

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

Аллаёров Жамшид Хасан угли

---

(подпись)

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

инфокоммуникаций

Воронкин Р. А.

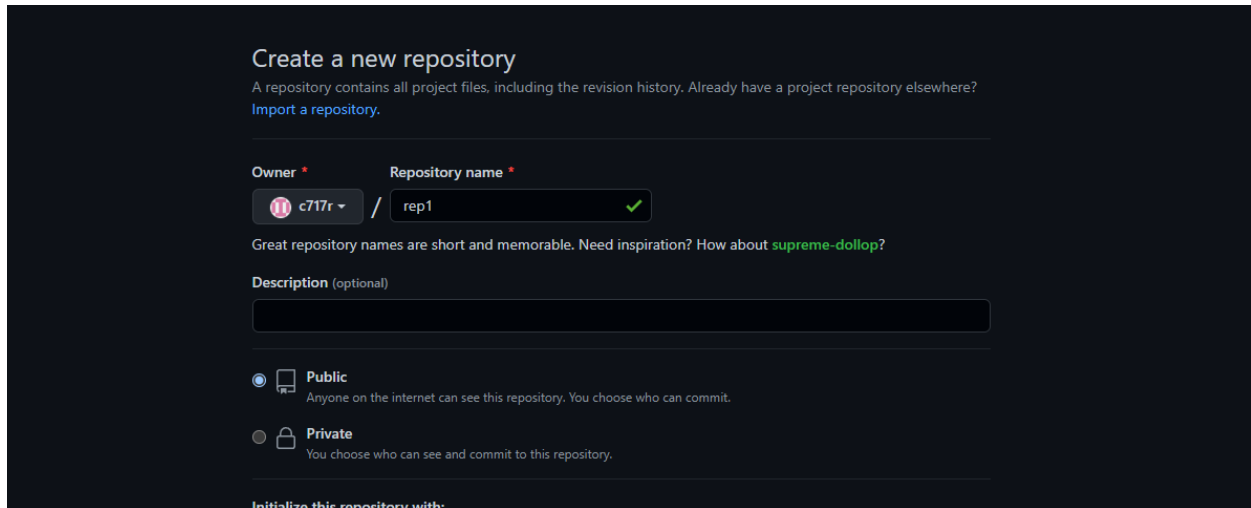
Работа защищена с оценкой:

---

(подпись)

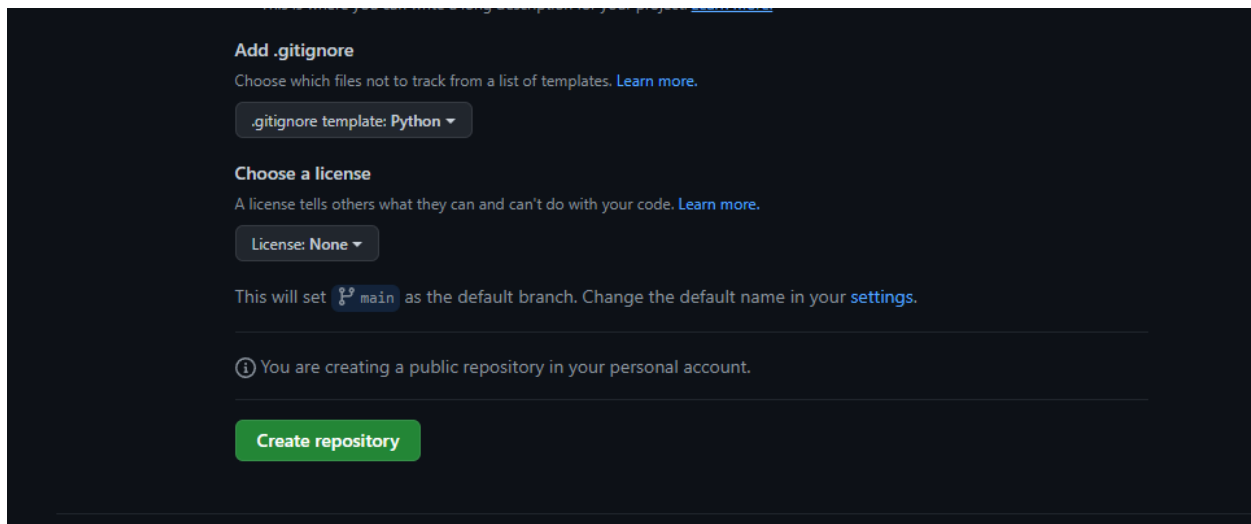
Ставрополь, 2022

Создать общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и выбранный Вами язык программирования (выбор языка программирования будет доступен после установки флажка **Add .gitignore**)



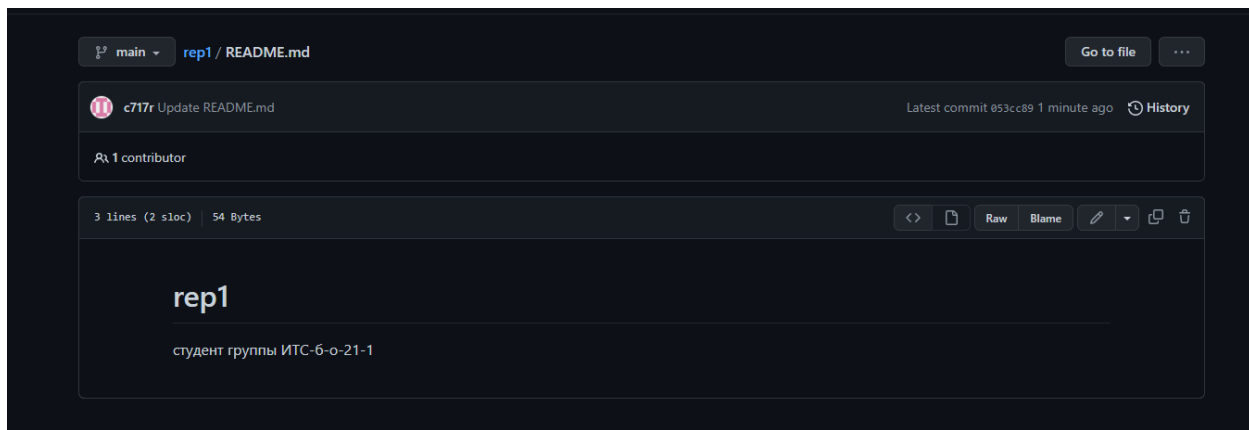
The screenshot shows the 'Create a new repository' page on GitHub. The title is 'Create a new repository'. Below it, a subtitle says 'A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)'. The form has two main sections: 'Owner' and 'Repository name'. The 'Owner' dropdown is set to 'c717r'. The 'Repository name' dropdown is set to 'rep1' with a green checkmark. Below these, there is a hint: 'Great repository names are short and memorable. Need inspiration? How about [supreme-dollop?](#)'. There is a 'Description (optional)' text input field. Below the description, there are two radio button options for visibility: 'Public' (selected) and 'Private'. The 'Public' option has a subtext: 'Anyone on the internet can see this repository. You choose who can commit.' The 'Private' option has a subtext: 'You choose who can see and commit to this repository.' At the bottom, there is a section 'Initialize this repository with:'.

Дополните файл .gitignore необходимыми правилами для выбранного языка программирования и интегрированной среды разработки.

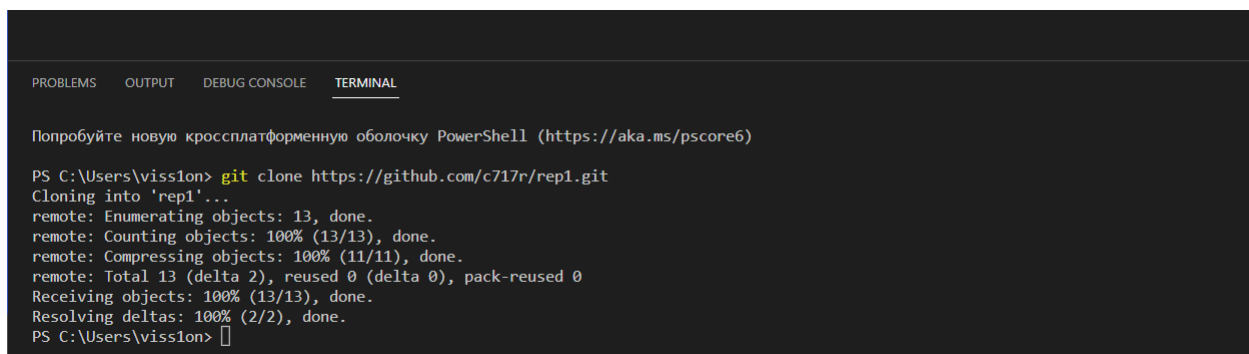


The screenshot shows the 'Add .gitignore' and 'Choose a license' section of the GitHub repository creation process. The title is 'Add .gitignore'. Below it, a subtitle says 'Choose which files not to track from a list of templates. [Learn more.](#)'. There is a dropdown menu for '.gitignore template:' set to 'Python'. Below this, there is a section 'Choose a license'. A subtitle says 'A license tells others what they can and can't do with your code. [Learn more.](#)'. There is a dropdown menu for 'License:' set to 'None'. Below the license section, there is a note: 'This will set `main` as the default branch. Change the default name in your [settings](#).' At the bottom, there is an information icon and a note: 'You are creating a public repository in your personal account.' A green 'Create repository' button is at the bottom.

Добавьте в файл README.md информацию о группе и ФИО студента, выполняющего лабораторную работу.



Выполните клонирование созданного репозитория на рабочий компьютер.



Напишите небольшую программу на выбранном Вами языке программирования.

Фиксируйте изменения при написании программы в локальном репозитории.

Должно быть

сделано не менее 7 коммитов.

```
python.py
python.py
1  print("Hello world!")
2  print(3 + 4)
3  print(3 * 5)
4  print(3 ** 2)
5

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  JUPYTER

Windows PowerShell
(C) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

Попробуйте новую кроссплатформенную оболочку PowerShell (https://aka.ms/pscore6)

PS C:\Users\viiss1on\Desktop\rep1> & C:/Users/viiss1on/AppData/Local/Programs/Python/Python310/python.exe c:/Users/viiss1on/Desktop/rep1/python.py
Hello world!
7
15
9
PS C:\Users\viiss1on\Desktop\rep1>
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  JUPYTER

Run

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 'viiss1on@DESKTOP-NE3NPQ2.(none)')
PS C:\Users\viiss1on\Desktop\rep1> git config --global user.email "jallayorov39@gmail.com"
PS C:\Users\viiss1on\Desktop\rep1> git config --global user.name "Jamshid"
PS C:\Users\viiss1on\Desktop\rep1>
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  JUPYTER

PS C:\Users\viiss1on\Desktop\rep1> git commit --m "1"
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
PS C:\Users\viiss1on\Desktop\rep1> git add .
PS C:\Users\viiss1on\Desktop\rep1> git commit --m "2"
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
PS C:\Users\viiss1on\Desktop\rep1>
```

```
EXPLORER  ...  python.py
✓ REP1
  .gitignore
  python.py
  README.md

python.py
1  print("Hello world!")
2  print(3 + 4)
3  print(3 * 5)
4  print(3 ** 2)
5
6  print(3 / 4)
7
8

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  JUPYTER

On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
PS C:\Users\viiss1on\Desktop\rep1> git add .
PS C:\Users\viiss1on\Desktop\rep1> git commit --m "2"
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
PS C:\Users\viiss1on\Desktop\rep1> git push
□
```

1. Что такое СКВ и каково ее назначение?

О системе контроля версий

Что такое «система контроля версий» и почему это важно? Система контроля версий — это система, записывающая изменения в файл или набор файлов в течение времени и позволяющая вернуться позже к определённой версии. Для контроля версий файлов в этой книге в качестве примера будет использоваться исходный код программного обеспечения, хотя на самом деле вы можете использовать контроль версий практически для любых типов файлов.

## 2. В чем недостатки локальных и централизованных СКВ?

серьёзная проблема, с которой сталкиваются люди, — это необходимость взаимодействовать с другими разработчиками. Для того, чтобы разобраться с ней, были разработаны централизованные системы контроля версий (ЦСКВ). Такие системы, как CVS, Subversion и Perforce, используют единственный сервер, содержащий все версии файлов, и некоторое количество клиентов, которые получают файлы из этого централизованного хранилища. Применение ЦСКВ являлось стандартом на протяжении многих лет.

## 3. К какой СКВ относится Git?

### **Распределённые системы контроля версий**

Здесь в игру вступают распределённые системы контроля версий (РСКВ). В РСКВ (таких как Git, Mercurial, Bazaar или Darcs) клиенты не просто скачивают снимок всех файлов (состояние файлов на определённый момент времени) — они полностью копируют репозиторий. В этом случае, если один из серверов, через который разработчики обменивались данными, умрёт, любой клиентский репозиторий может быть скопирован на другой сервер для продолжения работы. Каждая копия репозитория является полным бэкапом всех данных.

## 4. В чем концептуальное отличие Git от других СКВ?

Многие люди в качестве метода контроля версий применяют копирование файлов в отдельный каталог (возможно даже, каталог с отметкой по времени, если они достаточно сообразительны). Данный подход очень распространён из-за его простоты, однако он невероятно сильно подвержен появлению ошибок. Можно легко забыть в каком каталоге вы находитесь и случайно изменить не тот файл или скопировать не те файлы, которые вы хотели.

Для того, чтобы решить эту проблему, программисты давным-давно разработали локальные СКВ с простой базой данных, которая хранит записи о всех изменениях в файлах, осуществляя тем самым контроль ревизий.

#### 5. Как обеспечивается целостность хранимых данных в Git?

Контрольные суммы в Git На самом деле контрольная сумма используется в функции идентификатора коммита и обычно называется «SHA». Контрольные суммы, сгенерированные Git, включают метаданные о коммите, такие как автор, дата и SHA коммита, который был до него. Git обеспечивает целостность хранимых данных, используя контрольные суммы в качестве идентификаторов.

#### 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).

Если вы добавляете (с помощью `git add`) «Не отслеживаемый» файл, то он переходит в состояние «Подготовлено».

Если вы изменяете файл в состоянии «Не изменено», то он переходит в состояние «Изменено». Если вы сохраняете изменённый файл (то есть находящийся в состоянии «Изменено») он переходит в состояние «Подготовлено». Если вы делаете коммит файла (то есть находящийся в состоянии «Подготовлено») он переходит в состояние «Не изменено».

Если версии файла в HEAD и рабочей директории отличаются, то файл будет находиться в состоянии «Изменено», иначе (если версия в HEAD и в рабочем каталоге одинакова") файл будет находиться в состоянии «Не изменено».

#### 7. Что такое профиль пользователя в GitHub?

Убедитесь, что email для связи с вами доступен всем. Для этого зайдите в настройки профиля: там есть специальная кнопка «сделать публичным». Избегайте адресов в духе `sexubaby@mail.ru` — используйте ящик в доменной зоне `.com` с понятным неймингом. Не делайте публичным email, который привязан к вашему

аккаунту GitHub: это повышает вероятность взлома аккаунта и демонстрирует слабое владение правилами безопасности в интернете.

### Биография

Многие игнорируют этот раздел, а зря. Напишите немного о том, какими технологиями вы занимаетесь и что вам интересно. Тогда тот, кто просматривает ваш код, будет понимать, на что смотреть.

### Ссылка на соцсеть

Добавьте ссылку на профессиональную соцсеть, которую вы ведёте наиболее активно и подробно.

## 8. Какие бывают репозитории в GitHub?

Какие бывают репозитории Github  
инструменты тестирования;  
фрагменты кода;  
советы;  
концепции программирования;  
базы знаний;  
примеры;  
справочники;  
руководства;+

## 9. Укажите основные этапы модели работы с GitHub.

### Первоначальная настройка Git

Теперь, когда Git установлен в вашей системе, самое время настроить среду для работы с Git под себя. Это нужно сделать только один раз — при обновлении версии Git настройки сохранятся. Но, при необходимости, вы можете поменять их в любой момент, выполнив те же команды снова.

В состав Git входит утилита `git config`, которая позволяет просматривать и настраивать параметры, контролирующие все аспекты работы Git, а также его внешний вид. Эти параметры могут быть сохранены в трёх местах:



Файл `[path]/etc/gitconfig` содержит значения, общие для всех пользователей системы и для всех их репозиториев. Если при запуске `git config` указать параметр `-system`, то параметры будут читаться и сохраняться именно в этот файл. Так как этот файл является системным, то вам потребуются права суперпользователя для внесения изменений в него.

Файл `~/.gitconfig` или `~/.config/git/config` хранит настройки конкретного пользователя. Этот файл используется при указании параметра `--global` и применяется ко всем репозиториям, с которыми вы работаете в текущей системе.

Файл `config` в каталоге Git (т. е. `.git/config`) репозитория, который вы используете в данный момент, хранит настройки конкретного репозитория. Вы можете заставить Git читать и писать в этот файл с помощью параметра `--local`, но на самом деле это значение по умолчанию. Неудивительно, что вам нужно находиться где-то в репозитории Git, чтобы эта опция работала правильно.

Настройки на каждом следующем уровне подменяют настройки из предыдущих уровней, то есть значения в `.git/config` перекрывают соответствующие значения в `[path]/etc/gitconfig`.

В системах семейства Windows Git ищет файл `.gitconfig` в каталоге `$HOME` (`C:\Users\%USER` для большинства пользователей). Кроме того, Git ищет файл `[path]/etc/gitconfig`, но уже относительно корневого каталога MSys, который находится там, куда вы решили установить Git при запуске инсталлятора.

10. Как осуществляется первоначальная настройка Git после установки?

### Первоначальная настройка Git

Теперь, когда Git установлен в вашей системе, самое время настроить среду для работы с Git под себя. Это нужно сделать только один раз — при обновлении версии Git настройки сохраняются. Но, при необходимости, вы можете поменять их в любой момент, выполнив те же команды снова.

В состав Git входит утилита `git config`, которая позволяет просматривать и настраивать параметры, контролирующие все аспекты работы Git, а также его внешний вид. Эти параметры могут быть сохранены в трёх местах:

1. Файл `[path]/etc/gitconfig` содержит значения, общие для всех пользователей системы и для всех их репозиториях. Если при запуске `git config` указать параметр `--system`, то параметры будут читаться и сохраняться именно в этот файл. Так как этот файл является системным, то вам потребуются права суперпользователя для внесения изменений в него.
2. Файл `~/.gitconfig` или `~/.config/git/config` хранит настройки конкретного пользователя. Этот файл используется при указании параметра `--global` и применяется ко **всем** репозиториям, с которыми вы работаете в текущей системе.
3. Файл `config` в каталоге Git (т. е. `.git/config`) репозитория, который вы используете в данный момент, хранит настройки конкретного репозитория. Вы можете заставить Git читать и писать в этот файл с помощью параметра `--local`, но на самом деле это значение по умолчанию. Неудивительно, что вам нужно находиться где-то в репозитории Git, чтобы эта опция работала правильно.

Настройки на каждом следующем уровне подменяют настройки из предыдущих уровней, то есть значения в `.git/config` перекрывают соответствующие значения в `[path]/etc/gitconfig`.

В системах семейства Windows Git ищет файл `.gitconfig` в каталоге `$HOME` (`C:\Users\%USER` для большинства пользователей). Кроме того, Git ищет файл `[path]/etc/gitconfig`, но уже относительно корневого каталога MSys, который находится там, куда вы решили установить Git при запуске инсталлятора.