

**РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ**

**Факультет физико-математических и естественных наук**

**Кафедра прикладной информатики и теории вероятностей**

**ОТЧЕТ**

**ПО ЛАБОРАТОРНОЙ РАБОТЕ № 2**

дисциплина: *Операционные систем*

Студент:

Калистратова Ксения Евгеньевна

Группа: НПМбд-01-20

**МОСКВА**

2021 г.

1. **Цель работы:** Изучить идеологию и применение средств контроля версий.

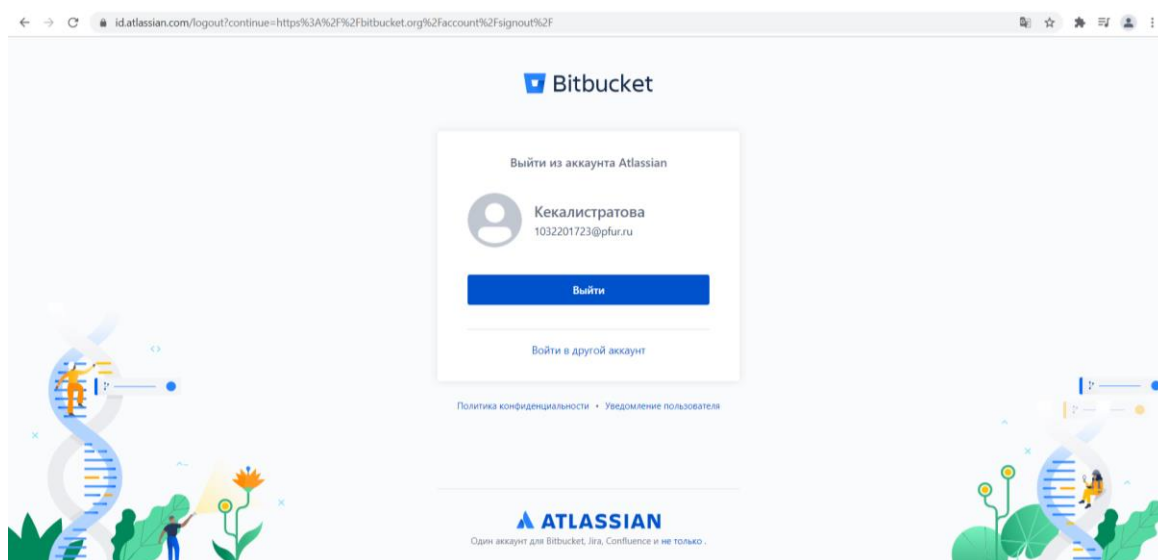
2. **Задание:**

Настроить git, подключить репозиторий к github, изменить первичную конфигурацию и конфигурацию get-flow.

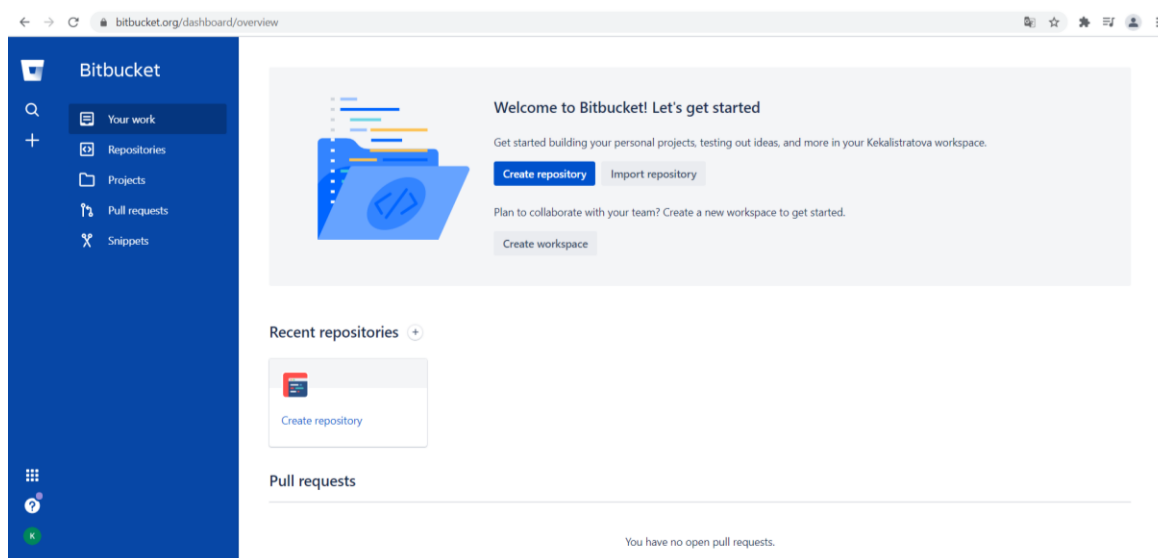
3. **Ход работы:**

1) Для начала я настроила систему контроля версий git, как это описано в указаниях к лабораторной работе с использованием сервера репозитория <http://bitbucket.org/>.

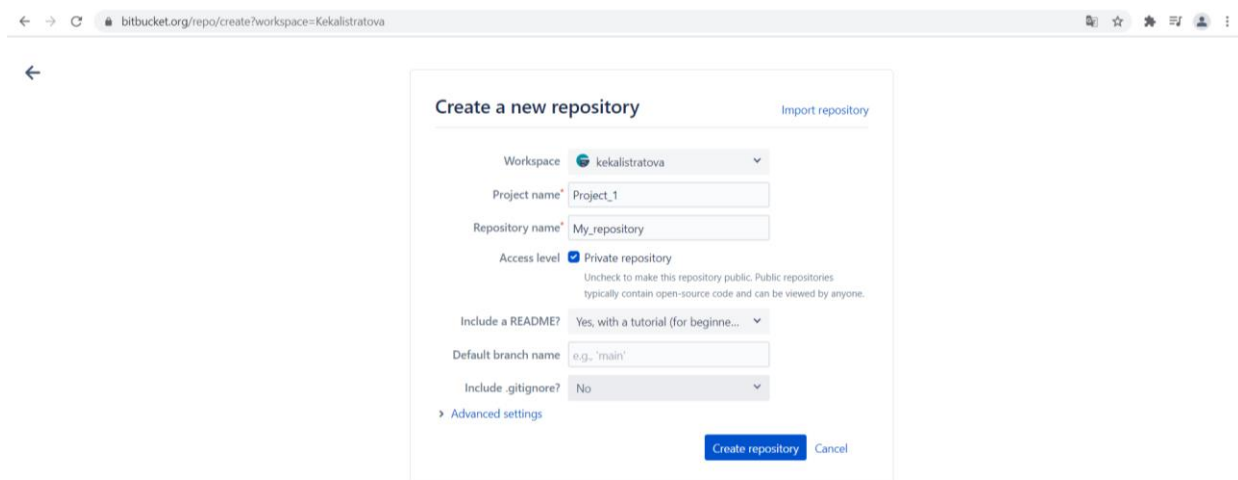
Зарегистрировалась на сайте <http://bitbucket.org/> (Рисунок 1).



Создала новый репозиторий для нового проекта («Create repository»).

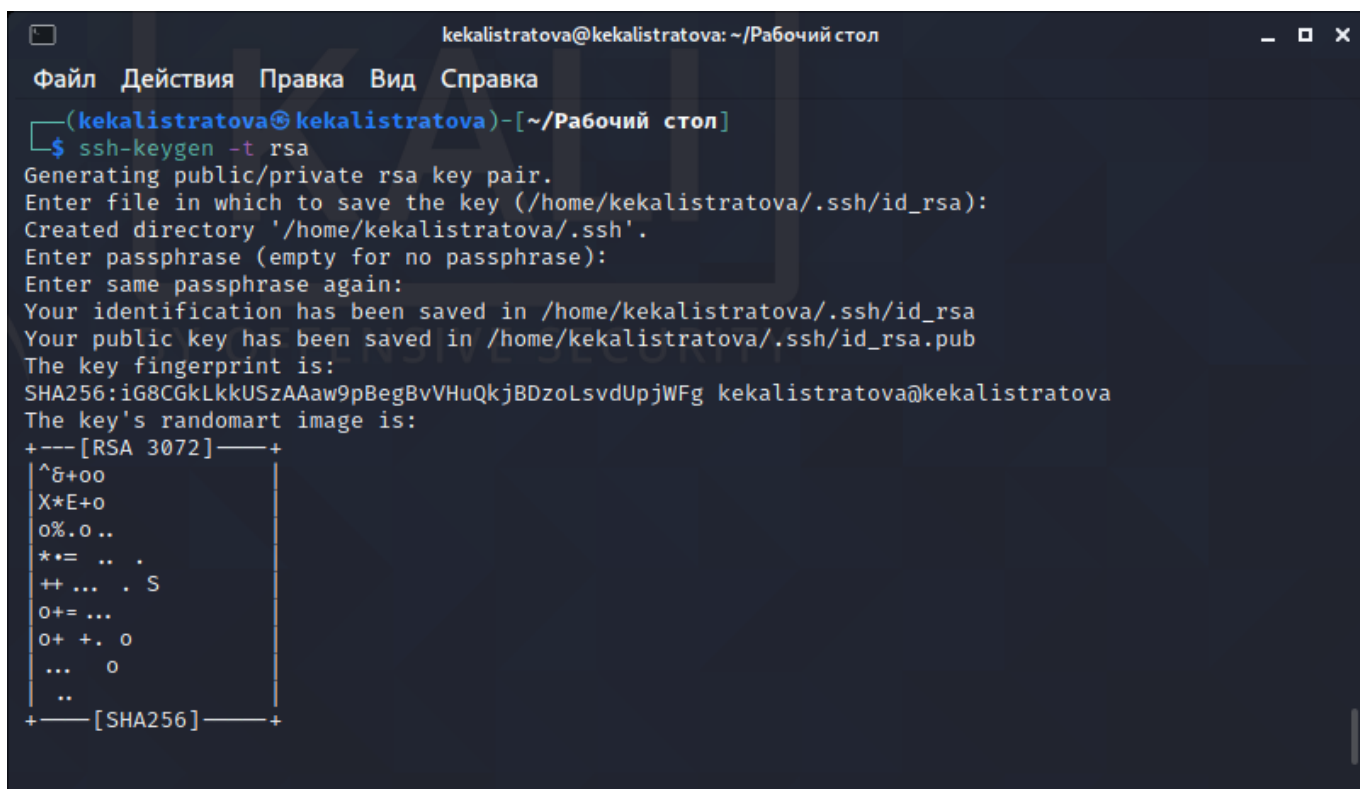


Ввела название репозитория и проекта, выбрала «Private repository», чтобы только я могла иметь доступ к файлам репозитория.



The screenshot shows the Bitbucket web interface for creating a new repository. The form is titled 'Create a new repository' and includes a link to 'Import repository'. The 'Workspace' is set to 'kekalistratova'. The 'Project name' is 'Project\_1' and the 'Repository name' is 'My\_repository'. The 'Access level' is set to 'Private repository' with a note: 'Uncheck to make this repository public. Public repositories typically contain open-source code and can be viewed by anyone.' The 'Include a README?' option is 'Yes, with a tutorial (for beginne...'. The 'Default branch name' is 'e.g., 'main''. The 'Include .gitignore?' option is 'No'. There is an 'Advanced settings' link and 'Create repository' and 'Cancel' buttons.

Далее я открыла терминал и сгенерировала SSH ключ, чтобы связать удаленное файловое хранилище с локальными репозиториями, используя команду «ssh-keygen -C “Ksenia Kalistratova <[1032201723@pfur.ru](mailto:1032201723@pfur.ru)>”»



```
kekalistratova@kekalistratova: ~/Рабочий стол
Файл Действия Правка Вид Справка
(kekalistratova@kekalistratova)-[~/Рабочий стол]
$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/kekalistratova/.ssh/id_rsa):
Created directory '/home/kekalistratova/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/kekalistratova/.ssh/id_rsa
Your public key has been saved in /home/kekalistratova/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:iG8CGkLkkUSzAAaw9pBegBvVHuQkjBDzoLsvdUpjWFg kekalistratova@kekalistratova
The key's randomart image is:
+---[RSA 3072]---+
| ^8+00
| X*E+0
| 0%.0 ..
| *+= .. .
| ++ ... . S
| O+= ...
| O+ +. O
| ... O
| ..
+---[SHA256]---+
```

Скопировала данный ключ в буфер обмена (команда

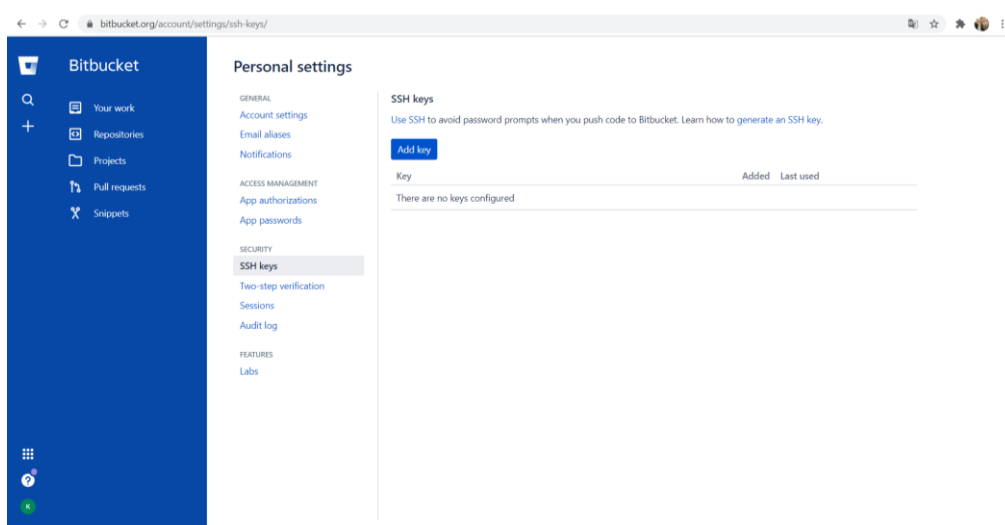
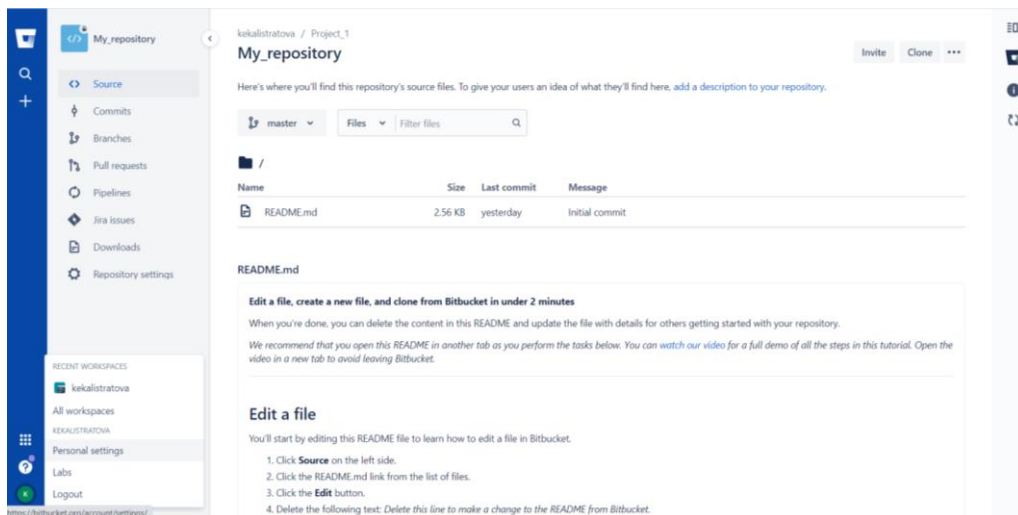
«cat ~/.ssh/id\_rsa.pub | xclip -sel clip»), предварительно установив пакет xclip с помощью команд «sudo apt update» и «sudo apt install xclip».

```
kekalistratova@kekalistratova: ~/Рабочий стол
Файл Действия Правка Вид Справка
(kekalistratova@kekalistratova)~[~/Рабочий стол]
$ sudo apt update
[sudo] пароль для kekalistratova:
Пол:1 http://mirror-1.truenetwork.ru/kali kali-rolling InRelease [30,5 kB]
Пол:2 http://mirror-1.truenetwork.ru/kali kali-rolling/main amd64 Packages [17,7 MB]
Пол:3 http://mirror-1.truenetwork.ru/kali kali-rolling/main amd64 Contents (deb) [39,7 MB]
71% [3 Contents-amd64 19,2 MB/39,7 MB 48%] 101 kB/s 3мин 22с ssh-k
Получено 57,5 MB за 6мин 20с (151 kB/s)
Чтение списков пакетов... Готово
Построение дерева зависимостей... Готово
Чтение информации о состоянии... Готово
Может быть обновлено 287 пакетов. Запустите «apt list --upgradable» для их показа.
```

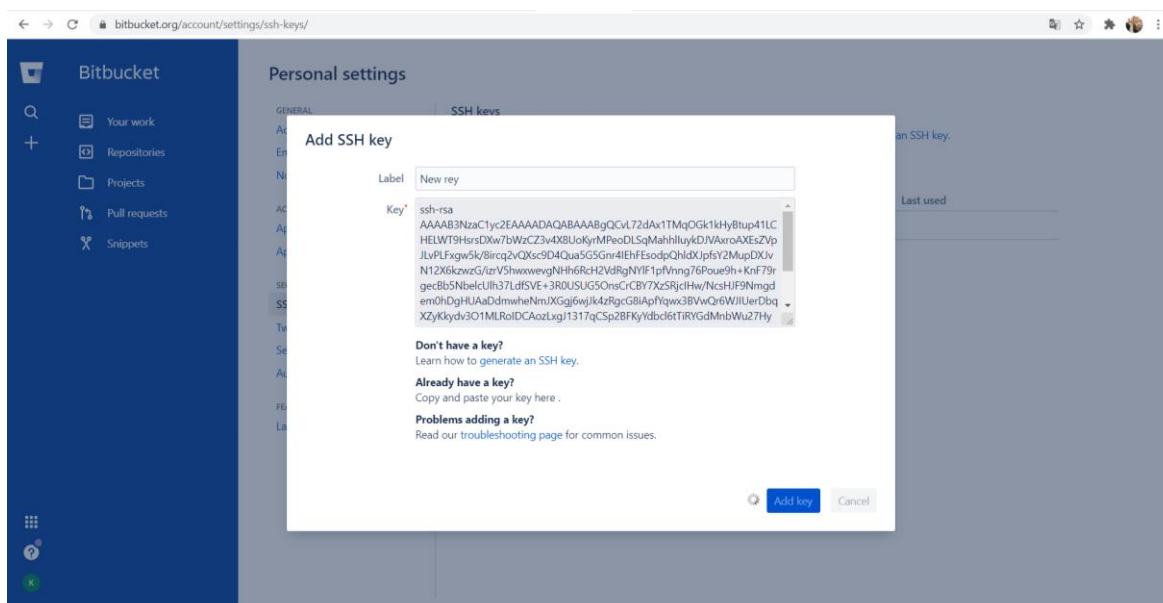
```
kekalistratova@kekalistratova: ~/Рабочий стол
Файл Действия Правка Вид Справка
(kekalistratova@kekalistratova)~[~/Рабочий стол]
$ sudo apt install xclip
[sudo] пароль для kekalistratova:
Чтение списков пакетов... Готово
Построение дерева зависимостей... Готово
Чтение информации о состоянии... Готово
Следующие НОВЫЕ пакеты будут установлены:
  xclip
Обновлено 0 пакетов, установлено 1 новых пакетов, для удаления отмечено 0 пакетов, и 287 пакетов
не обновлено.
Необходимо скачать 23,3 kB архивов.
После данной операции объём занятого дискового пространства возрастёт на 65,5 kB.
Пол:1 http://mirror-1.truenetwork.ru/kali kali-rolling/main amd64 xclip amd64 0.13-2 [23,3 kB]
Получено 23,3 kB за 1с (20,8 kB/s)
Выбор ранее не выбранного пакета xclip.
(Чтение базы данных ... на данный момент установлено 110318 файлов и каталогов.)
Подготовка к распаковке .../xclip_0.13-2_amd64.deb ...
Распаковывается xclip (0.13-2) ...
Настраивается пакет xclip (0.13-2) ...
Обрабатываются триггеры для man-db (2.9.3-2) ...
Обрабатываются триггеры для kali-menu (2021.1.4) ...
```

```
kekalistratova@kekalistratova: ~/Рабочий стол
Файл Действия Правка Вид Справка
(kekalistratova@kekalistratova)~[~/Рабочий стол]
$ cat ~/.ssh/id_rsa.pub | xclip -sel clip
cat: /home/kekalistratova/.ssh/id_rsa.pub: Нет такого файла или каталога
```

Добавила скопированный SSH ключ в Bitbucket. Для этого в левом нижнем углу экрана нажала на значок со своими инициалами (K), выбрала «Personal settings», далее раздел «Security» → «SSH keys» → «Add key»

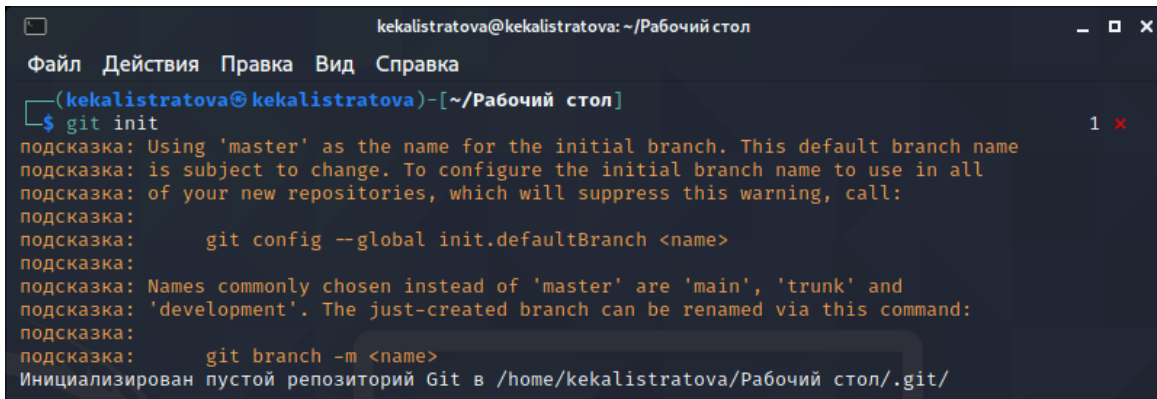


Ввела имя ключа и скопированный ключ.

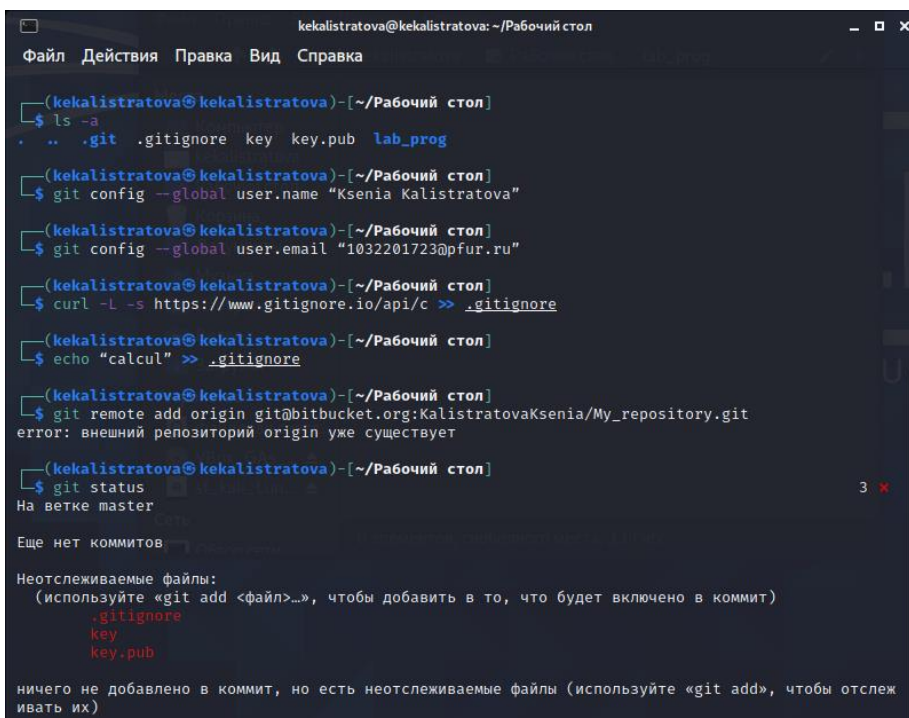


(Рисунок 16)

2) Для создания репозитория я вернулась в терминал и перешла в нужный каталог с помощью команды «`cd ~/home/kekalistratova/Рабочий стол/lab_prog/`». Выполнила инициализацию репозитория (создала основное дерево репозитория), используя команду «`git init`». С помощью команды «`ls -a`» убедилась в том, что появился скрытый каталог `.git`



```
kekalistratova@kekalistratova: ~/Рабочий стол
Файл Действия Правка Вид Справка
(kekalistratova@kekalistratova)-[~/Рабочий стол]
$ git init
подсказка: Using 'master' as the name for the initial branch. This default branch name
подсказка: is subject to change. To configure the initial branch name to use in all
подсказка: of your new repositories, which will suppress this warning, call:
подсказка:
подсказка:     git config --global init.defaultBranch <name>
подсказка:
подсказка: Names commonly chosen instead of 'master' are 'main', 'trunk' and
подсказка: 'development'. The just-created branch can be renamed via this command:
подсказка:
подсказка:     git branch -m <name>
Инициализирован пустой репозиторий Git в /home/kekalistratova/Рабочий стол/.git/
```



```
kekalistratova@kekalistratova: ~/Рабочий стол
Файл Действия Правка Вид Справка
(kekalistratova@kekalistratova)-[~/Рабочий стол]
$ ls -a
. .. .git .gitignore key key.pub lab_prog

(kekalistratova@kekalistratova)-[~/Рабочий стол]
$ git config --global user.name "Ksenia Kalistratova"

(kekalistratova@kekalistratova)-[~/Рабочий стол]
$ git config --global user.email "1032201723@pfur.ru"

(kekalistratova@kekalistratova)-[~/Рабочий стол]
$ curl -L -s https://www.gitignore.io/api/c >> .gitignore

(kekalistratova@kekalistratova)-[~/Рабочий стол]
$ echo "calcul" >> .gitignore

(kekalistratova@kekalistratova)-[~/Рабочий стол]
$ git remote add origin git@bitbucket.org:KalistratovaKsenia/My_repository.git
error: внешний репозиторий origin уже существует

(kekalistratova@kekalistratova)-[~/Рабочий стол]
$ git status
На ветке master

Еще нет коммитов

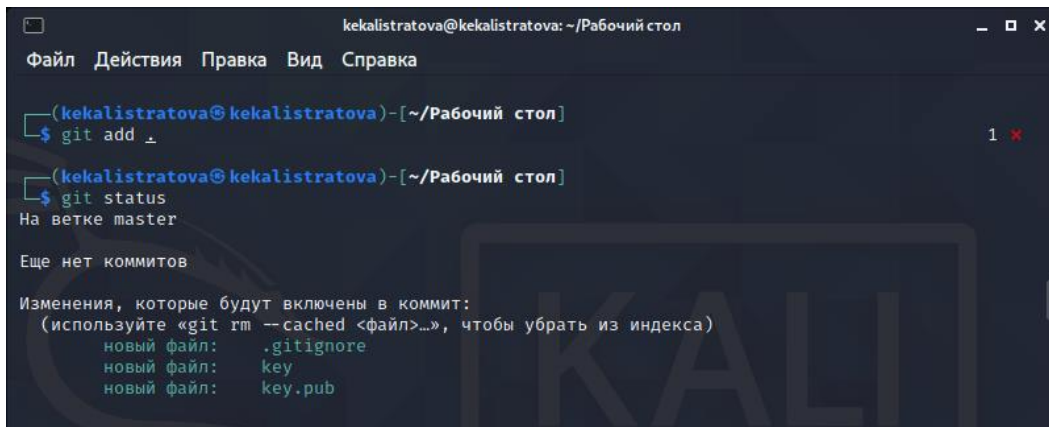
Неотслеживаемые файлы:
(используйте «git add <файл>...», чтобы добавить в то, что будет включено в коммит)
.gitignore
key
key.pub

ничего не добавлено в коммит, но есть неотслеживаемые файлы (используйте «git add», чтобы отслеживать их)
```

(Рисунок 18) Далее перешла в конфигурации. Использовала команды:

- «`git config --global user.name "Ksenia Kalistratova"`» – установка имени пользователя,
- «`git config --global user.email "1032201723@pfur.ru"`» – установка электронного адреса пользователя,

- «curl -L -s <https://www.gitignore.io/api/c> >> .gitignore» – установка списка игнорируемых файлов,
- «echo “calcul” >> .gitignore» – установка списка игнорируемых файлов,
- «git remote add origin git@bitbucket.org:KalistratovaKsenia/My\_repository.git» – добавление ссылки на удаленный репозиторий, чтобы потом загрузить туда локальный репозиторий,
- «git status» – просмотр статуса репозитория (Рисунки 19, 20). Далее добавила все файлы из каталога в индекс (команда «git add .»), добавила комментарий с помощью встроенного редактора (команда «git commit –am “Lab02”»)



```

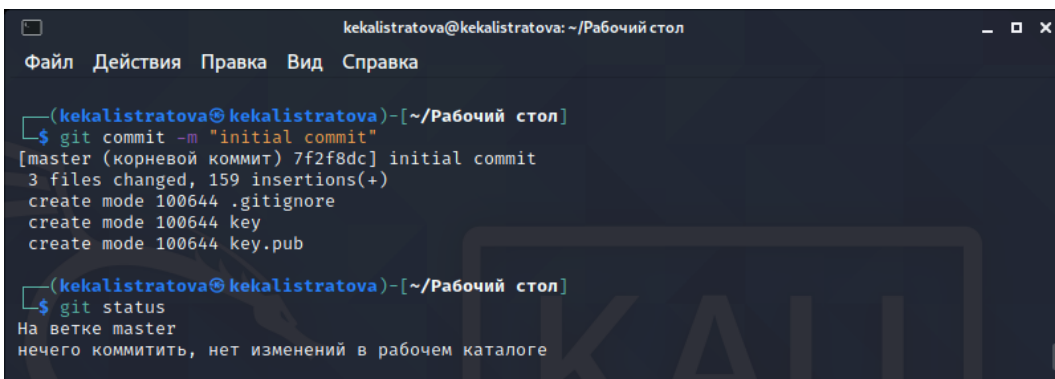
kekalistratova@kekalistratova: ~/Рабочий стол
Файл Действия Правка Вид Справка

(kekalistratova@kekalistratova)-[~/Рабочий стол]
$ git add .
1 ✖

(kekalistratova@kekalistratova)-[~/Рабочий стол]
$ git status
На ветке master

Еще нет коммитов

Изменения, которые будут включены в коммит:
(используйте «git rm --cached <файл>...», чтобы убрать из индекса)
    новый файл:   .gitignore
    новый файл:   key
    новый файл:   key.pub
  
```



```

kekalistratova@kekalistratova: ~/Рабочий стол
Файл Действия Правка Вид Справка

(kekalistratova@kekalistratova)-[~/Рабочий стол]
$ git commit -m "initial commit"
[master (корневой коммит) 7f2f8dc] initial commit
3 files changed, 159 insertions(+)
 create mode 100644 .gitignore
 create mode 100644 key
 create mode 100644 key.pub

(kekalistratova@kekalistratova)-[~/Рабочий стол]
$ git status
На ветке master
ничего коммитить, нет изменений в рабочем каталоге
  
```

4) Проверила статус файлов в каталоге (команда «git status»). Терминал вывел, что файл calculate.c был изменен. Добавила измененный файл в индекс с помощью команды «git add calculate.c», добавила новый комментарий (команда «git commit -m “New commit”») и проверила статус снова. После этого загрузила локальный репозиторий на удаленный репозиторий сервера Bitbucket с помощью команды «git push -u origin master». Проверила наличие файлов в удаленном репозитории.

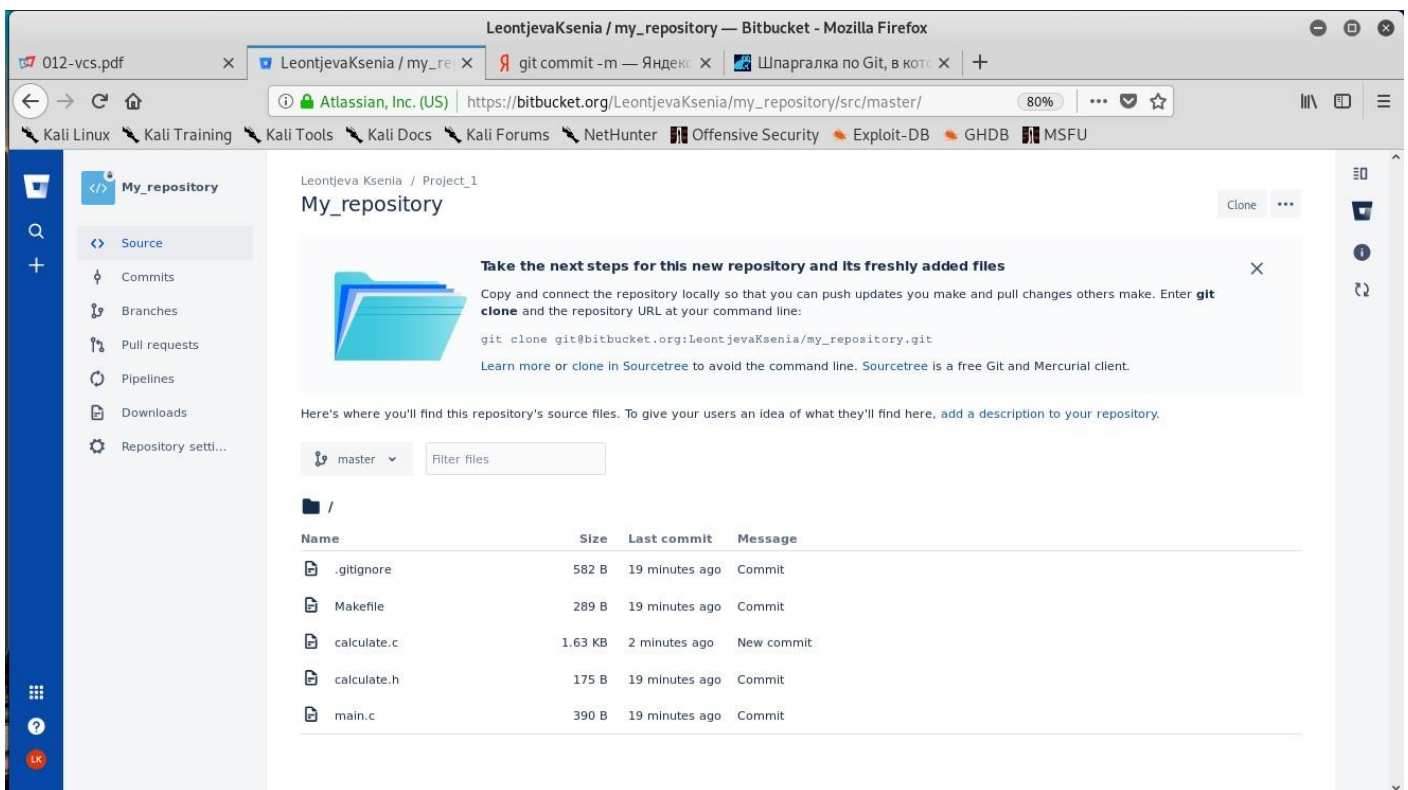


```
kaleontjeva@kaleontjeva: ~/work/os/lab_prog$ git status
На ветке master
Изменения, которые не в индексе для коммита:
(используйте «git add <файл>...», чтобы добавить файл в индекс)
(use «git restore <file>...» to discard changes in working directory)
изменено: calculate.c

нет изменений добавленных для коммита
(используйте «git add» и/или «git commit -a»)
kaleontjeva@kaleontjeva:~/work/os/lab_prog$ git add calculate.c
```

```
kaleontjeva@kaleontjeva:~/work/os/lab_prog$ git commit -m "New commit"
[master 2439df4] New commit
1 file changed, 2 insertions(+)
kaleontjeva@kaleontjeva:~/work/os/lab_prog$ git status
На ветке master
ничего коммитить, нет изменений в рабочем каталоге
kaleontjeva@kaleontjeva:~/work/os/lab_prog$ git push -u origin master
The authenticity of host 'bitbucket.org (18.205.93.0)' can't be established.
RSA key fingerprint is SHA256:zzXQ0XSRBEiUtuE8AikJYKwbHaxvSc0oJez9YXaGp1A.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'bitbucket.org,18.205.93.0' (RSA) to the list of known hosts.
Перечисление объектов: 10, готово.
Подсчет объектов: 100% (10/10), готово.
Сжатие объектов: 100% (10/10), готово.
Запись объектов: 100% (10/10), 1.89 KiB | 968.00 KiB/s, готово.
Всего 10 (изменения 2), повторно использовано 0 (изменения 0)
To bitbucket.org:LeontjevaKsenia/My_repository.git
* [new branch] master -> master
Ветка «master» отслеживает внешнюю ветку «master» из «origin».
kaleontjeva@kaleontjeva:~/work/os/lab_prog$
```

(Рисунок 24)



(Рисунок 25)

### 3. Контрольные вопросы:



1) Система контроля версий (Version Control System, VCS) – программное обеспечение для облегчения работы с изменяющейся информацией. VCS 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 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.

Сервер может сохранять не полную версию изменённых файлов, а производить так называемую дельта-компрессию – сохранять только изменения между последовательными версиями, что позволяет уменьшить объём хранимых данных.

Системы контроля версий поддерживают возможность отслеживания и разрешения конфликтов, которые могут возникнуть при работе нескольких человек над одним файлом. Можно объединить (слить) изменения, сделанные разными участниками (автоматически или вручную), вручную выбрать нужную версию, отменить изменения вовсе или заблокировать файлы для изменения. В зависимости от настроек блокировка не позволяет другим пользователям получить рабочую копию или препятствует изменению рабочей копии файла средствами файловой системы ОС, обеспечивая таким образом, привилегированный доступ только одному пользователю, работающему с файлом.

2) *Хранилище (репозиторий)* – в нем хранятся все документы вместе с историей их изменения и другой служебной информацией.

Рабочую копию необходимо периодически синхронизировать с репозиторием, эта операция предполагает отправку в него изменений, которые пользователь

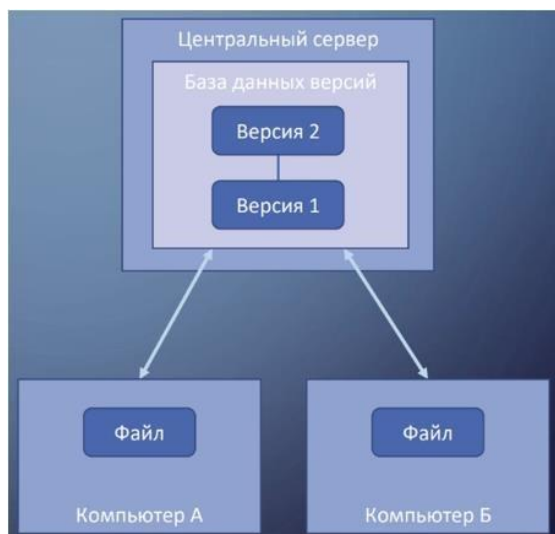
внес в свою рабочую копию. За это отвечает команда *commit*. С помощью этой команды можно также передать комментарий о сделанных изменениях.

*История* – список предыдущих изменений/коммитов.

*Рабочая копия* – копия проекта, связанная с репозиторием, в которой непосредственно работает пользователь.

### 3) Централизованные VCS

Клиент-серверная модель: один центральный репозиторий, с которым разработчики взаимодействуют по сети.

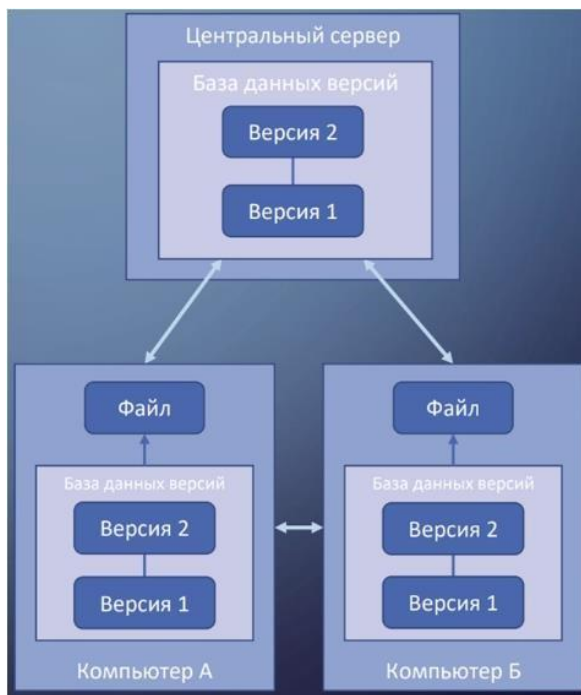


Примеры:

- CVS
- Subversion (SVN)
- Perforce

### Децентрализованная VCS

Клиенты полностью копируют репозиторий, вместо простого скачивания снимка всех файлов (состояния файлов в определенный момент времени). В этом случае, если сервер выйдет из строя, то клиентский репозиторий можно будет скопировать на другой, рабочий, сервер, ведь данный репозиторий является полным бэкапом всех данных.



Примеры:

- Git □ Mercurial
- Bazaar

4) Действия с VCS при единоличной работе с хранилищем:

- Создать новый репозиторий на локальном устройстве (если он не был создан),
- Внести изменения в исходные файлы,
- Выполнить индексацию необходимых файлов,
- Проверить внесенные изменения,
- Выполнить commit,
- Отправить локальный репозиторий на удаленный сервер, при необходимости.

5) Действия при работе с общим хранилищем VCS:

- Обычно проект уже создан и его нужно загрузить из общего удаленного хранилища,
- Необходимо создать свою рабочую ветку,
- Внести изменения внутри своей рабочей ветки,
- Выполнить индексацию необходимых файлов на своем локальном устройстве,

- Проверить внесенные изменения,
- Выполнить `commit`,
- Свою рабочую ветку отправить в общее хранилище,
- При необходимости внести изменения и отправить снова,
- После администратор объединит вашу ветку с `master branch`.

б) Git – это система управления версиями. У Git две основных задачи: первая – хранить информацию о всех изменениях в вашем коде, начиная с самой первой строчки, а вторая – обеспечение удобства командной работы над кодом.

Git запоминает не все изменения, а только те, которые вы скажите.

Обычно работа с Git выглядит так:

- сверстали шапку сайта – сделали `commit`;
- сверстали контент страницы – сделали второй `commit`;
- закончили верстать страницу – сделали третий `commit` и отправили код на сервер, чтобы вашу работу могли увидеть коллеги, либо чтобы опубликовать страницу с помощью

Capistrano.

7) 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*

8) Примеры использования VCS при работе с локальными репозиториями: Создание небольшого проекта на своем локальном устройстве, работа с файлами (например, каталог, содержащий документы, презентации, которые будут часто редактироваться), работа с фотографиями, видео и т.д.

Примеры использования VCS при работе с удаленными репозиториями: Примеры могут быть те же, но теперь над ними работают несколько человек. Такая система позволяет следить за работой других пользователей.

9) Ветвление («ветка», branch) – один из параллельных участков истории в одном хранилище, исходящих из одной версии (точки ветвления).

- Обычно есть главная ветка (master), или ствол (trunk).
- Между ветками, то есть их концами, возможно слияние.

Ветки используются для разработки одной части функционала изолированно от других. Каждая ветка представляет собой отдельную копию кода проекта.

Ветки позволяют одновременно работать над разными версиями проекта.

Каждый репозиторий по-умолчанию имеет ветку master. Всякий раз, когда требуется разработка нового функционала, не внося при этом изменений в основную рабочую версию, можно создавать новую ветку, на основе рабочей, и вести разработку в ней – новой копии кода проекта. Когда функционал доделан и оттестирован, можно сделать merge – слить отдельную ветку обратно с основной. При слиянии этой временной ветки в основную, все её коммиты разом перенесутся из одной в другую.

Ветвление позволяет обеспечить процесс, при котором всегда в наличии будет рабочая версия проекта, в которой не будет частично завершённого функционала находящегося в активной разработке или же протестированных фич.

10) 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.

Для этого сначала нужно получить список имеющихся шаблонов:

*curl -L -s https://www.gitignore.io/api/list* Затем скачать

шаблон, например, для C и C++ *curl -L -s*

*https://www.gitignore.io/api/c >> .gitignore curl -L -s*

*https://www.gitignore.io/api/c++ >> .gitignore*

**4. Вывод:** В ходе выполнения данной лабораторной работы я изучила идеологию и применение средств контроля версий.