

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ №2

дисциплина: Операционные системы

Студент: Аветисян Давид

Группа: НПМбд-01-20

Ст. билет №: 1032201709

Москва

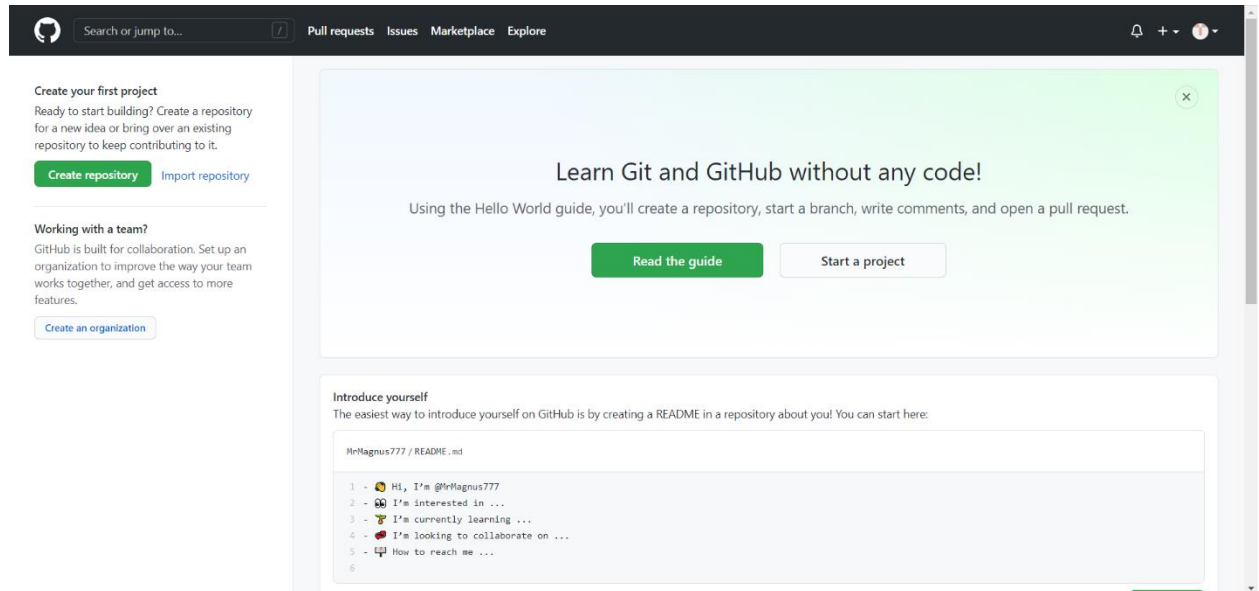
2021 г.

Цель работы:

Целью данной работы является изучение идеологии и применение средств контроля версий.

Ход работы:

Создаем учётную запись на <https://github.com> (рис. 1).



(рис. 1)

1) Настраиваем систему контроля версий git.

Синхронизируем учётную запись github с компьютером (рис. 2):

```
git config --global user.name "David Avetisyan"
```

```
git config --global user.email "David3777@yandex.ru"
```

Затем создаём новый ключ на github `ssh-keygen -C "David Avetisyan <David3777@yandex.ru>"` (рис. 3, рис. 4) и привязываем его к компьютеру через консоль (рис. 5).

```
Обзор Терминал C6, 1 мая 17:24 en
daavetisyan@localhost:~/tutorial

Файл Правка Вид Поиск Терминал Справка
[daavetisyan@localhost ~]$ git config --global user.name "David Avetisyan"
[daavetisyan@localhost ~]$ git config --global user.email <David3777@yandex.ru>
bash: синтаксическая ошибка рядом с неожиданным маркером «newline»
[daavetisyan@localhost ~]$ git config --global user.email <David3777@yandex.ru>
bash: синтаксическая ошибка рядом с неожиданным маркером «newline»
[daavetisyan@localhost ~]$ git config --global user.email <David3777@yandex.ru>
bash: синтаксическая ошибка рядом с неожиданным маркером «newline»
[daavetisyan@localhost ~]$ git config --global quotePath false
error: key does not contain a section: quotePath
[daavetisyan@localhost ~]$ git config --global user.email "David3777@yandex.ru"
[daavetisyan@localhost ~]$ git config --global quotePath false
error: key does not contain a section: quotePath
[daavetisyan@localhost ~]$ cd
[daavetisyan@localhost ~]$ mkdir tutorial
[daavetisyan@localhost ~]$ cd tutorial
[daavetisyan@localhost tutorial]$ git init
Инициализирован нусрой репозиторий Git в /home/daavetisyan/tutorial/.git/
[daavetisyan@localhost tutorial]$ echo 'hello world' > hello.txt
[daavetisyan@localhost tutorial]$ git add hello.txt
[daavetisyan@localhost tutorial]$ git commit -am 'Новый файл'
[master (корневой коммит) 312daeb] Новый файл
1 file changed, 1 insertion(+)
 create mode 100644 hello.txt
[daavetisyan@localhost tutorial]$ git status
На ветке master
ничего коммитить, нет изменений в рабочем каталоге
[daavetisyan@localhost tutorial]$ curl -L -s https://www.gitignore.io/api/list
lc,lc-bitrix,a-frame,actionsript,ada
adobe,advancedinstaller,adventuregamestudio,agda,al
alteraquartusii,altium,amplify,android,androidstudio
angular,anjuta,ansible,apachecordova,apachehadoop
appbuilder,appcelerator titanium,appcode,appcode+all,appcode+iml
appengine,aptnastudio,arcanist,archive,archives
archlinuxpackages,aspnecore,assembler,ate,atmelstudio
ats,audio,automationstudio,autotools,autotools+strict
awr,azurefunctions,backup,ballerina,basercms
basic,batch,bazaar,bazel,bitrise
bitrix,bitrise,blackbox,black,bluz
```

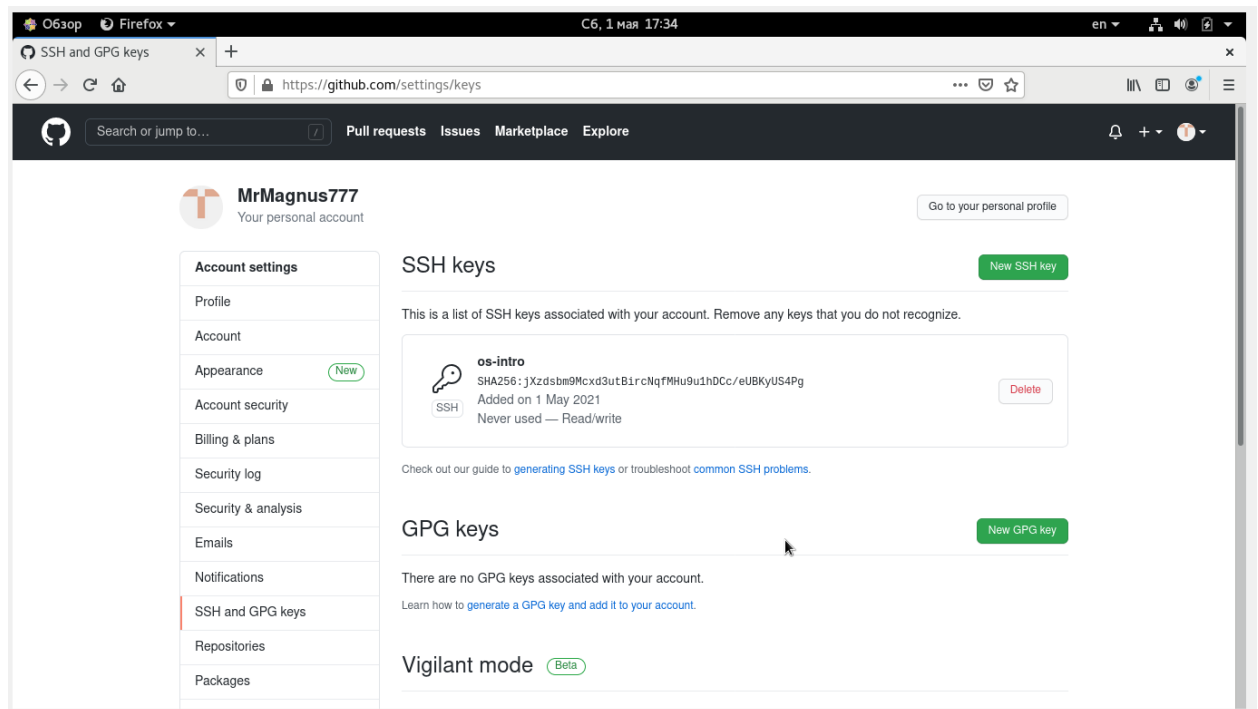
(рис. 2)

```
Обзор Терминал C6, 1 мая 17:42 en
daavetisyan@localhost:~/tutorial

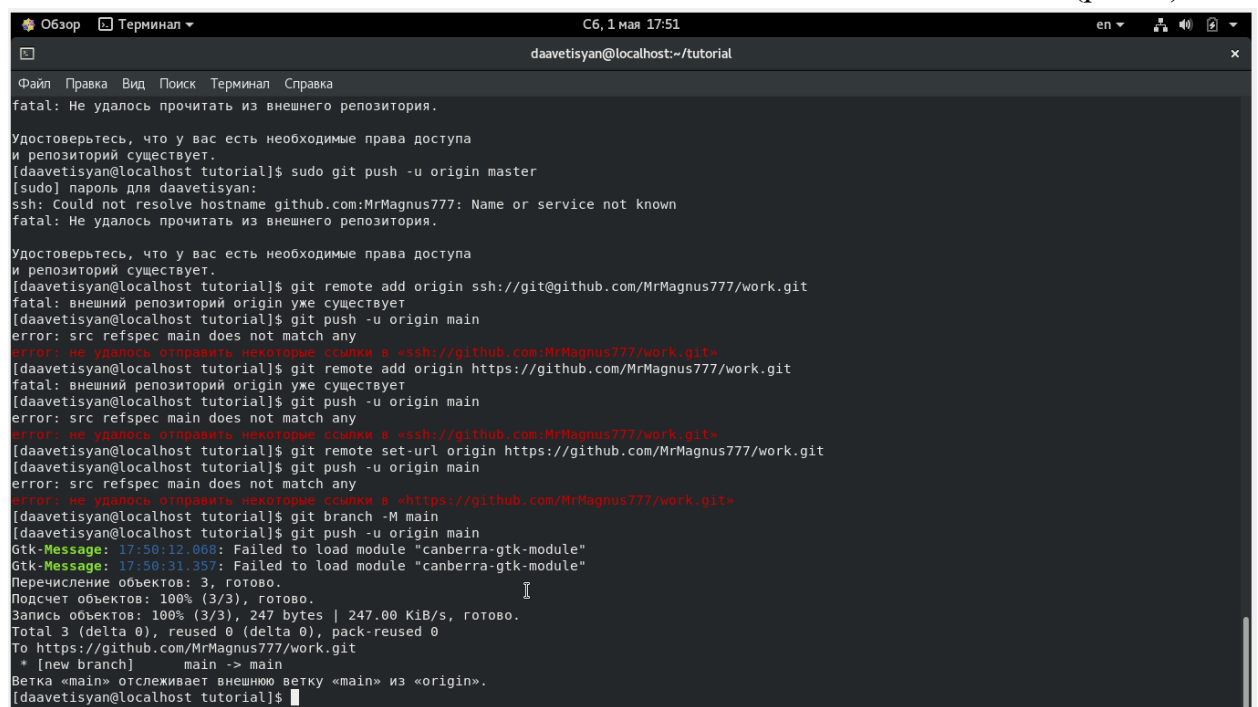
Файл Правка Вид Поиск Терминал Справка
[daavetisyan@localhost tutorial]$ curl -L -s https://www.gitignore.io/api/c >> .gitignore
[daavetisyan@localhost tutorial]$ curl -L -s https://www.gitignore.io/api/c++ >> .gitignore
[daavetisyan@localhost tutorial]$ ssh-keygen -C "David Avetisyan <David3777@yandex.ru>"
Generating public/private rsa key pair.
Enter file in which to save the key (/home/daavetisyan/.ssh/id_rsa):
Created directory '/home/daavetisyan/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/daavetisyan/.ssh/id_rsa.
Your public key has been saved in /home/daavetisyan/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:jXzdsbm9Mcxd3utBircNqfMhu9u1hDCc/eUBKyUS4Pg David Avetisyan <David3777@yandex.ru>
The key's randomart image is:
+---[RSA 3072]---+
  .   .   .
  o   .   . o.
  .. .   .o+
  ES o*o+o+o
  .   .o=*
  .   .+oB
  .o.*.o
  .o==B
+-----[SHA256]-----
[daavetisyan@localhost tutorial]$ cat ~/.ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQGCcNcCLZb6f1rZv9aiihKXft5YfjbrCV3D0Ywcl7bW0sR1EzAHGrogB/VoM65B1M/8wxWfCKWoNynWe0F+dX/Za8STLq2KfvNcP5j3633urtVv
CjI07xa18Rhd6Ye7jssx4w6g15c1oquvzk4RmVEvZ1lR6Kq9KI/Nx2qgo4JFsvGrBBR0HeSZXn43Pxj/u29svrP4oo7KgFQDhGnRxQ6bQ7nm67m6o8262aTkLCMGK/rKrdAnti2p6MyMBUa74uv91
6xxdxkG70ustFxyMnwS5SL3QXS3S+cf35JvW4kHtfrTI+cgLIGRHNCf7mclBXcd0GyDaxLy+xy155UsJPBg/ee0614eMRIZR8xAmUND0m/LBhvl2YPhfLVduTIzS7Moc/7kQ68IJwPX5W88r/yh1Co
mh96y44UXS9qOURr910vtaNb9hIQ+CBiIPmEfUMktgkLIYizIEKHfYERYqAHxJwR+je9K0t6eV3Qq4PQAHFxbylZQTufbqKPqd9sgeHc= David Avetisyan <David3777@yandex.ru>
[daavetisyan@localhost tutorial]$ git remote add origin
использование: git remote add [<опции>] <имя> <url>

-f, --fetch            извлечь внешние ветки
--tags                импортировать все метки и ассоциированные объекты при извлечении
                     или не извлекать метки вообще (--no-tags)
-t, --track <ветка>   отслеживаемые ветки
-m, --master <ветка>  мастер ветка
```

(рис. 3)



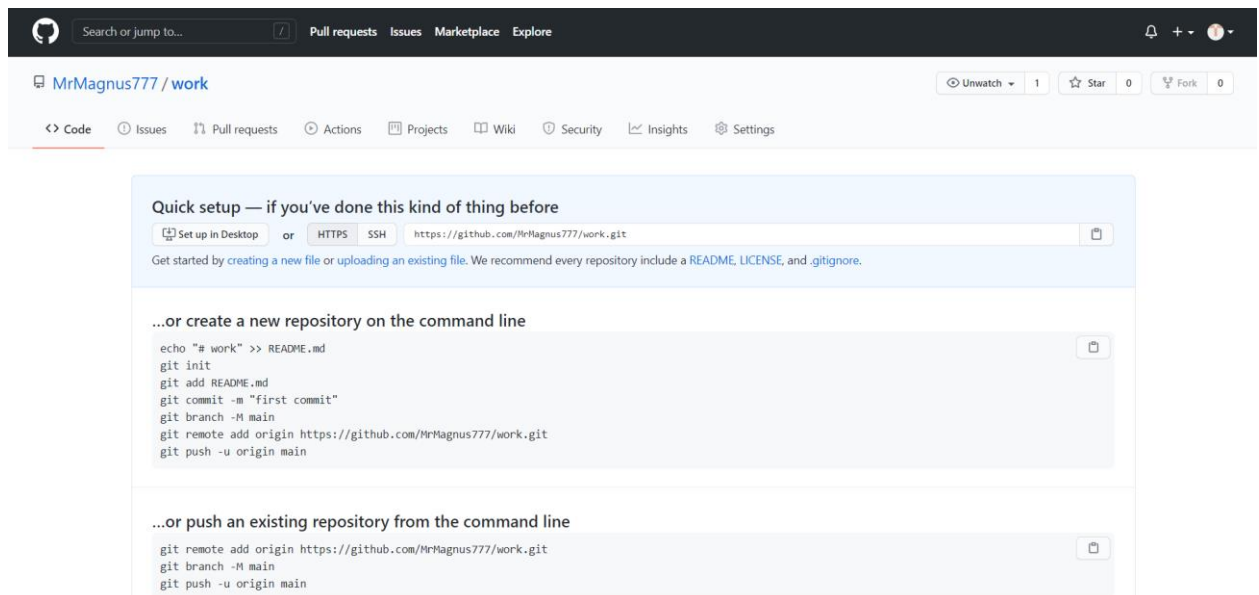
(рис. 4)



(рис. 5)

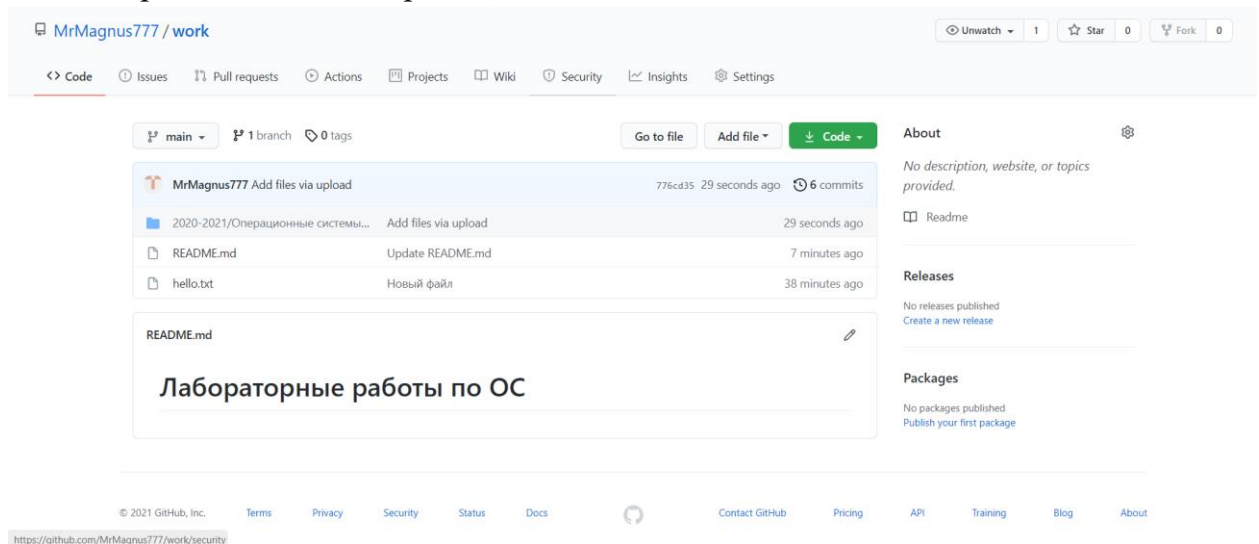
2) Созданием и подключаем репозиторий к github.

На сайте заходим в «repository» и создаём новый репозиторий под названием work. Переносим его на наш компьютер (рис. 6).



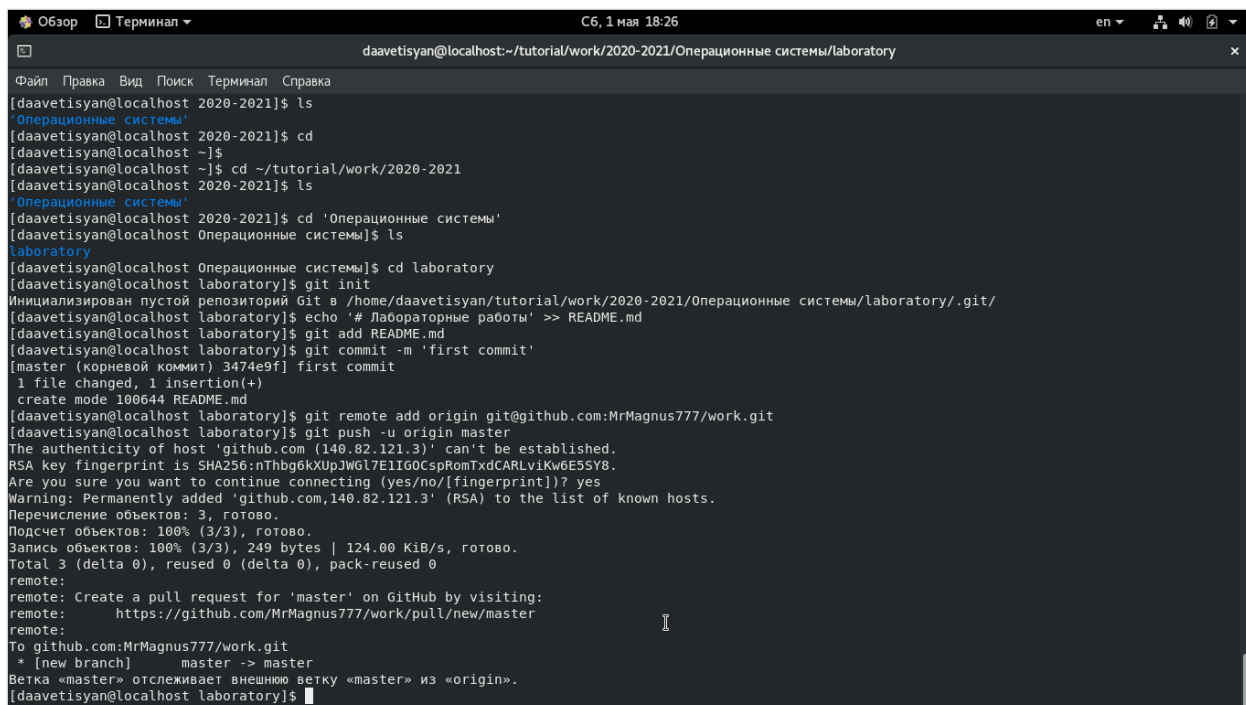
(рис. 6)

Создаем рабочий каталог (рис. 7)



(рис. 7)

Добавляем первый commit и выкладываем на github. Для того, чтобы правильно разместить первый коммит, необходимо добавить команду `git add .`, после этого с помощью команды `git commit -m "first commit"` выкладываем коммит. Сохраняем первый коммит, используя команду `git push` (рис. 8).



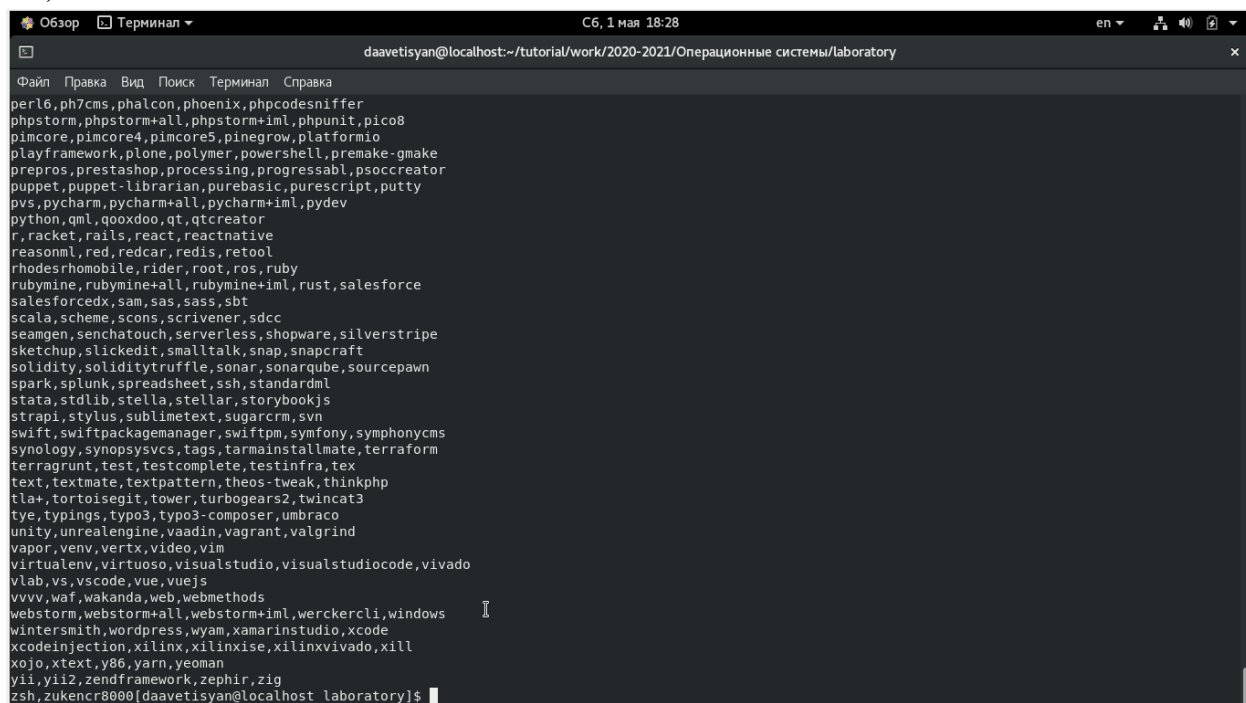
```
Обзор Терминал C6, 1 мая 18:26 en
daavetisyan@localhost:~/tutorial/work/2020-2021/Операционные системы/laboratory

Файл Правка Вид Поиск Терминал Справка
[daavetisyan@localhost 2020-2021]$ ls
'Операционные системы'
[daavetisyan@localhost 2020-2021]$ cd
[daavetisyan@localhost ~]$ cd ~/tutorial/work/2020-2021
[daavetisyan@localhost 2020-2021]$ ls
'Операционные системы'
[daavetisyan@localhost 2020-2021]$ cd 'Операционные системы'
[daavetisyan@localhost Операционные системы]$ ls
laboratory
[daavetisyan@localhost Операционные системы]$ cd laboratory
[daavetisyan@localhost laboratory]$ git init
Инициализирован пустой репозиторий Git в /home/daavetisyan/tutorial/work/2020-2021/Операционные системы/laboratory/.git/
[daavetisyan@localhost laboratory]$ echo '# Лабораторные работы' >> README.md
[daavetisyan@localhost laboratory]$ git add README.md
[daavetisyan@localhost laboratory]$ git commit -m 'first commit'
[master (корневой коммит) 3474e9f] first commit
1 file changed, 1 insertion(+)
 create mode 100644 README.md
[daavetisyan@localhost laboratory]$ git remote add origin git@github.com:MrMagnus777/work.git
[daavetisyan@localhost laboratory]$ git push -u origin master
The authenticity of host 'github.com (140.82.121.3)' can't be established.
RSA key fingerprint is SHA256:nThbg6kXUpJWGL7E1IG0CsPROMTxdCARLviKw6E5SY8.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'github.com,140.82.121.3' (RSA) to the list of known hosts.
Перечисление объектов: 3, готово.
Подсчет объектов: 100% (3/3), готово.
Запись объектов: 100% (3/3), 249 bytes | 124.00 KiB/s, готово.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
remote: Create a pull request for 'master' on GitHub by visiting:
remote:   https://github.com/MrMagnus777/work/pull/new/master
remote:
To github.com:MrMagnus777/work.git
 * [new branch]      master -> master
Ветка «master» отслеживает внешнюю ветку «master» из «origin».
[daavetisyan@localhost laboratory]$
```

(рис. 8)

3) Первичная конфигурация.

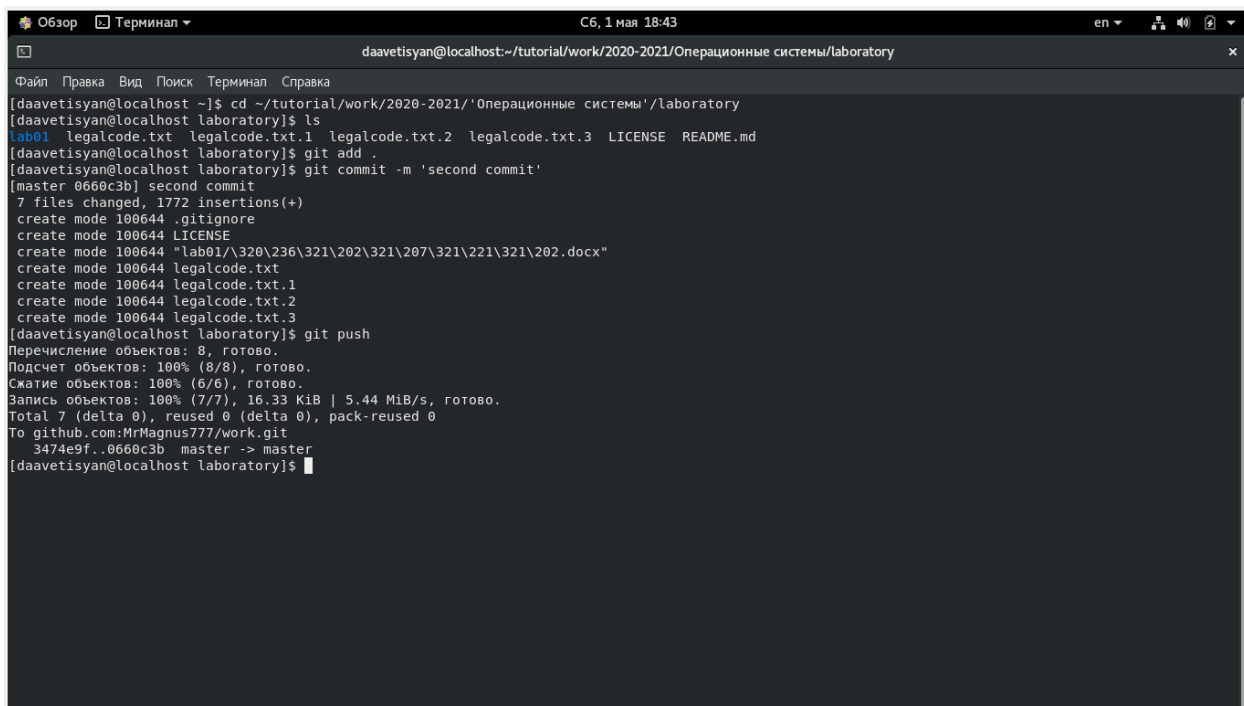
Добавляем файл лицензии. Добавляем шаблон игнорируемых файлов. Просматриваем список имеющихся шаблонов (рис. 9). Скачиваем шаблон (например, для C) и выполняем коммит. Отправляем на github (команда git push) (рис. 10).



```
Обзор Терминал C6, 1 мая 18:28 en
daavetisyan@localhost:~/tutorial/work/2020-2021/Операционные системы/laboratory

Файл Правка Вид Поиск Терминал Справка
perl6,ph7cms,phalcon,phoenix,phpcodesniffer
phpstorm,phpstorm+all,phpstorm+iml,phpunit,pico8
pimcore,pimcore4,pimcore5,pinegrow,platformio
playframework,plone,polymer,powershell,premake-gmake
prepros,prestashop,processing,progressabl,psoccreator
puppet,puppet-librarian,purebasic,purescript,putty
pvs,pycharm,pycharm+all,pycharm+iml,pydev
python,qml,goxdoq,qt,qtcreator
r,racket,rails,react,reactnative
reasonml,red,redcar,redis,retool
rhodesrhomobile,rider,root,ros,ruby
rubymine,rubymine+all,rubymine+iml,rust,salesforce
salesforcedx,sam,sas,sass,sbt
scala,scheme,scons,scrivener,sdcc
seamgen,senchatouch,serverless,shopware,silverstripe
sketchup,slickedit,smalltalk,snap,snapcraft
solidity,soliditytruffle,sonar,sonarqube,sourcepaw
spark,splunk,spreadsheet,ssh,standardml
stata,stdlib,stellar,stellar,storybookjs
strapi,stylus,sublimetext,sugarcrm,svn
swift,swiftpackagemanager,swiftpm,symfony,symphonycms
synology,synopsysvcs,tags,tarminstallmate,terraform
terragrunt,test,testcomplete,testinfra,txt
text,textmate,textpattern,theos-tweak,thinkphp
tla+,tortoisegit,tower,turbogears2,twincat3
tye,typings,typo3,typo3-composer,umbraco
unity,unrealengine,vaadin,vagrant,vaigrind
vapor,venv,vertx,video,vim
virtualenv,virtuoso,visualstudio,visualstudiocode,vivado
vlab,vs,vscode,vue,vuejs
vvvv,waf,wakanda,web,webmethods
webstorm,webstorm+all,webstorm+iml,werckercli,windows
wintersmith,wordpress,wyam,xamarinstudio,xcode
xcodeinjection,xilinx,xilinxise,xilinxvivado,xill
xojo,xtext,y86,yarn,yeoman
yii,yii2,zendframework,zephir,zig
zsh,zukencr8000[daavetisyan@localhost laboratory]$
```

(рис. 9)

A screenshot of a terminal window titled "Терминал" with a status bar showing "С6, 1 мая 18:43". The terminal shows a user named "daavetisyan" at a "localhost" in a directory "tutorial/work/2020-2021/Операционные системы/laboratory". The user runs a series of commands: "cd ~/tutorial/work/2020-2021/Операционные системы/laboratory", "ls", "git add .", "git commit -m 'second commit'", and "git push". The output shows the creation of several files (legalcode.txt, legalcode.txt.1, legalcode.txt.2, legalcode.txt.3, LICENSE, README.md) and the successful push to the "master" branch. The terminal window has a menu bar with "Файл", "Правка", "Вид", "Поиск", "Терминал", and "Справка".

(рис. 10)

4) Работаем с конфигурацией git-flow.

У нас не получилось установить git-flow, так как root этого не допустил. В связи с этим дальнейшие действия выполнить невозможно.

Контрольные вопросы:

1) Система контроля версий Git представляет собой набор программ командной строки. Доступ к ним можно получить из терминала посредством ввода команды git с различными опциями. Системы контроля версий (Version Control System, VCS) применяются при работе нескольких человек над одним проектом.

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

Системы контроля версий также могут обеспечивать дополнительные, более гибкие функциональные возможности. Например, они могут

поддерживать работу с несколькими версиями одного файла, сохраняя общую историю изменений до точки ветвления версий и собственные истории изменений каждой ветви. Кроме того, обычно доступна информация о том, кто из участников, когда и какие изменения вносил. Обычно такого рода информация хранится в журнале изменений, доступ к которому можно ограничить.

3) Централизованные системы — это системы, которые используют архитектуру клиент / сервер, где один или несколько клиентских узлов напрямую подключены к центральному серверу. **Пример - Wikipedia.**

В децентрализованных системах каждый узел принимает свое собственное решение. Конечное поведение системы является совокупностью решений отдельных узлов. **Пример — Bitcoin.**

В классических системах контроля версий используется централизованная модель, предполагающая наличие единого репозитория для хранения файлов. Выполнение большинства функций по управлению версиями осуществляется специальным сервером.

4) Создадим локальный репозиторий. Сначала сделаем предварительную конфигурацию, указав имя и email владельца репозитория:

```
git config --global user.name "Имя Фамилия"
git config --global user.email "work@mail"
и настроив utf-8 в выводе сообщений git:
git config --global quotepath false
```

Для инициализации локального репозитория, расположенного, например, в каталоге ~/tutorial, необходимо ввести в командной строке:

```
cd
mkdir tutorial
cd tutorial
git init
```

5) Для последующей идентификации пользователя на сервере репозитория необходимо сгенерировать пару ключей (приватный и открытый):

```
ssh-keygen -C "Имя Фамилия <work@mail>"
```

Ключи сохраняться в каталоге ~/.ssh/.

Скопировав из локальной консоли ключ в буфер обмена

cat ~/.ssh/id_rsa.pub | xclip -sel clip вставляем ключ в появившееся на сайте поле.

6) 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.

7) Основные команды 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 изменения: 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 стекущим 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) Использование git при работе с локальными репозиториями (добавления текстового документа в локальный репозиторий):

```
git add hello.txt
```

```
git commit -am 'Новый файл'
```

9) Проблемы, которые решают ветки git:

- нужно постоянно создавать архивы с рабочим кодом
- сложно "переключаться" между архивами
- сложно перетаскивать изменения между архивами
- легко что-то напутать или потерять

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 списки меняющихся шаблонов: curl -L -s <https://www.gitignore.io/api/list>

Затем скачать шаблон, например, для С и С++

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

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

Вывод:

Я изучил идеологию и применение контроля версий.