

Отчет по лабораторной работе №2

Управление версиями

Рябцев Илья Витальевич

НКАбд-03-22

1032225675

1 Цель работы

– Изучить идеологию и применение средств контроля версий. – Освоить умения по работе с git.

2 Задание

1. Зарегистрироваться на Github;
2. Создать базовую конфигурацию для работы с git;
3. Создать ключ SSH;
4. Создать ключ PGP;
5. Настроить подписи git;
6. Создать локальный каталог для выполнения заданий по предмету.

3 Теоретическое введение

В этой лабораторной работе мы познакомимся с системами контроля версий. Системы контроля версий (Version Control System, VCS) применяются при работе нескольких человек над одним проектом. Обычно основное дерево проекта хранится в локальном или удалённом репозитории, к которому настроен доступ для участников проекта. При внесении изменений в содержание проекта система контроля версий позволяет их фиксировать, совмещать изменения, произведённые разными участниками проекта, производить откат к любой более ранней версии проекта, если это требуется. Существуют классические и распределённые системы контроля версий (РСКВ). Сегодня мы будем работать с распределённой VSC – Git. В РСКВ (таких как Git, Mercurial, Bazaar или Darcs) клиенты не просто скачивают снимок всех файлов - 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. Более того, многие РСКВ могут одновременно взаимодействовать с несколькими удалёнными репозиториями, благодаря этому вы можете работать с различными группами людей, применяя различные подходы одновременно в рамках одного проекта. Это позволяет применять сразу несколько подходов в разработке, например,

иерархические модели, что совершенно невозможно в централизованных системах. [1]

4 Выполнение лабораторной работы

№1 Создаем учетную запись на Github и заполняем основные данные. (рис. 1)

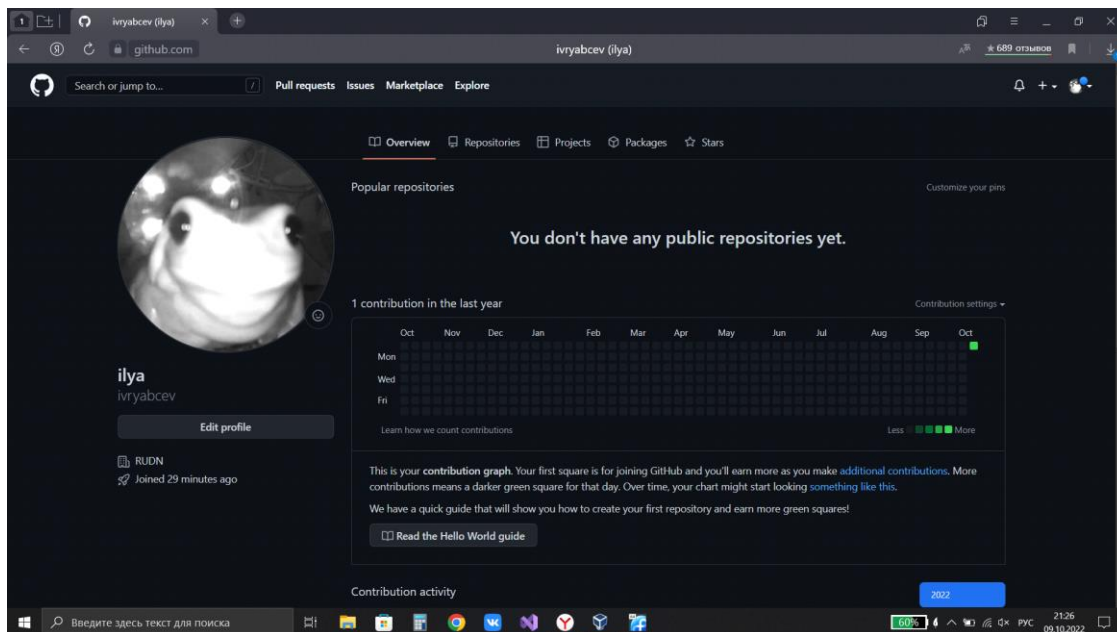


Рис. 1: Создание учетной записи на GitHub

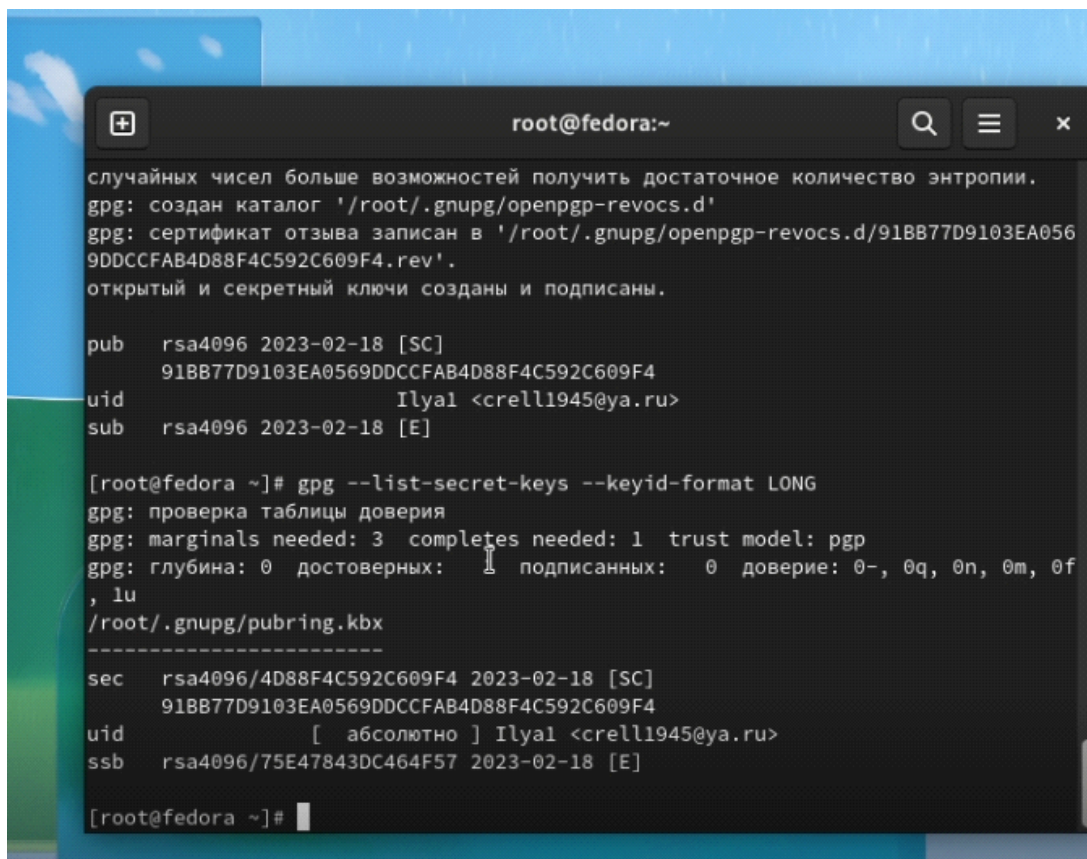
№2 Далее установим программное обеспечение git-flow в Fedora Linux

Перейдем к базовой настройке Git: зададим имя и почту владельца репозитория; настроим utf-8 в выводе сообщений git; настроим верификацию и подписание коммитов git (Зададим имя начальной ветки (будем называть её master), параметр autocrlf, параметр safecrlf).

№3 Создаем ключ ssh: по алгоритму rsa с ключём размером 4096 бит: (рис. 5)

```
root@fedora:~  
|. |  
|.o |  
|. o. |  
| +o S |  
|*.O... + + + |  
|=+ o.EB = + o |  
|+o. =*.^ . . |  
|.oo..*%+0 |  
+-----[SHA256]-----+  
[root@fedora ~]# gpg --full-generate-key  
gpg (GnuPG) 2.3.7; Copyright (C) 2021 Free Software Foundation, Inc.  
This is free software: you are free to change and redistribute it.  
There is NO WARRANTY, to the extent permitted by law.  
  
Выберите тип ключа:  
  (1) RSA and RSA  
  (2) DSA and Elgamal  
  (3) DSA (sign only)  
  (4) RSA (sign only)  
  (9) ECC (sign and encrypt) *default*  
 (10) ECC (только для подписи)  
 (14) Existing key from card  
Ваш выбор? 1
```

Рис. 5: Создания ключа ssh по алгоритму rsa с ключем размером 4096
по алгоритму ed25519: (рис. 6)



```
root@fedora:~
случайных чисел больше возможностей получить достаточное количество энтропии.
gpg: создан каталог '/root/.gnupg/openpgp-revocs.d'
gpg: сертификат отзыва записан в '/root/.gnupg/openpgp-revocs.d/91BB77D9103EA056
9DDCCFAB4D88F4C592C609F4.rev'.
открытый и секретный ключи созданы и подписаны.

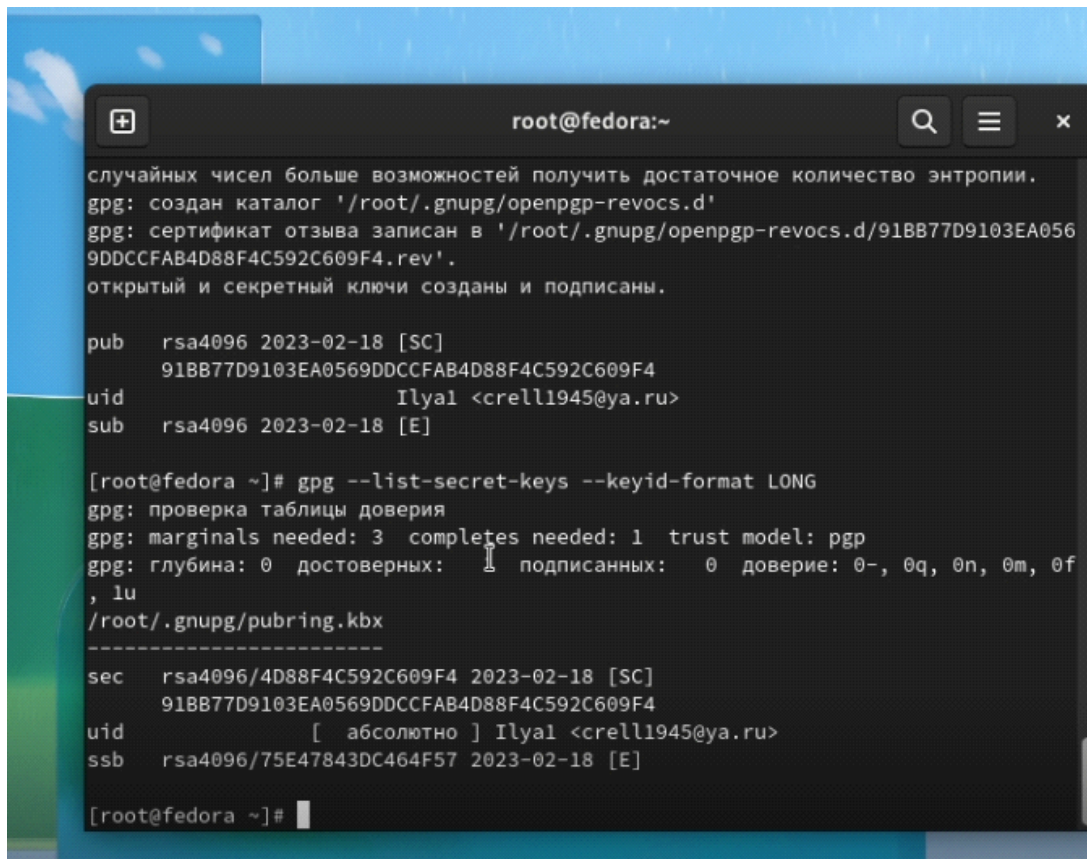
pub  rsa4096 2023-02-18 [SC]
    91BB77D9103EA0569DDCCFAB4D88F4C592C609F4
uid                               Ilya1 <crell1945@ya.ru>
sub   rsa4096 2023-02-18 [E]

[root@fedora ~]# gpg --list-secret-keys --keyid-format LONG
gpg: проверка таблицы доверия
gpg: marginals needed: 3  completes needed: 1  trust model: pgp
gpg: глубина: 0  достоверных: 1  подписанных: 0  доверие: 0-, 0q, 0n, 0m, 0f
, 1u
/root/.gnupg/pubring.kbx
-----
sec  rsa4096/4D88F4C592C609F4 2023-02-18 [SC]
    91BB77D9103EA0569DDCCFAB4D88F4C592C609F4
uid      [ абсолютно ] Ilya1 <crell1945@ya.ru>
ssb  rsa4096/75E47843DC464F57 2023-02-18 [E]

[root@fedora ~]#
```

Рис. 6: Создания ключа ssh по алгоритму ed25519

№4 Создаем ключ gpg. Генерируем ключ и из предложенных опций выбираем: • Тип RSA and RSA; • Размер 4096; • Выберите срок действия; значение по умолчанию — 0 (срок действия не истекает никогда). • GPG запросит личную информацию, которая сохранится в ключе: • Имя (не менее 5 символов). • Адрес электронной почты. • При вводе email убедитесь, что он соответствует адресу, используемому на GitHub. • Комментарий. Можно ввести что угодно или нажать клавишу ввода, чтобы оставить это поле пустым. (рис. 7)

A terminal window titled 'root@fedora:~' with search, menu, and close icons. It displays the output of GPG commands. The first part shows the creation of a key pair with a 4096-bit RSA public key and a 4096-bit RSA secret key. The second part shows the output of 'gpg --list-secret-keys --keyid-format LONG', which lists the secret key and the public key. The terminal text is as follows:

```
случайных чисел больше возможностей получить достаточное количество энтропии.
gpg: создан каталог '/root/.gnupg/openpgp-revocs.d'
gpg: сертификат отзыва записан в '/root/.gnupg/openpgp-revocs.d/91BB77D9103EA056
9DDCCFAB4D88F4C592C609F4.rev'.
открытый и секретный ключи созданы и подписаны.

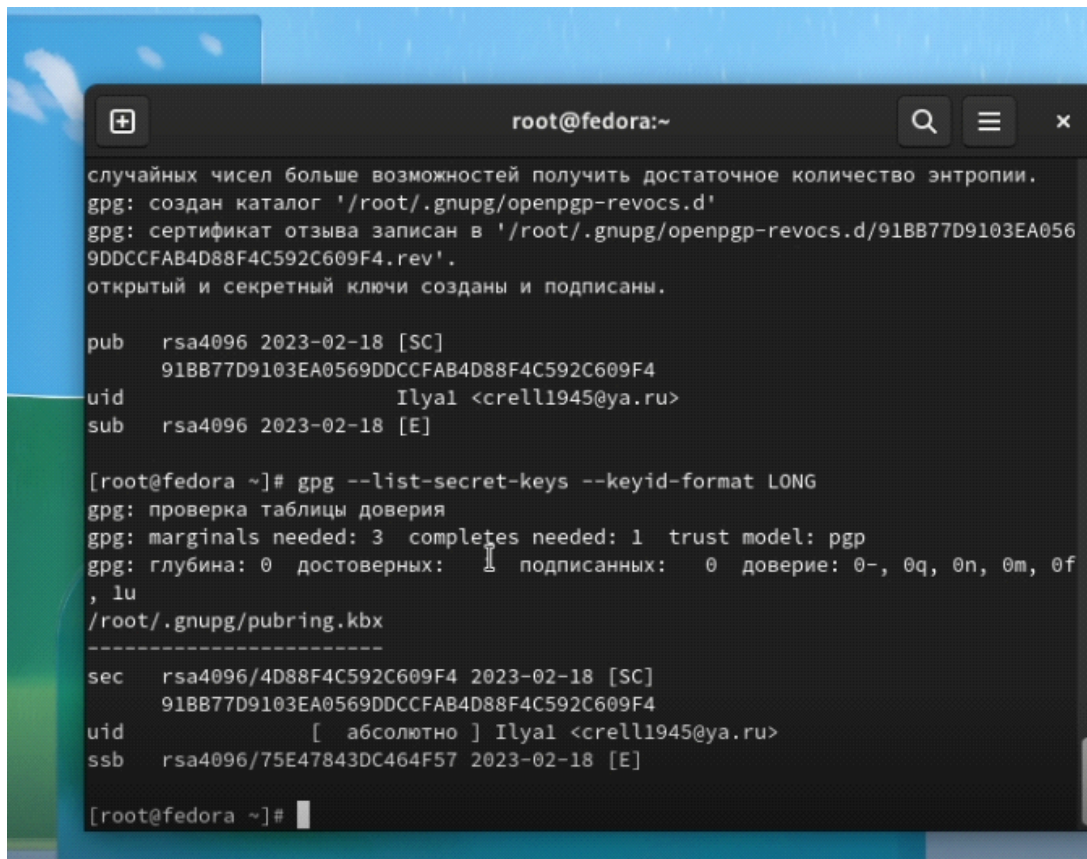
pub   rsa4096 2023-02-18 [SC]
      91BB77D9103EA0569DDCCFAB4D88F4C592C609F4
uid           Ilya1 <crell1945@ya.ru>
sub   rsa4096 2023-02-18 [E]

[root@fedora ~]# gpg --list-secret-keys --keyid-format LONG
gpg: проверка таблицы доверия
gpg: marginals needed: 3  completes needed: 1  trust model: pgp
gpg: глубина: 0  достоверных: 1  подписанных: 0  доверие: 0-, 0q, 0n, 0m, 0f
, 1u
/root/.gnupg/pubring.kbx
-----
sec   rsa4096/4D88F4C592C609F4 2023-02-18 [SC]
      91BB77D9103EA0569DDCCFAB4D88F4C592C609F4
uid           [ абсолютно ] Ilya1 <crell1945@ya.ru>
ssb   rsa4096/75E47843DC464F57 2023-02-18 [E]

[root@fedora ~]#
```

Рис. 7: Создание ключа gpg

Добавим pgp ключ в GitHub. Выводим список ключей и копируем отпечаток приватного ключа. (рис. 8)

A terminal window titled 'root@fedora:~' with search, menu, and close icons. It shows the output of GPG commands: key generation, listing secret keys, and checking the trust database. The key generation output shows a public key (pub), user ID (uid), and subkey (sub). The listing command shows a secret key (sec), user ID (uid), and subkey (ssb). The trust database check shows the number of marginal and complete keys needed, the trust model, and the depth of the trust database.

```
случайных чисел больше возможностей получить достаточное количество энтропии.
gpg: создан каталог '/root/.gnupg/openpgp-revocs.d'
gpg: сертификат отзыва записан в '/root/.gnupg/openpgp-revocs.d/91BB77D9103EA056
9DDCCFAB4D88F4C592C609F4.rev'.
открытый и секретный ключи созданы и подписаны.

pub  rsa4096 2023-02-18 [SC]
    91BB77D9103EA0569DDCCFAB4D88F4C592C609F4
uid                               Ilya1 <crell1945@ya.ru>
sub  rsa4096 2023-02-18 [E]

[root@fedora ~]# gpg --list-secret-keys --keyid-format LONG
gpg: проверка таблицы доверия
gpg: marginals needed: 3  completes needed: 1  trust model: pgp
gpg: глубина: 0  достоверных: 1  подписанных: 0  доверие: 0-, 0q, 0n, 0m, 0f
, 1u
/root/.gnupg/pubring.kbx
-----
sec  rsa4096/4D88F4C592C609F4 2023-02-18 [SC]
    91BB77D9103EA0569DDCCFAB4D88F4C592C609F4
uid                               [ абсолютно ] Ilya1 <crell1945@ya.ru>
ssb  rsa4096/75E47843DC464F57 2023-02-18 [E]

[root@fedora ~]#
```

Рис. 8: Вывод списка ключей

Скопируем сгенерированный PGP ключ в буфер обмена. (рис. 9)


```
root@fedora:~  
gpg: глубина: 0 доверенных: 1 подписанных: 0 доверие: 0-, 0q, 0n, 0m, 0f  
, 1u  
/root/.gnupg/pubring.kbx  
-----  
sec  rsa4096/4D88F4C592C609F4 2023-02-18 [SC]  
91BB77D9103EA0569DDCCFAB4D88F4C592C609F4  
uid  [ абсолютно ] Ilya1 <crell1945@ya.ru>  
ssb  rsa4096/75E47843DC464F57 2023-02-18 [E]  
  
[root@fedora ~]# gpg --armor --export 4D88F4C592C609F4 | xclip -sel clip  
[root@fedora ~]# git config --global user.signingkey 4D88F4C592C609F4  
[root@fedora ~]# git config --global commit.gpgsign true  
[root@fedora ~]# git config --global gpg.program $(which gpg2)  
  
[root@fedora ~]# gh auth login  
? What account do you want to log into? GitHub.com  
? What is your preferred protocol for Git operations? SSH  
? Upload your SSH public key to your GitHub account? /root/.ssh/id_rsa.pub  
? Title for your SSH key: GitHub CLI  
? How would you like to authenticate GitHub CLI? [Use arrows to move, type to filter]  
> Login with a web browser  
Paste an authentication token
```

Рис. 9: Копирование сгенерированного PGP ключа в буфер обмена

Перейдем в настройки GitHub (<https://github.com/settings/keys>), нажмем на кнопку New GPG key и вставим полученный ключ в поле ввода. (рис. 10)

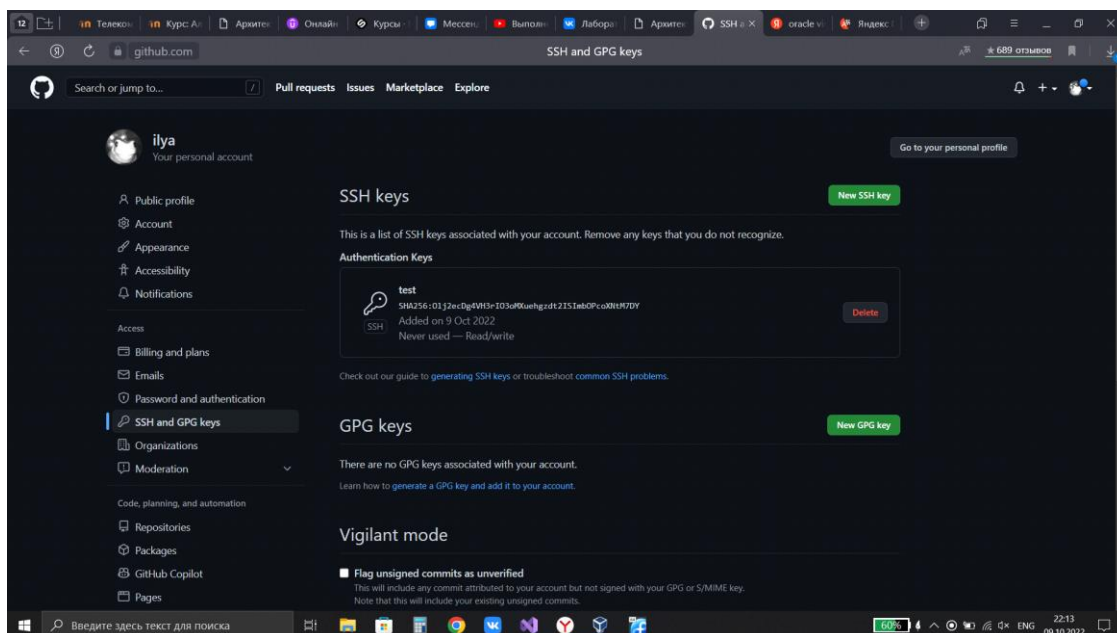
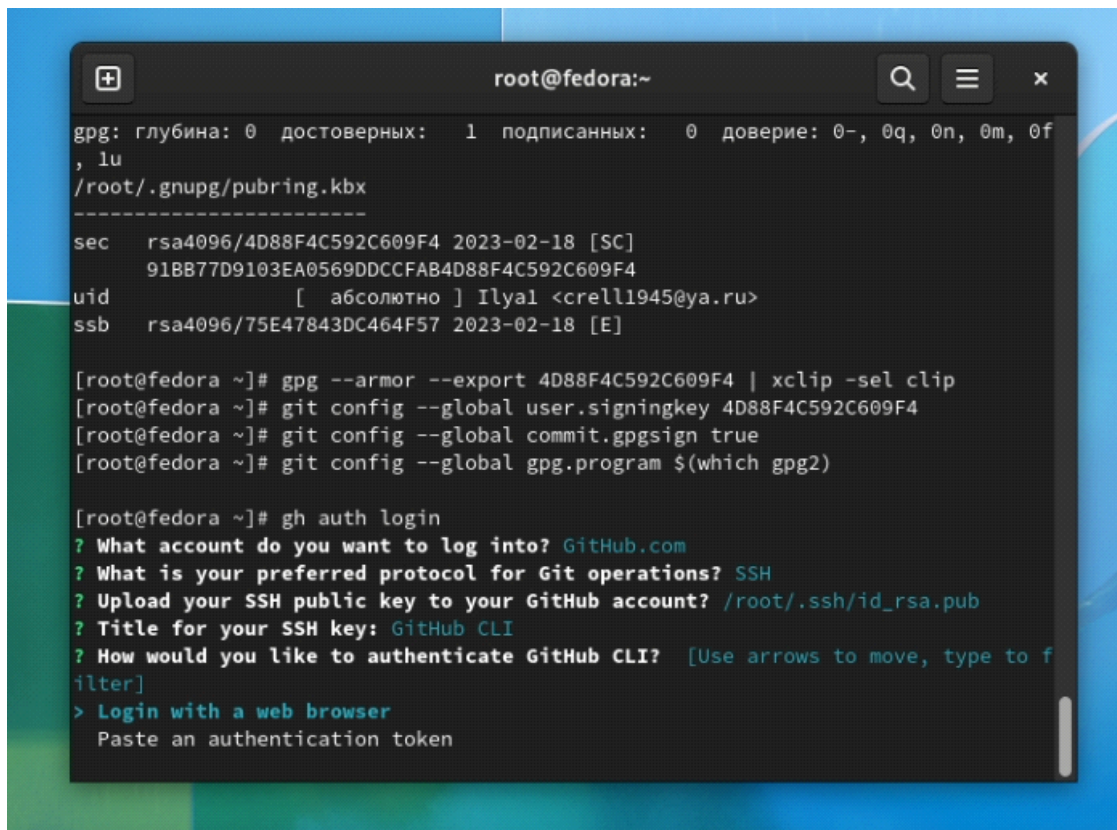


Рис. 10: Добавление PGP ключа в GitHub

№5 Настроим автоматические подписи коммитов git. Используя введённый email, укажем Git применять его при подписи коммитов. (рис. 11)



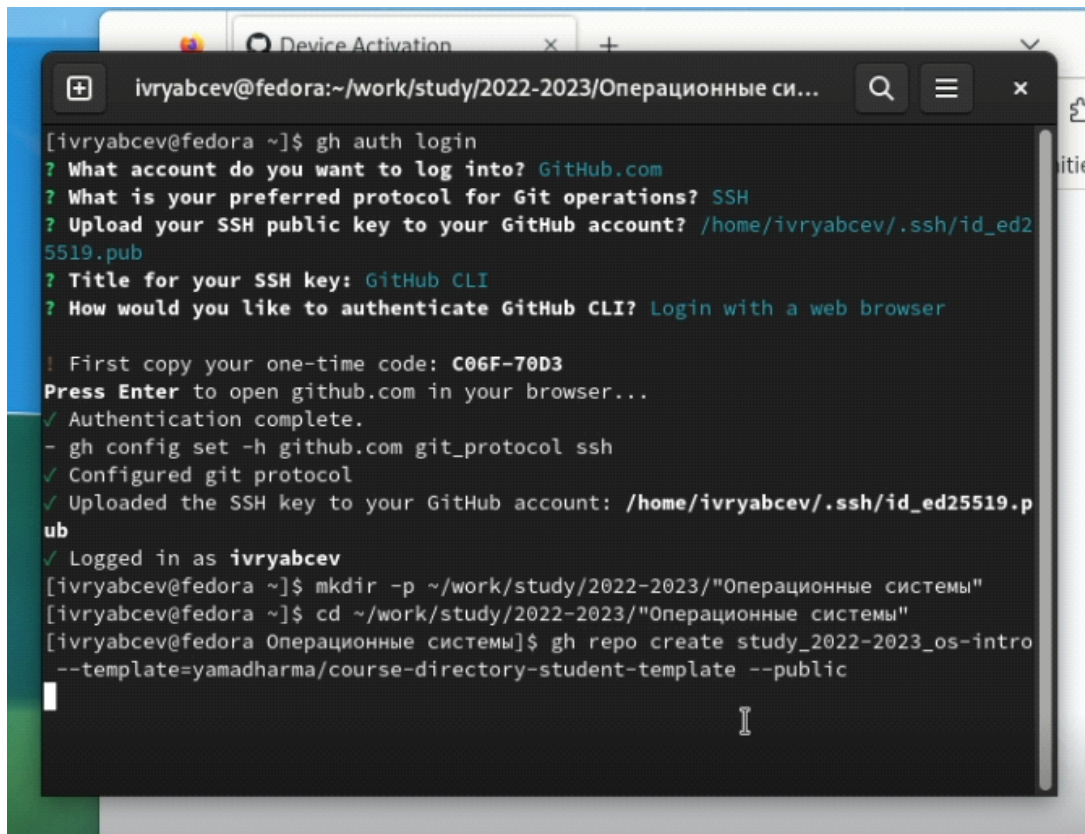
```
root@fedora:~
gpg: глубина: 0  достоверных: 1  подписанных: 0  доверие: 0-, 0q, 0n, 0m, 0f
, 1u
/root/.gnupg/pubring.kbx
-----
sec   rsa4096/4D88F4C592C609F4 2023-02-18 [SC]
      91BB77D9103EA0569DDCCFAB4D88F4C592C609F4
uid           [ абсолютно ] Ilya1 <crell1945@ya.ru>
ssb   rsa4096/75E47843DC464F57 2023-02-18 [E]

[root@fedora ~]# gpg --armor --export 4D88F4C592C609F4 | xclip -sel clip
[root@fedora ~]# git config --global user.signingkey 4D88F4C592C609F4
[root@fedora ~]# git config --global commit.gpgsign true
[root@fedora ~]# git config --global gpg.program $(which gpg2)

[root@fedora ~]# gh auth login
? What account do you want to log into? GitHub.com
? What is your preferred protocol for Git operations? SSH
? Upload your SSH public key to your GitHub account? /root/.ssh/id_rsa.pub
? Title for your SSH key: GitHub CLI
? How would you like to authenticate GitHub CLI? [Use arrows to move, type to filter]
> Login with a web browser
  Paste an authentication token
```

Рис. 11: Настройка автоматических подписей коммитов git

Настроим gh. (рис. 12)



```
[ivryabcev@fedora ~]$ gh auth login
? What account do you want to log into? GitHub.com
? What is your preferred protocol for Git operations? SSH
? Upload your SSH public key to your GitHub account? /home/ivryabcev/.ssh/id_ed25519.pub
? Title for your SSH key: GitHub CLI
? How would you like to authenticate GitHub CLI? Login with a web browser

! First copy your one-time code: C06F-70D3
Press Enter to open github.com in your browser...
✓ Authentication complete.
- gh config set -h github.com git_protocol ssh
✓ Configured git protocol
✓ Uploaded the SSH key to your GitHub account: /home/ivryabcev/.ssh/id_ed25519.pub
✓ Logged in as ivryabcev
[ivryabcev@fedora ~]$ mkdir -p ~/work/study/2022-2023/"Операционные системы"
[ivryabcev@fedora ~]$ cd ~/work/study/2022-2023/"Операционные системы"
[ivryabcev@fedora Операционные системы]$ gh repo create study_2022-2023_os-intro
--template=yamadharma/course-directory-student-template --public
```

Рис. 12: Настройка gh

Для начала необходимо авторизоваться. Ответим на несколько наводящих вопросов, которые задаст утилита. Авторизуемся можно через браузер. (рис. 13)

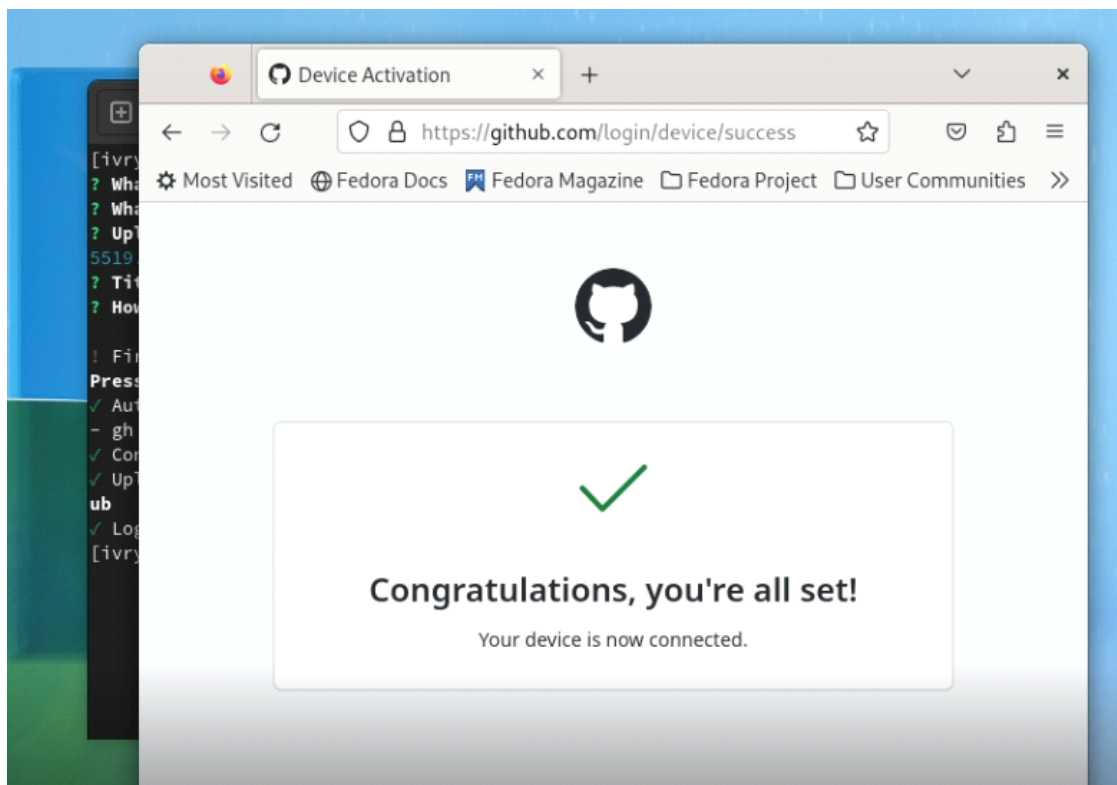


Рис. 13: Подтверждение авторизации

Покажем, что ключ SSH добавился.

Рис. 14: Добавление ssh ключа в GitHub

№6 Необходимо создать шаблон рабочего пространства. (рис. 15)

```
[dmbelicheva@fedora Операционные системы]$ gh repo create study_2021-2022_os-intro --template=yamadharma/course-directory-student-template --public
/ Created repository dmbelicheva/study_2021-2022_os-intro on GitHub
[dmbelicheva@fedora Операционные системы]$ git clone --recursive git@github.com:<owner>/study_2021-2022_os-intro.git os-intro
bash: owner: Нет такого файла или каталога
[dmbelicheva@fedora Операционные системы]$ git clone --recursive git@github.com:<dmbelicheva>/study_2021-2022_os-intro.git os-intro
bash: dmbelicheva: Нет такого файла или каталога
[dmbelicheva@fedora Операционные системы]$ git clone --recursive git@github.com:dmbelicheva/study_2021-2022_os-intro.git os-intro
Клонирование в «os-intro»...
The authenticity of host 'github.com (140.82.121.4)' can't be established.
ED25519 key fingerprint is SHA256:+DiY3wvV6TuJJhpZisF/zLDA0zPMSvHdkr4Uvc0qU.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added 'github.com' (ED25519) to the list of known hosts.
remote: Enumerating objects: 20, done.
remote: Counting objects: 100% (20/20), done.
remote: Compressing objects: 100% (18/18), done.
remote: Total 20 (delta 2), reused 15 (delta 2), pack-reused 0
Получение объектов: 100% (20/20), 12.49 Киб | 4.16 Миб/с, готово.
Определение изменений: 100% (2/2), готово.
Подмодуль «template/presentation» (https://github.com/yamadharma/academic-presentation-markdown-template.git) зарегистрирован по пути «template/presentation»
Подмодуль «template/report» (https://github.com/yamadharma/academic-laboratory-report-template.git) зарегистрирован по пути «template/report»
Клонирование в «/home/dmbelicheva/work/study/2021-2022/Операционные системы/os-intro/template/presentation»...
remote: Enumerating objects: 42, done.
remote: Counting objects: 100% (42/42), done.
remote: Compressing objects: 100% (34/34), done.
remote: Total 42 (delta 9), reused 40 (delta 7), pack-reused 0
Получение объектов: 100% (42/42), 31.19 Киб | 3.46 Миб/с, готово.
Определение изменений: 100% (9/9), готово.
Клонирование в «/home/dmbelicheva/work/study/2021-2022/Операционные системы/os-intro/template/report»...
remote: Enumerating objects: 78, done.
remote: Counting objects: 100% (78/78), done.
remote: Compressing objects: 100% (52/52), done.
remote: Total 78 (delta 31), reused 69 (delta 22), pack-reused 0
Получение объектов: 100% (78/78), 292.27 Киб | 2.20 Миб/с, готово.
Определение изменений: 100% (31/31), готово.
Submodule path 'template/presentation': checked out '3eae7b7586f8a9aded2b506cd1018e625b228b93'
Submodule path 'template/report': checked out 'df7b2ef80f8def3b9a496f8695277469a1a7842a'
```

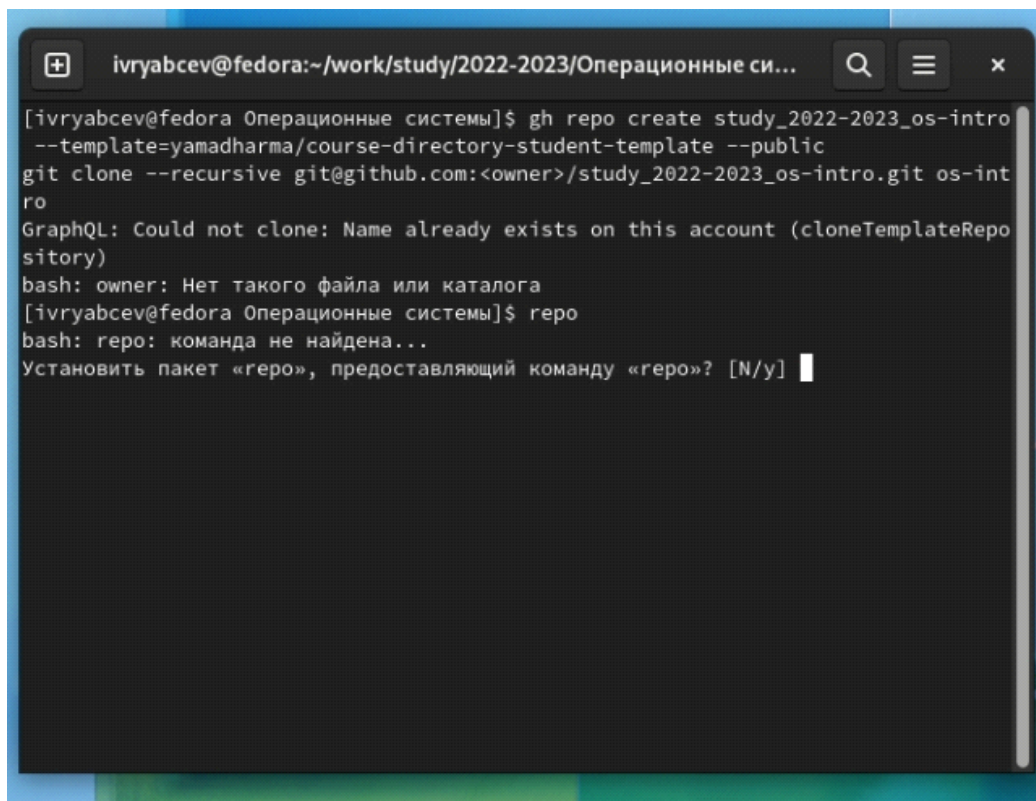
Рис. 15: Создание репозитория курса на основе шаблона

Перейдем в каталог курса. Удалим лишние файлы. Создадим необходимые каталоги. Отправим файлы на сервер. (рис. 16)

```
[dmbelicheva@fedora Операционные системы]$ cd ~/work/study/2021-2022/"Операционные системы"/os-intro
[dmbelicheva@fedora os-intro]$ rm package.json
[dmbelicheva@fedora os-intro]$ make COURSE=os-intro
[dmbelicheva@fedora os-intro]$ git add .
[dmbelicheva@fedora os-intro]$ git commit -am 'feat(main): make course structure'
[master ea26a6b] feat(main): make course structure
149 files changed, 16590 insertions(+), 14 deletions(-)
create mode 100644 labs/lab01/presentation/Makefile
create mode 100644 labs/lab01/presentation/presentation.md
create mode 100644 labs/lab01/report/Makefile
create mode 100644 labs/lab01/report/bib/cite.bib
create mode 100644 labs/lab01/report/image/placeimg_800_600_tech.jpg
create mode 100644 labs/lab01/report/pandoc/csl/gost-r-7-0-5-2008-numeric.csl
create mode 100644 labs/lab01/report/report.md
create mode 100644 labs/lab02/presentation/Makefile
create mode 100644 labs/lab02/presentation/presentation.md
create mode 100644 labs/lab02/report/Makefile
create mode 100644 labs/lab02/report/bib/cite.bib
create mode 100644 labs/lab02/report/image/placeimg_800_600_tech.jpg
create mode 100644 labs/lab02/report/pandoc/csl/gost-r-7-0-5-2008-numeric.csl
create mode 100644 labs/lab02/report/report.md
```

Рис. 16: Настройка каталога курса

Продолжаем настройку каталога. (рис. 17)



```
ivryabcev@fedora:~/work/study/2022-2023/Операционные си...
[ivryabcev@fedora Операционные системы]$ gh repo create study_2022-2023_os-intro
--template=yamadharm/course-directory-student-template --public
git clone --recursive git@github.com:<owner>/study_2022-2023_os-intro.git os-intro
GraphQL: Could not clone: Name already exists on this account (cloneTemplateRepository)
bash: owner: Нет такого файла или каталога
[ivryabcev@fedora Операционные системы]$ repo
bash: repo: команда не найдена...
Установить пакет «геро», предоставляющий команду «геро»? [N/y]
```

Рис. 17: Настройка каталога курса

5 Выводы

Изучила средства контроля версий и научилась применять их. Освоила работу с git, научилась подключать репозитории, добавлять и удалять необходимые файлы.

6 Ответы на контрольные вопросы

- Что такое системы контроля версий (VCS) и для решения каких задач они предназначены? Система контроля версий — программное обеспечение для облегчения работы с изменяющейся информацией. Система управления версиями позволяет хранить несколько версий одного и того же документа, при необходимости возвращаться к более ранним версиям, определять, кто и когда сделал то или иное изменение, и многое другое. Системы контроля версий (Version Control System, VCS) применяются для:
 - Хранение полной истории изменений
 - причин всех производимых изменений
 - Откат изменений, если что-то пошло не так
 - Поиск причины и ответственного за появления ошибок в программе
 - Совместная работа группы над одним проектом
 - Возможность изменять код, не мешая работе других пользователей
- Объясните следующие понятия VCS и их отношения: хранилище, commit, история, рабочая копия. Репозиторий - хранилище версий - в нем хранятся

все документы вместе с историей их изменения и другой служебной информацией. Commit — отслеживание изменений, сохраняет разницу в изменениях Рабочая копия - копия проекта, связанная с репозиторием (текущее состояние файлов проекта, основанное на версии из хранилища (обычно на последней)) История хранит все изменения в проекте и позволяет при необходимости обратиться к нужным данным.

- Что представляют собой и чем отличаются централизованные и децентрализованные VCS? Приведите примеры VCS каждого вида. Централизованные VCS (Subversion; CVS; TFS; VAULT; AccuRev): • Одно основное хранилище всего проекта • Каждый пользователь копирует себе необходимые ему файлы из этого репозитория, изменяет и, затем, добавляет свои изменения обратно Децентрализованные VCS (Git; Mercurial; Bazaar): • У каждого пользователя свой вариант (возможно не один) репозитория • Присутствует возможность добавлять и забирать изменения из любого репозитория [2] В классических системах контроля версий используется централизованная модель, предполагающая наличие единого репозитория для хранения файлов. Выполнение большинства функций по управлению версиями осуществляется специальным сервером. В отличие от классических, в распределённых системах контроля версий центральный репозиторий не является обязательным.
- Опишите действия с VCS при единоличной работе с хранилищем. Сначала создаем и подключаем удаленный репозиторий. Затем по мере изменения проекта отправлять эти изменения на сервер.
- Опишите порядок работы с общим хранилищем VCS. Участник проекта (пользователь) перед началом работы посредством определённых команд получает нужную ему версию файлов. После внесения изменений, пользователь размещает новую версию в хранилище. При этом предыдущие версии не удаляются из центрального хранилища и к ним можно вернуться в любой момент.
- Каковы основные задачи, решаемые инструментальным средством git? Первая — хранить информацию о всех изменениях в вашем коде, начиная с самой первой строчки, а вторая — обеспечение удобства командной работы над кодом.
- Назовите и дайте краткую характеристику командам git. Наиболее часто используемые команды git: • создание основного дерева репозитория: git init • получение обновлений (изменений) текущего дерева из центрального репозитория: git pull • отправка всех произведённых изменений локального дерева в центральный репозиторий: git push • просмотр списка изменённых файлов в текущей директории: git status • просмотр текущих изменений: git diff • сохранение текущих изменений: – добавить все изменённые и/или созданные файлы и/или каталоги: git add. – добавить конкретные изменённые и/или созданные файлы и/или каталоги: git add имена_файлов • удалить файл и/или каталог из индекса репозитория (при этом файл и/или каталог остаётся в локальной директории): git rm имена_файлов • сохранение добавленных изменений: – сохранить все добавленные изменения и все изменённые файлы: git

commit -am 'Описание коммита' – сохранить добавленные изменения с внесением комментария через встроенный редактор git commit • создание новой ветки, базирующейся на текущей: git checkout -b имя_ветки • переключение на некоторую ветку: git checkout имя_ветки (при переключении на ветку, которой ещё нет в локальном репозитории, она будет создана и связана с удалённой) • отправка изменений конкретной ветки в центральный репозиторий: git push origin имя_ветки • слияние ветки с текущим деревом: git merge --no-ff имя_ветки • удаление ветки: – удаление локальной уже слитой с основным деревом ветки: git branch -d имя_ветки – принудительное удаление локальной ветки: git branch -D имя_ветки – удаление ветки с центрального репозитория: git push origin :имя_ветки

- Приведите примеры использования при работе с локальным и удалённым репозиториями. git push --all (push origin master/любой branch)
- Что такое и зачем могут быть нужны ветви (branches)? Ветвление («ветка», branch) — один из параллельных участков истории в одном хранилище, исходящих из одной версии (точки ветвления). [3] • Обычно есть главная ветка (master), или ствол (trunk). • Между ветками, то есть их концами, возможно слияние. Используются для разработки новых функций.
- Как и зачем можно игнорировать некоторые файлы при commit? Во время работы над проектом так или иначе могут создаваться файлы, которые не требуется добавлять в последствии в репозиторий. Например, временные файлы, создаваемые редакторами, или объектные файлы, создаваемые компиляторами. Можно прописать шаблоны игнорируемых при добавлении в репозиторий типов файлов в файл .gitignore с помощью сервисов.

Список литературы

1. О системе контроля версий [Электронный ресурс]. 2016. URL: <https://git-scm.com/book/ru/v2/Введение-О-системе-контроля-версий>.
2. Евгений Г. Системы контроля версий [Электронный ресурс]. 2016. URL: https://glebradchenko.susu.ru/courses/bachelor/engineering/2016/SUSU_SE_2016_R_EP_3_VCS.pdf.
3. Системы контроля версий [Электронный ресурс]. 2016. URL: http://uii.mpei.ru/study/courses/sdt/16/lecture02.2_vcs.slides.pdf.