

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 ПРАКТИЧЕСКОМУ ЗАНЯТИЮ №1**  
**дисциплины «Основы программной инженерии»**

Выполнил:  
Хетагуров Тамерлан Аланович  
2 курс, группа ПИЖ-б-о-22-1,  
09.03.04 «Программная инженерия»,  
направленность (профиль) «Разработка  
и сопровождение программного  
обеспечения», очная форма обучения

---

(подпись)

Руководитель практики:  
Богданов. С. С., ассистент кафедры  
инфокоммуникаций

---

(подпись)

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

Ставрополь, 2023 г.

## Create a new repository

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

Required fields are marked with an asterisk (\*).

Owner \* Tamikh / Repository name \* Program1  
Program1 is available.

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

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:

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

Add .gitignore

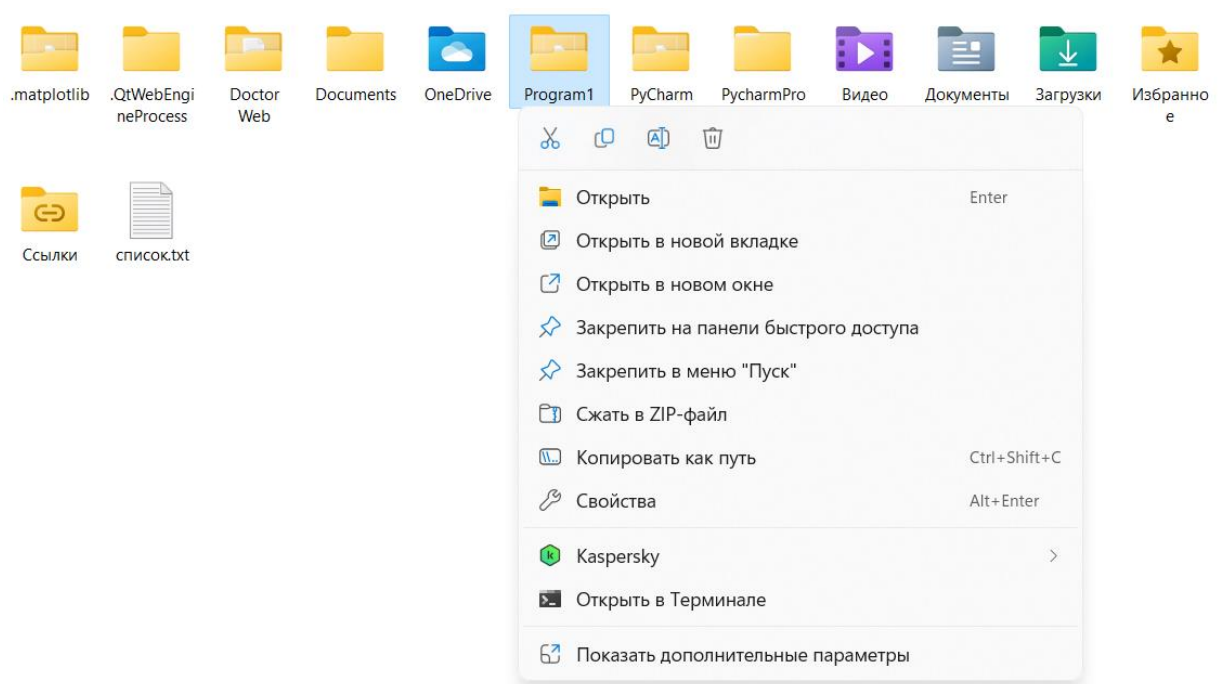
.gitignore template: Python

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

Choose a license

License: None

## Рисунок 1 – Создание общественного репозитория



## Рисунок 2 – Клонирование репозитория на компьютер

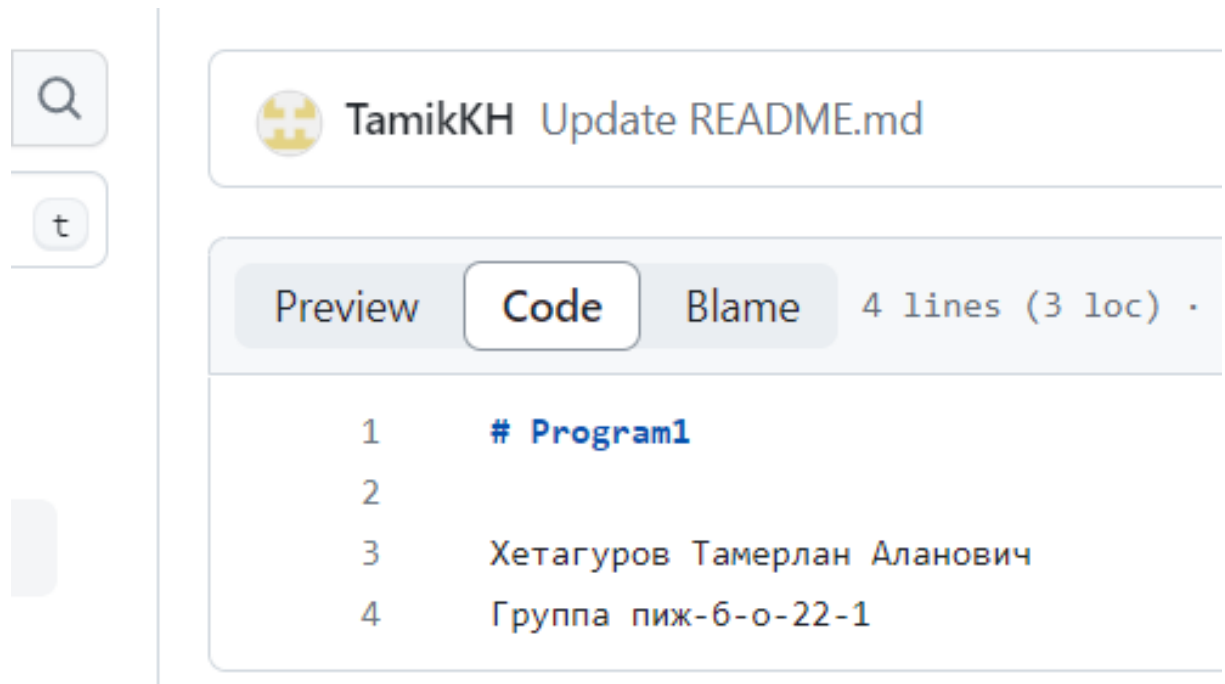


Рисунок 3 – Внесение личных данных в репозиторий

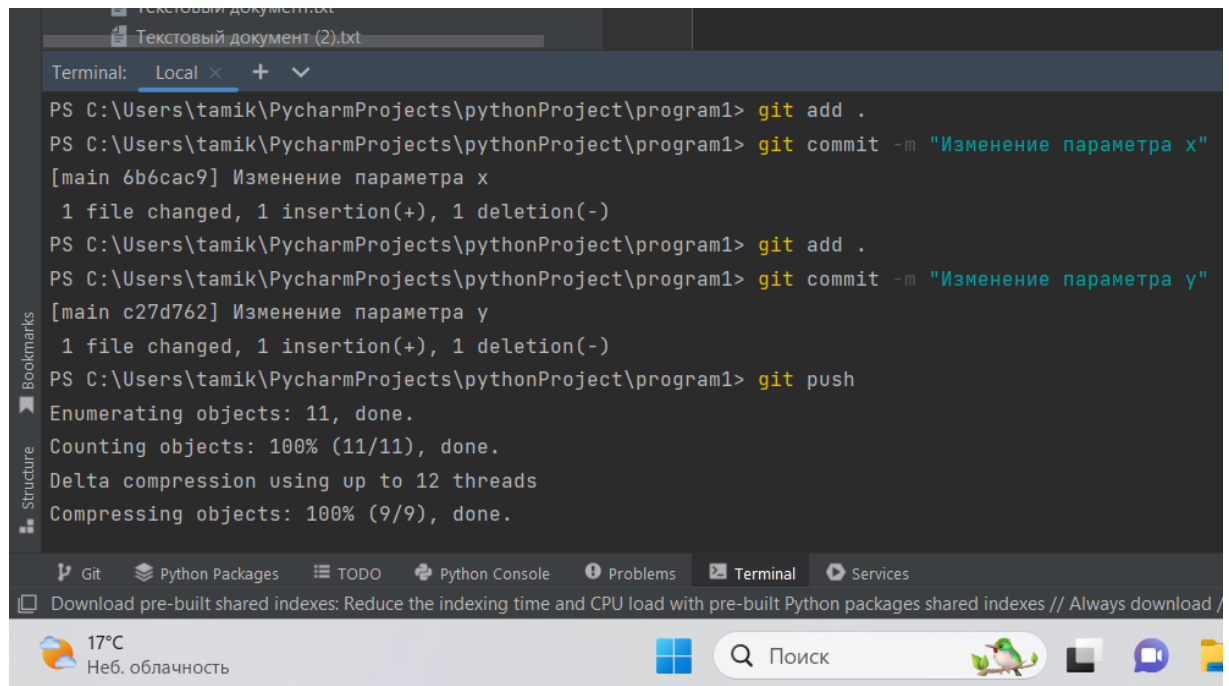
```
Terminal: Local x + v
fatal: not a git repository (or any of the parent directories): .git
fatal: not a git repository (or any of the parent directories): .git
PS C:\Users\tamik\PycharmProjects\pythonProject> git add .
fatal: not a git repository (or any of the parent directories): .git
PS C:\Users\tamik\PycharmProjects\pythonProject> git commit -m "Обновлено содержание программы"
fatal: not a git repository (or any of the parent directories): .git
PS C:\Users\tamik\PycharmProjects\pythonProject> cd Program1
git config --global user.email "you@example.com"
git config --global user.name "Your Name"

to set your account's default identity.
Omit --global to set the identity only in this repository.

fatal: unable to auto-detect email address (got 'tamik@LAPTOP-FF8K6N0J.(none)')
PS C:\Users\tamik\PycharmProjects\pythonProject\Program1> git config --global user.name TamikKH
PS C:\Users\tamik\PycharmProjects\pythonProject\Program1> git config --global user.email tamikhetag@gmail.com
PS C:\Users\tamik\PycharmProjects\pythonProject\Program1> git add .
PS C:\Users\tamik\PycharmProjects\pythonProject\Program1> git commit -m "Обновлено содержание программы"
[main c3e2ad1] Обновлено содержание программы
1 file changed, 10 insertions(+)
PS C:\Users\tamik\PycharmProjects\pythonProject\Program1> git commit -m "Изменение параметра T"
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
(use "git push" to publish your local commits)

Changes not staged for commit:
```

Рисунок 4 – Создание коммитов



```
Terminal: Local x + v
PS C:\Users\tamik\PycharmProjects\pythonProject\program1> git add .
PS C:\Users\tamik\PycharmProjects\pythonProject\program1> git commit -m "Изменение параметра x"
[main 6b6cac9] Изменение параметра x
1 file changed, 1 insertion(+), 1 deletion(-)
PS C:\Users\tamik\PycharmProjects\pythonProject\program1> git add .
PS C:\Users\tamik\PycharmProjects\pythonProject\program1> git commit -m "Изменение параметра y"
[main c27d762] Изменение параметра y
1 file changed, 1 insertion(+), 1 deletion(-)
PS C:\Users\tamik\PycharmProjects\pythonProject\program1> git push
Enumerating objects: 11, done.
Counting objects: 100% (11/11), done.
Delta compression using up to 12 threads
Compressing objects: 100% (9/9), done.
```

Рисунок 5 – Сохранение коммитов в репозиторий

Ответы на контрольные 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 локальных СКВ в том, что он невероятно сильно подвержен появлению ошибок. Можно легко забыть, в какой директории вы находитесь, и случайно изменить не тот файл или скопировать не те файлы, которые вы хотели.

Самый очевидный минус ЦСКВ — 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, над которыми работает, а также никто не сможет обмениваться этими изменениями с другими разработчиками.

3. К какой СКВ относится Git?

Распределенные СКВ.

4. В чем концептуальное отличие Git от других СКВ?

Простое ветвление. В других СКВ создание веток— утомительная и трудоёмкая задача, так как весь код копируется в новую ветку. В Git управление ветками реализовано гораздо проще и эффективнее.

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 ваши файлы: *изменён* (modified), *индексирован* (staged) и *зафиксирован* (committed):

7. Что такое профиль пользователя в GitHub?

На странице пользователя отображаются сведения о работе через репозитории, вклад, который был внесен, и беседы.

8. Какие бывают репозитории в GitHub?

Репозиторий Git бывает локальный и удалённый. Локальный репозиторий — это подкаталог .git, создаётся (в пустом виде) командой git init и (в непустом виде с немедленным копированием содержимого родительского удалённого репозитория и простановкой ссылки на родителя) командой git clone.

9. Укажите основные этапы модели работы с GitHub.

GitHub - это платформа для размещения кода. Иными словами, это место, где разработчики могут хранить свои проекты и работать вместе. Таким образом контролировать версии программ и сотрудничать становится гораздо проще. GitHub основан на популярной системе управления версиями под названием Git и предоставляет некоторые дополнительные функции,

такие как веб-интерфейс, инструменты совместной работы, средство отслеживания ошибок, статистика проекта и многое другое.

#### 10. Как осуществляется первоначальная настройка Git после установки?

Чтобы убедиться, что Git был успешно установлен, введите команду ниже в терминале, чтобы отобразить текущую версию вашего Git:7

Если она сработала, давайте добавим в настройки Git ваше имя, фамилию и адрес электронной почты, связанный с вашей учетной записью GitHub:

#### 11. Опишите этапы создания репозитория в GitHub.

В правом верхнем углу, рядом с аватаром есть кнопка с плюсиком, нажимая которую мы переходим к созданию нового репозитория. В результате будет выполнен переход на страницу создания репозитория, на ней будут поля после заполнения которых, нажимаем кнопку Create repository. Отлично, ваш репозиторий готов!

#### 12. Какие типы лицензий поддерживаются GitHub при создании репозитория?

В нашем случае используется MIT License.

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

После создания репозитория его необходимо клонировать на ваш компьютер. 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.

#### 14. Как проверить состояние локального репозитория Git?

Использовать 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` и отправки изменений на сервер с помощью команды `git push` ?

На локальном репозитории создается или изменяется файл. С помощью команды `git commit` происходит сохранение изменений. С помощью команды `git push` изменения отправляются в удаленный репозиторий.

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.

Использовать команду `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

Bitbucket поддерживает функцию `pull` запроса, которая помогает загрузить проект из платформы. GitHub также поддерживает функцию `pull` запроса и помогает пользователю получить проект с платформы. В GitLab такая функция `pull` запроса отсутствует, и вместо нее в платформе GitLab поддерживается `merge` запрос.

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.

**SmartGit** также является кроссплатформенным, мощным, популярным **Git**-клиентом с графическим интерфейсом для **Linux**, **Mac OS X** и **Windows**. Он называется **Git** для профессионалов. Он позволяет пользователям справляться с ежедневными задачами **Git** и повышает их производительность за счет эффективных рабочих процессов.