

Отчёт по лабораторной работе 3

(лаб 2 переделанная в markdown)

Свояк Валерия Дмитриевна

Содержание

1 Цель работы	5
2 Задание	6
3 Выполнение лабораторной работы	7
3.1 Настройка git	7
3.2 Теперь делаем первичную конфигурацию – создаем лицензию: . .	14
3.3 Конфигурация git flow	16
4 Выводы	21

List of Tables

List of Figures

1 Цель работы

Изучить идеологию и применение средств контроля версий (в лабораторной 2)

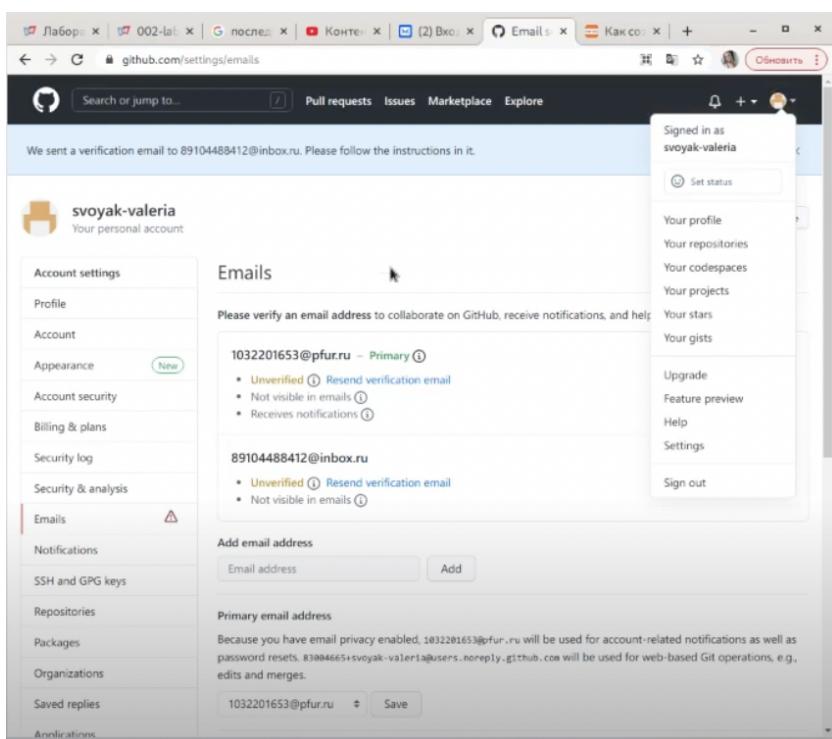
Научиться оформлять отчёты с помощью легковесного языка разметки Markdown (в лабораторной 3)

2 Задание

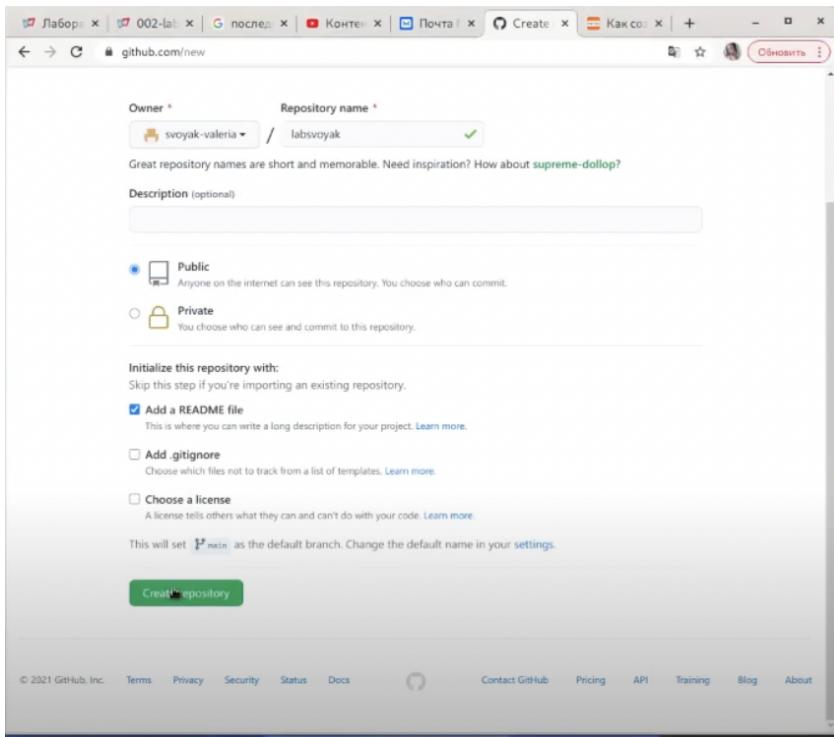
Лабораторная работа является небольшой научно-исследовательской работой, которую и оформлять следует по всем утверждённым требованиям. При подготовке отчета по лабораторной работе вы освоите ряд важных элементов, которые в дальнейшем пригодятся вам при написании курсовой и дипломной работы.

3 Выполнение лабораторной работы

3.1 Настройка git

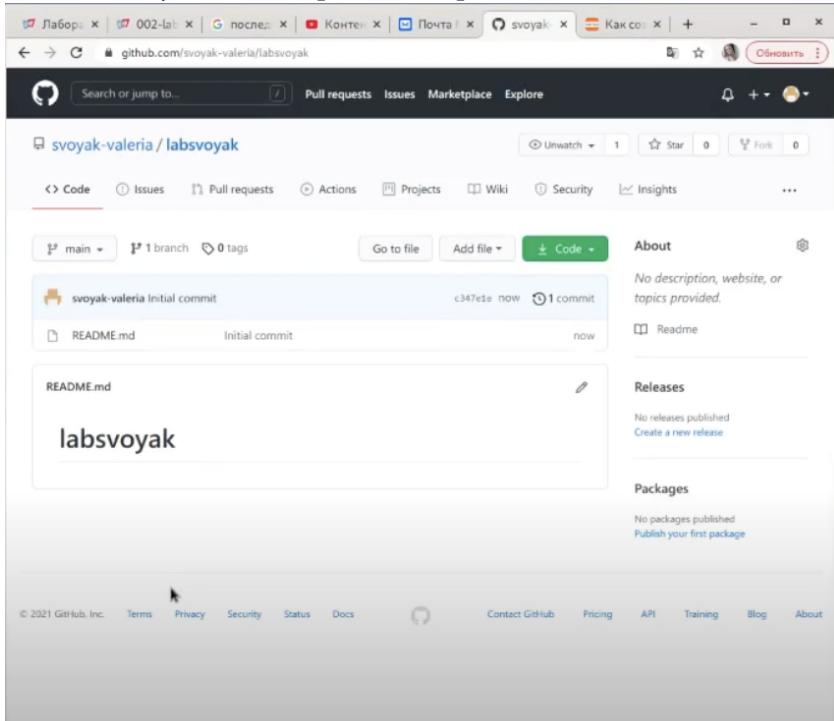


Создайте учётную запись на <https://github.com>.



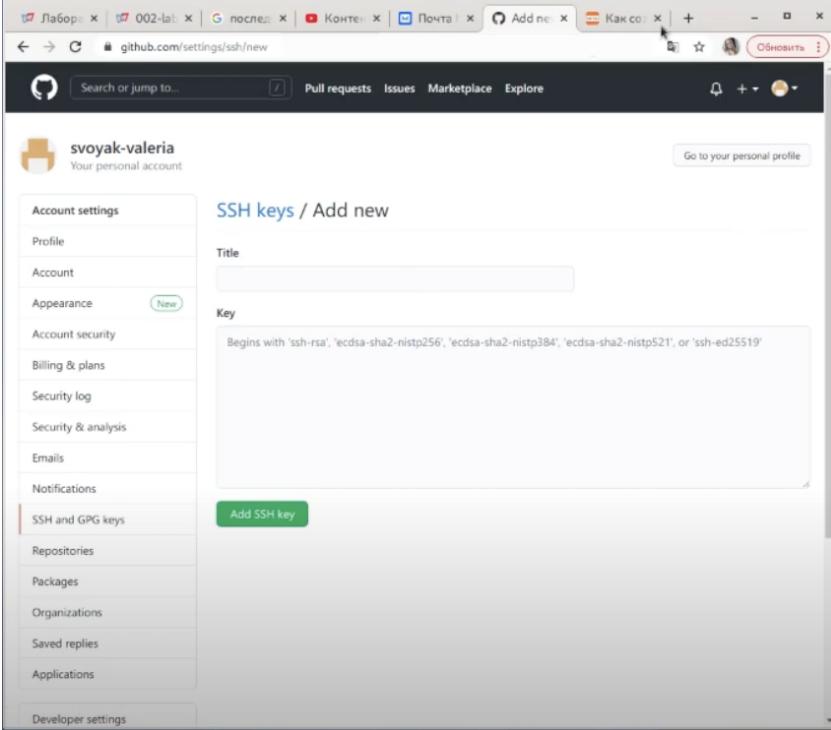
Настройте систему контроля версий git, как это описано выше с использованием сервера репозиториев <https://github.com/>.

Создаем публичный репозиторий на гитхаб 1



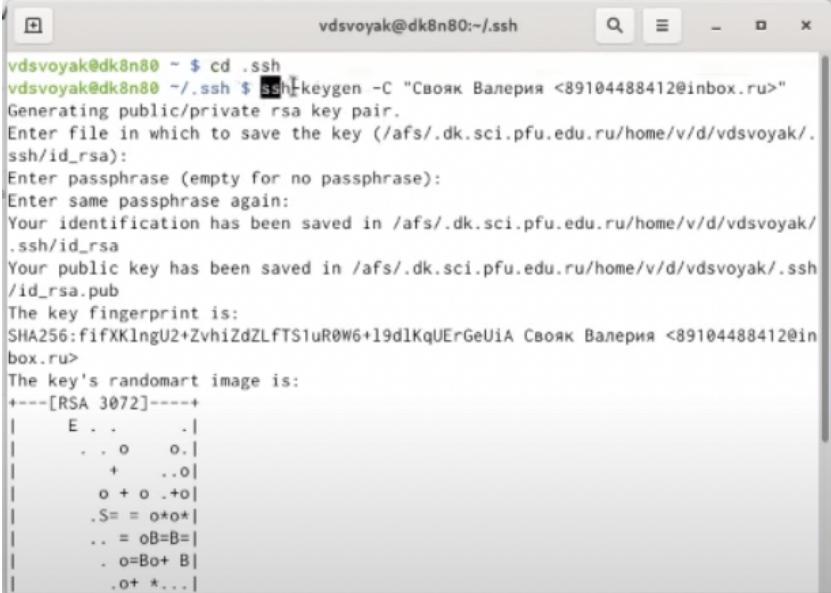
Появился

Устанавливаем ключ для связи



The screenshot shows the GitHub account settings page for 'svoyak-valeria'. The left sidebar has a 'SSH and GPG keys' section highlighted. The main area is titled 'SSH keys / Add new' with fields for 'Title' and 'Key'. A note says 'Begins with 'ssh-rsa'', 'ecdsa-sha2-nistp256'', 'ecdsa-sha2-nistp384'', 'ecdsa-sha2-nistp521'', or 'ssh-ed25519''. A green 'Add SSH key' button is at the bottom.

Заходим в настройки гитхаб и в установку SSH ключей

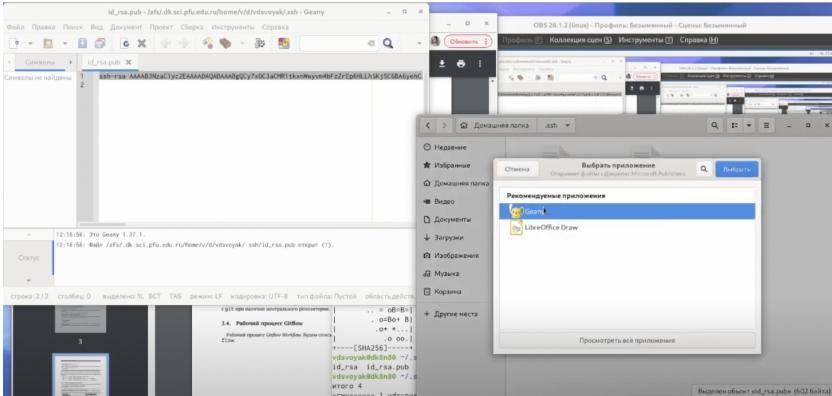


```
vdsvoyak@dk8n80 ~ $ cd .ssh  
vdsvoyak@dk8n80 ~/ssh $ ssh-keygen -C "Свойк Валерия <89104488412@inbox.ru>"  
Generating public/private rsa key pair.  
Enter file in which to save the key (/afs/.dk.sci.pfu.edu.ru/home/v/d/vdsvoyak/.  
ssh/id_rsa):  
Enter passphrase (empty for no passphrase):  
Enter same passphrase again:  
Your identification has been saved in /afs/.dk.sci.pfu.edu.ru/home/v/d/vdsvoyak/  
.ssh/id_rsa  
Your public key has been saved in /afs/.dk.sci.pfu.edu.ru/home/v/d/vdsvoyak/.ssh/  
/id_rsa.pub  
The key fingerprint is:  
SHA256:fifXKlNgU2+ZvhizdZLFTS1uR0W6+19dlKqUErGeUiA Свойк Валерия <89104488412@in  
box.ru>  
The key's randomart image is:  
+---[RSA 3072]---+  
| E . . . |  
| . . o o . |  
| + ..o |  
| o + o .+o |  
| .S= = o*o* |  
| .. = oB=B= |  
| . o=Bo+ B |  
| .o+ *... |
```

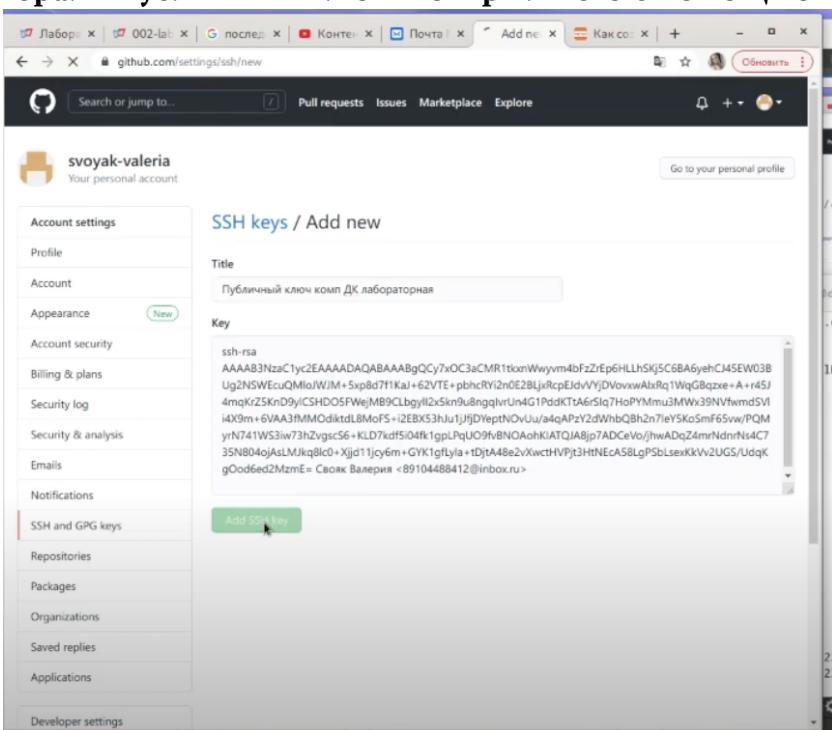
Прописываем команду cd.ssh для перехода в каталог и генерируем публичный и приватный ключи ssh-keygen -C “Свойк Валерия 89104488412@inbox.ru”

```
vdsvoyak@dk8n80 ~/.ssh $ ls
id_rsa id_rsa.pub
vdsvoyak@dk8n80 ~/.ssh $ ls -l
итого 4
-rw----- 1 vdsvoyak studsci 2643 апр 22 12:12 id_rsa
-rw----r-- 1 vdsvoyak studsci 602 апр 22 12:12 id_rsa.pub
vdsvoyak@dk8n80 ~/.ssh $
```

Убедились, что ключ появился:



Чтобы вставить ключ в гетхаб, нам нужно получить на него ссылку. Для этого воспользуемся текстовым редактором. На фото можно видеть, как мы выбрали публичный ключ и открыли его с помощью текстового редактора



Вставляем ссылку на гитхаб

The screenshot shows the GitHub 'Settings' page under 'SSH and GPG keys'. On the left, a sidebar lists account settings like Profile, Account, Appearance, Account security, Billing & plans, Security log, Security & analysis, Emails, Notifications, SSH and GPG keys (which is selected), Repositories, Packages, Organizations, Saved replies, Applications, and Developer settings. The main content area is titled 'SSH keys' and displays a single public key entry:

- Публичный ключ комм ДК лабораторная**
- SHA256: f1xx81ngU2+zvh12dZLFTS1uRtM+19d1kqfErGet1a
- Added on 22 Apr 2021
- Never used — Read/write
- Delete**

Below this, there's a note about generating SSH keys and troubleshooting common SSH problems, and a 'New SSH key' button.

Ключ установлен

```
vdsvoyak@dk8n80:~/.ssh $ cd ..
vdsvoyak@dk8n80:~ $ mkdir Laboratory
vdsvoyak@dk8n80:~ $ cd Laboratory/
^C^C^C^C

vdsvoyak@dk8n80:~/Laboratory $ 
vdsvoyak@dk8n80:~/Laboratory $ 
vdsvoyak@dk8n80:~/Laboratory $ ^C
vdsvoyak@dk8n80:~/Laboratory $ git clone git@github.com:svoyak-valeria/labsvoyak.git
Клонирование в «labsvoyak»...
The authenticity of host 'github.com (140.82.121.3)' can't be established.
RSA key fingerprint is SHA256:nThbg6kXUpJWG17E1IGOCspRomTxdCARLviKw6E5SY8.
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,140.82.121.3' (RSA) to the list of known hosts.
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Получение объектов: 100% (3/3), готово.
vdsvoyak@dk8n80:~/Laboratory $ ls
labsvoyak
vdsvoyak@dk8n80:~/Laboratory $ cd labsvoyak/
vdsvoyak@dk8n80:~/Laboratory/labsvoyak $ ls -al
итого 7
```

Создаем каталог лаборатория и клонируем в него наш ключ

```
vdsvoyak@dk8n80 ~/Laboratory/labsvoyak $ ls -al
итого 7
drwxr-xr-x 3 vdsvoyak studsci 2048 апр 22 12:26 .
drwxr-xr-x 3 vdsvoyak studsci 2048 апр 22 12:26 ..
drwxr-xr-x 7 vdsvoyak studsci 2048 апр 22 12:26 .git
-trw-r--r-- 1 vdsvoyak studsci 11 апр 22 12:26 README.md
```

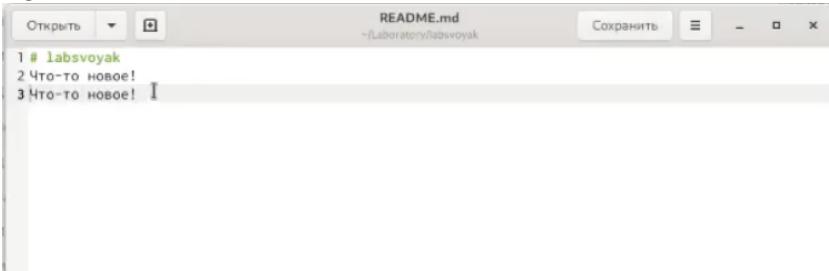
Посмотрим, какие файлы есть в каталоге

Чтобы создать первый коммит, изменим содержание файла README.md.

Для этого:

```
vdsvoyak@dk8n80 ~/Laboratory/labsvoyak $ echo "Что-то новое!" >> README.md
vdsvoyak@dk8n80 ~/Laboratory/labsvoyak $ echo "\nЧто-то новое!" >> README.md
```

В текстовый документ добавляем фразу «Что-то новое» с помощью команды echo



Текст добавился

```
vdsvoyak@dk8n80 ~/Laboratory/labsvoyak $ cat README.md
# labsvoyak
Что-то новое!
\пЧто-то новое!
```

Можем просмотреть это через команду cat

Изменения произошли, можно создавать коммит. Для этого:

```
vdsvoyak@dk8n80:~/Laboratory/labsvoyak
```

```
-rw-r--r-- 1 vdsvoyak studsci 11 апр 22 12:26 README.md
vdsvoyak@dk8n80 ~/Laboratory/labsvoyak $ ls -al
итого 7
drwxr-xr-x 3 vdsvoyak studsci 2048 апр 22 12:26 .
drwxr-xr-x 3 vdsvoyak studsci 2048 апр 22 12:26 ..
drwxr-xr-x 7 vdsvoyak studsci 2048 апр 22 12:26 .git
-rw-r--r-- 1 vdsvoyak studsci 11 апр 22 12:26 README.md
vdsvoyak@dk8n80 ~/Laboratory/labsvoyak $ echo "Что-то новое!" >> README.md
vdsvoyak@dk8n80 ~/Laboratory/labsvoyak $ echo "\nЧто-то новое!" >> README.md
vdsvoyak@dk8n80 ~/Laboratory/labsvoyak $ cat README.md
# labsvoyak
Что-то новое!
\пЧто-то новое!
vdsvoyak@dk8n80 ~/Laboratory/labsvoyak $ git commit -m "first commit"
*** Пожалуйста, скажите мне кто вы есть.

Запустите

git config --global user.email "you@example.com"
git config --global user.name "Ваше Имя"

для указания идентификационных данных аккаунта по умолчанию.
Пропустите параметр --global для указания данных только для этого репозитория.

fatal: не удалось выполнить автоопределение адреса электронной почты (получено «vdsvoyak@dk8n80.(none)»)
vdsvoyak@dk8n80 ~/Laboratory/labsvoyak $ ^C
vdsvoyak@dk8n80 ~/Laboratory/labsvoyak $ git config --global user.email "89104488412@inbox.ru"
vdsvoyak@dk8n80 ~/Laboratory/labsvoyak $ git config --global user.email "svoyak-valeria"
vdsvoyak@dk8n80 ~/Laboratory/labsvoyak $ git commit -m "first commit"
На ветке main
Ваша ветка обновлена в соответствии с «origin/main».

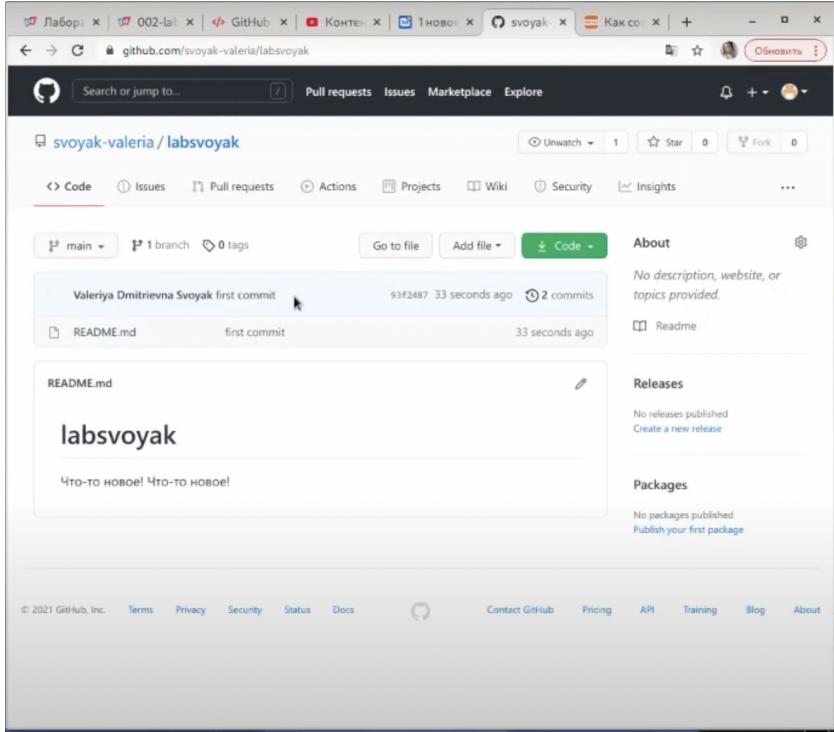
Изменения, которые не в индексе для коммита:
(используйте «git add <файл>», чтобы добавить файл в индекс)
(use "git restore <file>..." to discard changes in working directory)
изменено: README.md

нет изменений добавленных для коммита
(используйте «git add» и/или «git commit -a»)
vdsvoyak@dk8n80 ~/Laboratory/labsvoyak $
```

- Используем команду git commit -m “first commit”
- Используя команду git config –global user.name вводим почту и имя
- Устанавливаем коммит командой git commit -m “first commit”

```
vdsvoyak@dk8n80 ~/Laboratory/labsvoyak $ git add README.md
vdsvoyak@dk8n80 ~/Laboratory/labsvoyak $ git commit -m "first commit"
[main 93f2487] first commit
1 file changed, 3 insertions(+), 1 deletion(-)
vdsvoyak@dk8n80 ~/Laboratory/labsvoyak $ git push
Перечисление объектов: 5, готово.
Подсчет объектов: 100% (5/5), готово.
При сжатии изменений используется до 6 потоков
Сжатие объектов: 100% (2/2), готово.
Запись объектов: 100% (3/3), 284 bytes | 284.00 KiB/s, готово.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:svoyak-valeria/labsvoyak.git
 c347e1e..93f2487 main -> main
```

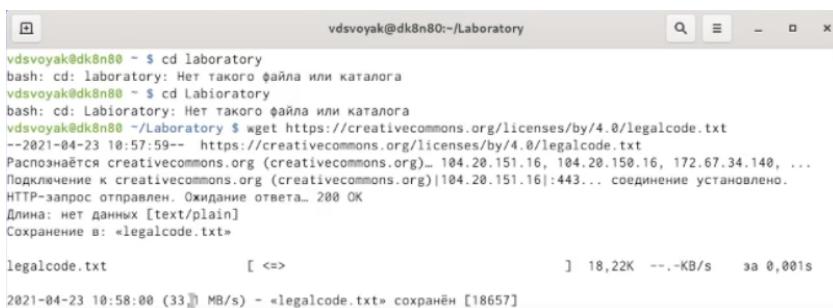
Добавляем файл README.md и с помощью команды git push передаем данные в гетхаб



The screenshot shows a GitHub repository page for 'svoyak-valeria / labsvoyak'. The 'Code' tab is selected, showing a single commit on the 'main' branch. The commit is titled 'Valeriya Dmitrievna Svojak first commit' and was made 33 seconds ago by '93f2487'. The commit message is 'Что-то новое! Что-то новое!'. The repository has 1 star, 0 forks, and 0 issues. It includes sections for About, Releases, and Packages.

Коммит появился

3.2 Теперь делаем первичную конфигурацию – создаем лицензию:



```
vdsvoyak@dk8n80:~/Laboratory
vdsvoyak@dk8n80 ~ $ cd laboratory
bash: cd: laboratory: Нет такого файла или каталога
vdsvoyak@dk8n80 ~ $ cd Labioratory
bash: cd: Labioratory: Нет такого файла или каталога
vdsvoyak@dk8n80 ~/Laboratory $ wget https://creativecommons.org/licenses/by/4.0/legalcode.txt
--2021-04-23 10:57:59-- https://creativecommons.org/licenses/by/4.0/legalcode.txt
Распознаётся creativecommons.org (creativecommons.org)...
Подключение к creativecommons.org (creativecommons.org)|104.20.151.16|:443... соединение установлено.
HTTP-запрос отправлен. Ожидание ответа... 200 OK
Длина: нет данных [text/plain]
Сохранение в: «legalcode.txt»

legalcode.txt          [ =>           ]  18,22K  --.-KB/s   за 0,001s

2021-04-23 10:58:00 (33.0 MB/s) - «legalcode.txt» сохранён [18657]
```

Вставляем файл лицензии - wget <https://creativecommons.org/licenses/by/4.0/legalcode.txt>

```
vdsvoyak@dk8n80:~/Laboratory
```

code.txt [<=>] 18,22K --.-KB/s за 0,001s

```
04-23 10:58:00 (33,1 MB/s) - «legalcode.txt» сохранён [18657]
```

```
yak@dk8n80 ~/Laboratory $ curl -L -s https://www.gitignore.io/api/list
-bitrix,a-frame,actionscript,ada
,advancedinstaller,adventuregamestudio,agda,al
aquareusii,altium,amplify,android,androidstudio
ar,anjuta,ansible,apachecordova,apachehadoop
ilder,appceleratortitanium,appcode,appcode+all,appcode+iml
gine,aptanastudio,arcانist,archive,archives
inuxpackages,aspnetcore,assembler,ate,atmelstudio
udio,automationstudio,autotools,autotools+strict
zurefunctions,backup,ballerina,basercms
,batch,bazaar,bazel,bitrise
x,bittorrent,blackbox,bloop,bluej
own,bower,brixcc,buck,c
ake,cakephp,cakephp2,cakephp3
ash,carthage,certificates,ceylon,cfwheels
ookbook,chocolatey,clean,clion,clion+all
+iml,closure,cloud9,cmake,cocoapods
2dx,cocoscreator,code,code-java,codeblocks
omposerstudio,codeigniter,codeio,codekit,codesniffer
escript,commonlisp,compodoc,composer,compressed
essedarchive,compression,conan,concrete5,coq
```

Добавляем шаблон игнорируемых файлов curl -L -s https://www.gitignore.io/api/list

```
vdsvoyak@dk8n80 ~/Laboratory/labsvoyak $ curl -L -s https://www.gitignore.io/api/c >> .gitignore
vdsvoyak@dk8n80 ~/Laboratory/labsvoyak $ git add .
vdsvoyak@dk8n80 ~/Laboratory/labsvoyak $
```

Скачиваем шаблон и добавим все файлы (git add .)

```
vdsvoyak@dk8n80 ~/Laboratory/labsvoyak $ git commit -am "second commit"
[main 366469a] second commit
 2 files changed, 455 insertions(+)
 create mode 100644 .gitignore
 create mode 100644 legalcode.txt
```

Устанавливаем второй коммит

```
vdsvoyak@dk8n80 ~/Laboratory/labsvoyak $ git push
Перечисление объектов: 5, готово.
Подсчет объектов: 100% (5/5), готово.
При сжатии изменений используется до 6 потоков
Сжатие объектов: 100% (4/4), готово.
Запись объектов: 100% (4/4), 6.44 KiB | 6.44 MiB/s, готово.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:svoyak-valeria/labsvoyak.git
  93f2487..366469a  main -> main
```

И отправляем его на гитхаб

Проверяем – они появились

3.3 Конфигурация git flow

```
vdsvojyak@dk8n80:~/Laboratory/labsvoyak
Подсчет объектов: 100% (5/5), готово.
При сжатии изменений используется до 6 потоков
Сжатие объектов: 100% (4/4), готово.
Запись объектов: 100% (4/4), 6.44 KiB | 6.44 MiB/s, готово.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:svoyak-valeria/labsvoyak.git
  93f2487..366469a main -> main
vdsvojyak@dk8n80 ~/Laboratory/labsvoyak $ git status
На ветке main
Ваша ветка обновлена в соответствии с «origin/main».

Нечего коммитить, нет изменений в рабочем каталоге
vdsvojyak@dk8n80 ~/Laboratory/labsvoyak $ git flow init

Which branch should be used for bringing forth production releases?
- main
Branch name for production releases: [main]
Branch name for "next release" development: [develop]

How to name your supporting branch prefixes?
Feature branches? [feature/]
Bugfix branches? [bugfix/]
Release branches? [release/]
Hotfix branches? [hotfix/]
Support branches? [support/]
Version tag prefix? [] v
Hooks and filters directory? [/afs/.dk.sci.pfu.edu.ru/home/v/d/vdsvojyak/Laboratory/labsvoyak/.git/hooks]
vdsvojyak@dk8n80 ~/Laboratory/labsvoyak $
```

Инициализируем git-flow: git flow init

Префикс для ярлыков установим в v.

The screenshot shows two terminal windows side-by-side. The left window displays the output of a git command, likely related to object counting or packing. The right window shows the execution of the 'git flow init' command. The user is prompted to choose a branch for production releases ('main'), which is selected. They are also asked about support branches, version tag prefixes, and hook/filter directories, with 'develop' and 'main' being chosen for support branches. The right terminal window also shows the current branch status with 'develop' and 'main' branches listed.

```
Подсчет объектов: 100% (5/5), готово.  
При сжатии изменений используется до 6 потоков  
Сжатие объектов: 100% (4/4), готово.  
Запись объектов: 100% (4/4), 6.44 KiB | 6.44 MiB/s, готово.  
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0  
To github.com:svoyak-valeria/labsvoyak.git  
 93f2487..366469a main -> main  
vdsvoyak@dk8n80 ~/Laboratory/labsvoyak $ git status  
На ветке main  
Ваша ветка обновлена в соответствии с «origin/main».  
  
Нечего коммитить, нет изменений в рабочем каталоге  
vdsvoyak@dk8n80 ~/Laboratory/labsvoyak $ git flow init  
  
Which branch should be used for bringing forth production releases?  
- main  
Branch name for production releases: [main]  
Branch name for "next release" development: [develop]  
  
How to name your supporting branch prefixes?  
Feature branches? [feature/]  
Bugfix branches? [bugfix/]  
Release branches? [release/]  
Hotfix branches? [hotfix/]  
Support branches? [support/]  
Version tag prefix? [] v  
Hooks and filters directory? [/afs/.dk.sci.pfu.edu.ru/home/v/d/vdsvoyak/Laboratory/labsvoyak/.git/hooks]  
vdsvoyak@dk8n80 ~/Laboratory/labsvoyak $ git branch
```

```
* develop  
  main  
lines 1-2/2 (END)
```

Проверьте, что Вы на ветке develop: git branch

The screenshot shows a terminal window displaying the output of the 'git branch' command. It lists the 'develop' and 'main' branches. The 'develop' branch is marked with an asterisk (*), indicating it is the current active branch.

```
vdsvoyak@dk8n80 ~/Laboratory/labsvoyak $ git flow release start 1.0.0  
Переключено на новую ветку «release/1.0.0»  
  
Summary of actions:  
- A new branch 'release/1.0.0' was created, based on 'develop'  
- You are now on branch 'release/1.0.0'  
  
Follow-up actions:  
- Bump the version number now!  
- Start committing last-minute fixes in preparing your release  
- When done, run:  
  
    git flow release finish '1.0.0'  
  
vdsvoyak@dk8n80 ~/Laboratory/labsvoyak $
```

Создадим релиз с версией 1.0.0 - git flow release start 1.0.0

The screenshot shows a terminal window where the 'git flow release start' command is being run. The command is followed by the version number '1.0.0'. The output indicates that a new branch 'release/1.0.0' has been created based on the 'develop' branch, and the user is now on this new branch.

```
vdsvoyak@dk8n80 ~/Laboratory/labsvoyak $ echo "1.0.0" >> VERSION  
vdsvoyak@dk8n80 ~/Laboratory/labsvoyak $ git add .
```

Запишем версию: echo "1.0.0" » VERSION

The screenshot shows a terminal window where the user runs the 'echo' command to write the version number '1.0.0' into a file named 'VERSION'. The command is followed by '>>' to append the text to the existing file. The output shows the command being run and the resulting commit message in the git history.

```
vdsvoyak@dk8n80 ~/Laboratory/labsvoyak $ git add .  
vdsvoyak@dk8n80 ~/Laboratory/labsvoyak $ git commit -am "chore(main): add version"  
[release/1.0.0 b846bb7] chore(main): add version  
1 file changed, 1 insertion(+)  
create mode 100644 VERSION
```

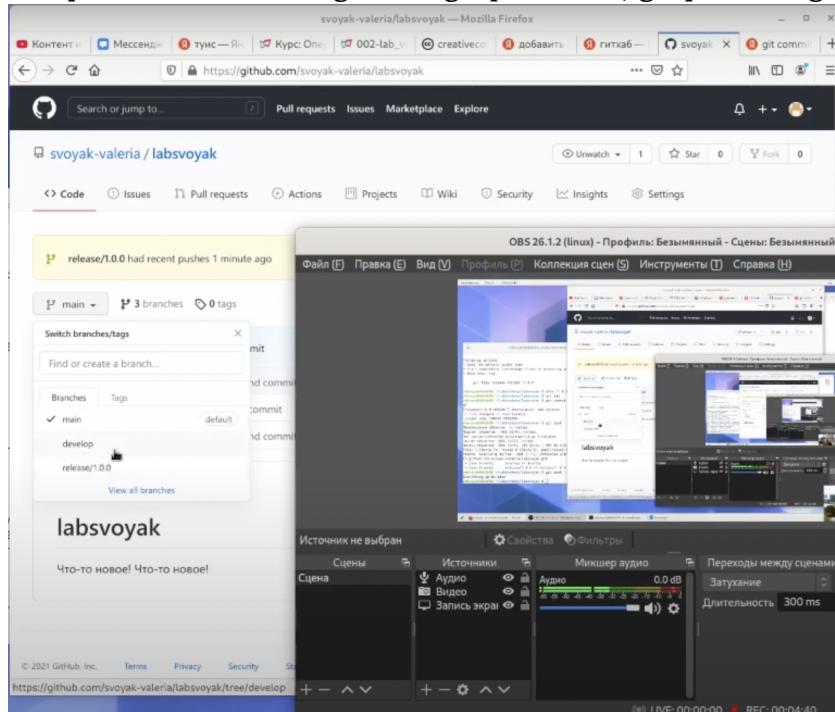
Добавим в индекс: git add . git commit -am 'chore(main): add version'

```
vdsvoyak@dk8n80 ~/Laboratory/labsvoyak $ git push --all
Перечисление объектов: 4, готово.
Подсчет объектов: 100% (4/4), готово.
При сжатии изменений используется до 6 потоков
Сжатие объектов: 100% (2/2), готово.
Запись объектов: 100% (3/3), 289 bytes | 289.00 KiB/s, готово.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:svoyak-valeria/labsvoyak.git
 * [new branch]      develop -> develop
 * [new branch]      release/1.0.0 -> release/1.0.0
vdsvoyak@dk8n80 ~/Laboratory/labsvoyak $ git push --tags
Everything up-to-date
```

Зальём релизную ветку в основную ветку: git flow release finish 1.0.0

```
vdsvoyak@dk8n80 ~/Laboratory/labsvoyak $ git push --all
Перечисление объектов: 4, готово.
Подсчет объектов: 100% (4/4), готово.
При сжатии изменений используется до 6 потоков
Сжатие объектов: 100% (2/2), готово.
Запись объектов: 100% (3/3), 289 bytes | 289.00 KiB/s, готово.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:svoyak-valeria/labsvoyak.git
 * [new branch]      develop -> develop
 * [new branch]      release/1.0.0 -> release/1.0.0
vdsvoyak@dk8n80 ~/Laboratory/labsvoyak $ git push --tags
Everything up-to-date
```

Отправим данные на github: git push –all; git push –tags



Создадим релиз на github (появились ветки и в них файлы)

Ответы на контрольные вопросы:

1. Система управления версиями - ПО 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.

2. Основным понятием VCS является репозиторий (repository) – специальное хранилище файлов и папок проекта, изменения в которых отслеживаются. В распоряжении разработчика имеется так называемая “рабочая копия” (working copy) проекта, с которой он непосредственно работает. Рабочую копию необходимо периодически синхронизировать с репозиторием, эта операция предполагает отправку в него изменений, которые пользователь внес в свою рабочую копию (такая операция называется commit) и актуализацию рабочей копии, в процессе которой к пользователю загружается последняя версия из репозитория (этот процесс носит название update).
3. Централизованная, предусматривающая концентрацию функций в рамках единой службы материально-технического обеспечения.

Децентрализованная, предусматривающая осуществление МТО бизнес единицами самостоятельно.

Смешанная форма организации материально-технического обеспечения, состоящая в комбинирования указанных выше способов.

6. Система контроля версий ГИТ представляет собой набор программ командной строки.

Доступ к ним можно получить из терминала посредством ввода команды git с различными опциями.

7. `git add`

Команда `git add` добавляет содержимое рабочей директории в индекс (staging area) для последующего коммита. По умолчанию `git commit` использует лишь

этот индекс, так что вы можете использовать git add для сборки слепка вашего следующего коммита.

git status

Команда git status показывает состояния файлов в рабочей директории и индексе: какие файлы изменены, но не добавлены в индекс; какие ожидают коммита в индексе. Вдобавок к этому выводятся подсказки о том, как изменить состояние файлов.

git commit

Команда git commit берёт все данные, добавленные в индекс с помощью git add, и сохраняет их слепок во внутренней базе данных, а затем сдвигает указатель текущей ветки на этот слепок.

git branch

Команда git branch — это своего рода “менеджер веток”. Она умеет перечислять ваши ветки, создавать новые, удалять и переименовывать их.

9. Ветка в Git’е — это просто «скользящий» указатель на один из коммитов. Когда вы создаёте новые коммиты, указатель ветки автоматически сдвигается вперёд, к вновь созданному коммиту.
10. Игнорируемые файлы обычно представляют собой файлы, специфичные для платформы, или автоматически созданные из сборочных систем. Файл .gitignore указывает, какие неотслеживаемые файлы должен игнорировать Git .

4 Выводы

Изучила идеологию и применение средств контроля версииGit (во 2 лабораторной)
Научилась оформлять отчёты с помощью легковесного языка разметки Markdown
(в лабораторной 3)