

Отчёт по лабораторной работе №3

Управление версиями

Калистратова Ксения Евгеньевна

Содержание

1	Цель работы	4
2	Задание	5
3	Выполнение лабораторной работы	6
4	Выводы	13
5	Ответы на контрольные вопросы	14
6	Библиография	18

Список иллюстраций

3.1	Создание репозитория	6
3.2	Инициализация репозитория	7
3.3	Создание SSH-ключа	7
3.4	Загрузка файлов	8
3.5	Отправка в сетевой репозиторий по SSH	9
3.6	Инициализация git-flow и начало релиза	10
3.7	Завершение релиза и отправка изменений в сетевой репозиторий	11
3.8	Объединение веток в сетевом репозитории	12

1 Цель работы

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

2 Задание

1. Создать учетную запись на github.com.
2. Настроить репозиторий и организовать доступ по ssh.
3. Изучить механизм управления версиями.

3 Выполнение лабораторной работы

1. Создаем учетную запись на github.com и репозиторий.

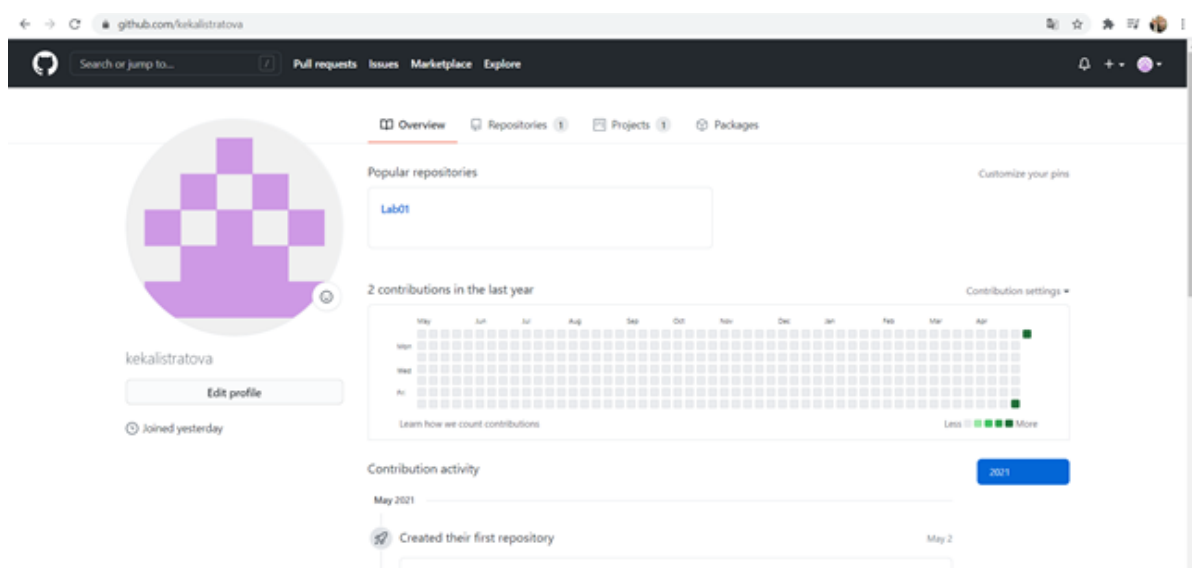


Рис. 3.1: Создание репозитория

2. Инициализируем локальный репозиторий и создаю в нем файл README.md.

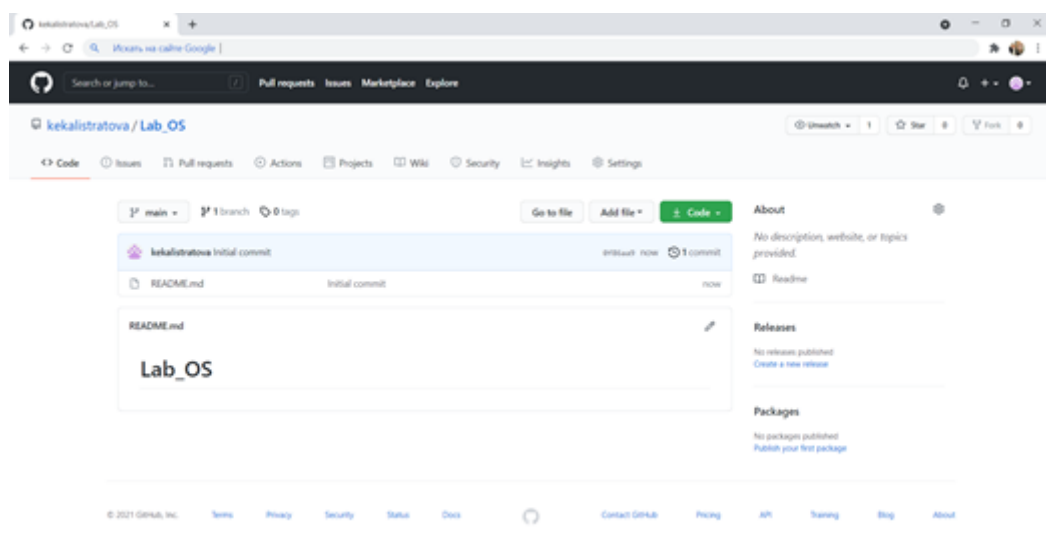


Рис. 3.2: Инициализация репозитория

3. Создаем SSH-ключ и прописываем его в настройках на github.com.

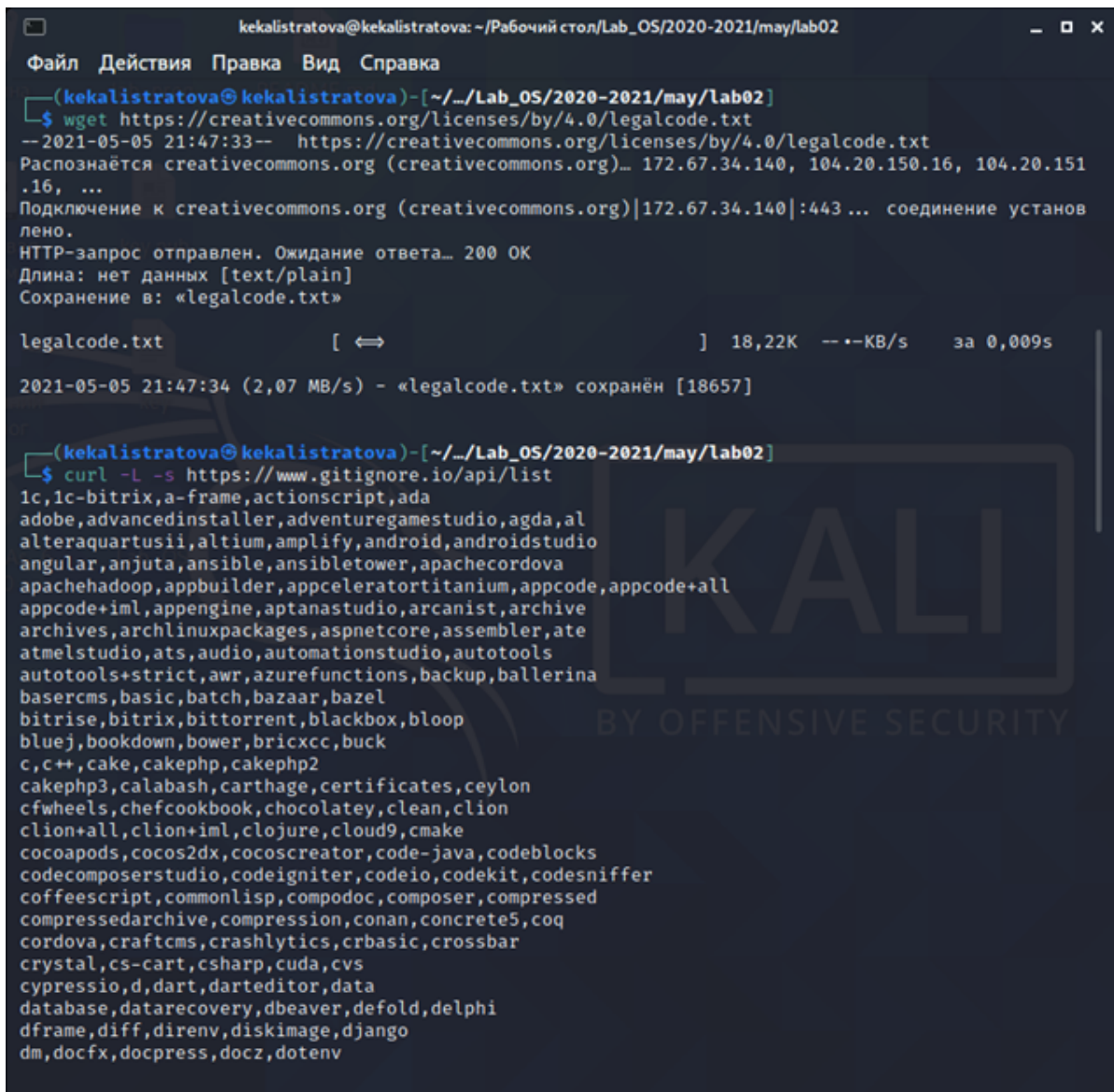
```

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]-----+
| ^B+00                                     |
| X*E+0                                     |
| O%.O..                                    |
| *+= .. .                                  |
| ++ ... . S                               |
| O+= ...                                   |
| O+ +. O                                   |
| ... O                                     |
| ..                                       |
+---[SHA256]-----+

```

Рис. 3.3: Создание SSH-ключа

4. Загружаем файлы лицензионного соглашения и gitignore. Отправляем все файлы в сетевой репозиторий.



```
kekalistratova@kekalistratova: ~/Рабочий стол/Lab_OS/2020-2021/may/lab02
Файл Действия Правка Вид Справка
(kekalistratova@kekalistratova)-[~/./Lab_OS/2020-2021/may/lab02]
$ wget https://creativecommons.org/licenses/by/4.0/legalcode.txt
--2021-05-05 21:47:33-- https://creativecommons.org/licenses/by/4.0/legalcode.txt
Распознаётся creativecommons.org (creativecommons.org)... 172.67.34.140, 104.20.150.16, 104.20.151.16, ...
Подключение к creativecommons.org (creativecommons.org)[172.67.34.140]: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 0,009s
2021-05-05 21:47:34 (2,07 MB/s) - «legalcode.txt» сохранён [18657]

(kekalistratova@kekalistratova)-[~/./Lab_OS/2020-2021/may/lab02]
$ curl -L -s https://www.gitignore.io/api/list
1c,1c-bitrix,a-frame,actionsript,ada
adobe,advancedinstaller,adventuregamestudio,agda,al
alteraquartusii,altium,amplify,android,androidstudio
angular,anjuta,ansible,ansibletower,apachecordova
apachehadoop,appbuilder,appcelerator titanium,appcode,appcode+all
appcode+iml,appengine,aptanastudio,arcanist,archive
archives,archlinuxpackages,aspnetcore,assembler,ate
atmelstudio,ats,audio,automationstudio,autotools
autotools+strict,awr,azurefunctions,backup,ballerina
basercms,basic,batch,bazaar,bazel
bitrise,bitrix,bittorrent,blackbox,bloop
bluej,bookdown,bower,bricxcc,buck
c,c++,cake,cakephp,cakephp2
cakephp3,calabash,carthage,certificates,ceylon
cfwheels,chefcookbook,chocolatey,clean,clion
clion+all,clion+iml,clojure,cloud9,cmake
cocoapods,cocos2dx,cocoscreator,code-java,codeblocks
codecomposerstudio,codeigniter,codeio,codekit,codesniffer
coffeescript,commonlisp,compodoc,composer,compressed
compressedarchive,compression,conan,concrete5,coq
cordova,craftcms,crashlytics,crbasic,crossbar
crystal,cs-cart,csharp,cuda,cvs
cypressio,d,dart,darteditor,data
database,datarecovery,dbeaver,defold,delphi
dframe,diff,direnv,diskimage,django
dm,docfx,docpress,docz,dotenv
```

Рис. 3.4: Загрузка файлов


```
(kekalistratova@kekalistratova)~[/../Lab_OS/2020-2021/may/lab02]
$ curl -L -s https://www.gitignore.io/api/c >> .gitignore

(kekalistratova@kekalistratova)~[/../Lab_OS/2020-2021/may/lab02]
$ git add .

(kekalistratova@kekalistratova)~[/../Lab_OS/2020-2021/may/lab02]
$ git commit -a
Отмена коммита из-за пустого сообщения коммита.

(kekalistratova@kekalistratova)~[/../Lab_OS/2020-2021/may/lab02]
$ [[200~git push
zsh: bad pattern: ^[[200~git 1 x

(kekalistratova@kekalistratova)~[/../Lab_OS/2020-2021/may/lab02]
$ git commit -am "Create template for C" 1 x

[main 47f43fc] Create template for C
2 files changed, 455 insertions(+)
create mode 100644 2020-2021/may/lab02/.gitignore
create mode 100644 2020-2021/may/lab02/legalcode.txt

(kekalistratova@kekalistratova)~[/../Lab_OS/2020-2021/may/lab02]
$ git push
Username for 'https://github.com': kekalistratova
Password for 'https://kekalistratova@github.com':
Перечисление объектов: 11, готово.
Подсчет объектов: 100% (11/11), готово.
При сжатии изменений используется до 2 потоков
Сжатие объектов: 100% (5/5), готово.
Запись объектов: 100% (7/7), 6.60 KiB | 6.60 MiB/s, готово.
Total 7 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/kekalistratova/Lab_OS.git
16cbf18..47f43fc main -> main
```

Рис. 3.5: Отправка в сетевой репозиторий по SSH

5. Использование системы управления версиями. Создаем ветку, начинаем и завершаем в ней релиз.

```
kekalistratova@kekalistratova: ~/Рабочий стол
Файл Действия Правка Вид Справка
(kekalistratova@kekalistratova)~[~/Рабочий стол]
$ 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? []
Hooks and filters directory? [/home/kekalistratova/Рабочий стол/.git/hooks]

(kekalistratova@kekalistratova)~[~/Рабочий стол]
$ git branch
* develop
main

(kekalistratova@kekalistratova)~[~/Рабочий стол]
$ git flow release start 1.0.
fatal: «release/1.0.» не является действительным именем ветки.
Fatal: Could not create release branch 'release/1.0.'.

(kekalistratova@kekalistratova)~[~/Рабочий стол]
$ 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'

(kekalistratova@kekalistratova)~[~/Рабочий стол]
$ echo "1.0.0" >> VERSION

(kekalistratova@kekalistratova)~[~/Рабочий стол]
$ git add .
```

Рис. 3.6: Инициализация git-flow и начало релиза

```
kekalistratova@kekalistratova: ~/Рабочий стол/Lab_OS/2020-2021/may/lab02
Файл Действия Правка Вид Справка
(kekalistratova@kekalistratova)-[~/Lab_OS/2020-2021/may/lab02]
$ git commit -am 'chore(main): add version'
[release/1.0.0 d98df70] chore(main): add version
1 file changed, 1 insertion(+)

(kekalistratova@kekalistratova)-[~/Lab_OS/2020-2021/may/lab02]
$ git flow release finish 1.0.0
Переключено на ветку «main»
Ваша ветка обновлена в соответствии с «origin/main».
Merge made by the 'recursive' strategy.
2020-2021/may/lab02/VERSION | 2 ++
1 file changed, 2 insertions(+)
create mode 100644 2020-2021/may/lab02/VERSION
Уже на «main»
Ваша ветка опережает «origin/main» на 3 коммита.
(используйте «git push», чтобы опубликовать ваши локальные коммиты)
Переключено на ветку «develop»
Merge made by the 'recursive' strategy.
2020-2021/may/lab02/VERSION | 1 +
1 file changed, 1 insertion(+)
Ветка release/1.0.0 удалена (была d98df70).

Summary of actions:
- Release branch 'release/1.0.0' has been merged into 'main'
- The release was tagged '1.0.0'
- Release tag '1.0.0' has been back-merged into 'develop'
- Release branch 'release/1.0.0' has been locally deleted
- You are now on branch 'develop'
```

Рис. 3.7: Завершение релиза и отправка изменений в сетевой репозиторий

6. Выполним объединение веток.

```
kekalistratova@kekalistratova: ~/Рабочий стол/Lab_OS/2020-2021/may/lab02
Файл Действия Правка Вид Справка
(kekalistratova@kekalistratova)-[~/Lab_OS/2020-2021/may/lab02]
$ git push --all
Username for 'https://github.com': kekalistratova
Password for 'https://kekalistratova@github.com':
Перечисление объектов: 18, готово.
Подсчет объектов: 100% (18/18), готово.
При сжатии изменений используется до 2 потоков
Сжатие объектов: 100% (8/8), готово.
Запись объектов: 100% (14/14), 1.22 KiB | 1.22 MiB/s, готово.
Total 14 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), done.
To https://github.com/kekalistratova/Lab_OS.git
  47f43fc..1af7d21  main -> main
* [new branch]      develop -> develop

(kekalistratova@kekalistratova)-[~/Lab_OS/2020-2021/may/lab02]
$ git push --tags
Username for 'https://github.com': kekalistratova
Password for 'https://kekalistratova@github.com':
Перечисление объектов: 1, готово.
Подсчет объектов: 100% (1/1), готово.
Запись объектов: 100% (1/1), 160 bytes | 160.00 KiB/s, готово.
Total 1 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/kekalistratova/Lab_OS.git
* [new tag]         1.0.0 -> 1.0.0
```

Рис. 3.8: Объединение веток в сетевом репозитории

4 Выводы

Мы приобрели практические навыки работы с системой контроля версий git и создали свой репозиторий.

5 Ответы на 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 core.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 сохраняются 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 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 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 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
```

## 6 Библиография

1. [https://esystem.rudn.ru/pluginfile.php/1142090/mod\\_resource/content/2/008-lab\\_shell\\_prog\\_1.pdf](https://esystem.rudn.ru/pluginfile.php/1142090/mod_resource/content/2/008-lab_shell_prog_1.pdf)
2. Кулябов Д.С. Операционные системы: лабораторные работы: учебное пособие / Д.С. Кулябов, М.Н. Геворкян, А.В. Королькова, А.В. Демидова. — М. : Изд-во РУДН, 2016. — 117 с. — ISBN 978-5-209-07626-1 : 139.13; То же [Электронный ресурс]. — URL: <http://lib.rudn.ru/MegaPro2/Download/MObject/6118>.
3. Робачевский А.М. Операционная система UNIX [текст] : Учебное пособие / А.М. Робачевский, С.А. Немнюгин, О.Л. Стесик. — 2-е изд., перераб. и доп. — СПб. : БХВ-Петербург, 2005, 2010. — 656 с. : ил. — ISBN 5-94157-538-6 : 164.56. (ЕТ 60)
4. Таненбаум Эндрю. Современные операционные системы [Текст] / Э. Таненбаум. — 2-е изд. — СПб. : Питер, 2006. — 1038 с. : ил. — (Классика Computer Science). — ISBN 5-318-00299-4 : 446.05. (ЕТ 50)