобы убедиться, что Git был успешно установлен,

нужно ввести команду `git version` ниже в терминале, чтобы отобразить текущую версию Git.

11. Опишите этапы создания репозитория в GitHub.

Имя репозитория. Оно может быть любое, необязательно уникальное во всем github, потому что привязано к аккаунту, но уникальное в рамках тех репозиториях, которые создавали.

Описание (Description). Можно оставить пустым.

Public/private. Выбираем открытый (Public), НЕ ставим галочку “Initialize this repository with a README” (В README потом будет лежать какая-то основная информация, что же такое ваш проект и как с ним работать).

gitignore и LICENSE можно сейчас не выбирать.

После заполнения этих полей нажимаем кнопку Create repository.

12. Какие типы лицензий поддерживаются GitHub при создании репозитория?

Apache, GPL, MIT и другие

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

Для этого на странице репозитория необходимо найти кнопку Clone или Code и щелкнуть по ней, чтобы отобразить адрес репозитория для клонирования.

Откройте командную строку или терминал и перейдите в каталог, куда вы хотите скопировать хранилище. Затем напишите `git clone` и введите адрес. Это необходимо, чтобы иметь локальное хранилище.

14. Как проверить состояние локального репозитория Git?

Необходимо ввести команду `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` :

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`.

Необходимо клонировать (`git clone`) исходный репозиторий на каждый из компьютеров

Также на каждом компьютере добавить сразу все измененные файлы в версионный контроль посредством команды `git add`

Зафиксировать изменения с описательным комментарием (`git commit -m`)

Чтобы внести изменения, сделанные кем-то другим, использовать `git pull`

И отправить изменения на сервер (`git push`)

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

GitLab, SourceForge, BitBucket, Launchpad, Apache Allura, Cloud Source, AWS code commit, FogCreek/DevHub, BeanStalk, GitKraken.

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, GitKraken, SmartGit, SourceTree. GitHub Desktop

Рабочий стол GitHub в основном является расширением рабочего процесса GitHub. Можно входить в систему, используя свои учетные данные GitHub и начинать работу с репозиториями. Есть возможность создавать новые репозитории, добавлять локальные репозитории и выполнять большинство

операций Git из пользовательского интерфейса. GitHub Desktop является полностью открытым исходным кодом, и он доступен для MacOS и Windows.