Отчет по лабораторной работе 2

Управление версиями

Шалыгин Георгий Эдуардович

# Цель работы

Изучить идеологию и применение средств контроля версий.

Объект исследования: Система контроля версий Git в связке с github.

Предмет исследования: процесс использования git и git flow при разработке ПО.

# Задание

Настройка git, подключение репозитория к github, создать первичную конфигурацию, воспользоваться конфигурацией git-flow.

# Техническое обеспечение:

* Характеристики техники: AMD Ryzen 5 3500U 2.1 GHz, 8 GB оперативной памяти, 50 GB свободного места на жёстком диске;
* ОС Windows 10 Home
* Git 2.31.1
* Google Chrome 91.0.4472.19

# Условные обозначения и термины:

* Понятия Git и GitHib
* **Git или Гит** — система контроля и управления версиями файлов.
* **GitHub или Гитхаб** — веб-сервис для размещения репозиториев и совместной разработки проектов.
* **Репозиторий Git** — каталог файловой системы, в котором находятся: файлы конфигурации, файлы журналов операций, выполняемых над репозиторием, индекс расположения файлов и хранилище, содержащее сами контролируемые файлы.
* **Локальный репозиторий** — репозиторий, расположенный на локальном компьютере разработчика в каталоге. Именно в нём происходит разработка и фиксация изменений, которые отправляются на удалённый репозиторий.
* **Удалённый репозиторий** — репозиторий, находящийся на удалённом сервере. Это общий репозиторий, в который приходят все изменения и из которого забираются все обновления.
* **Форк (Fork)** — копия репозитория. Его также можно рассматривать как внешнюю ветку для текущего репозитория. Копия вашего открытого репозитория на Гитхабе может быть сделана любым пользователем, после чего он может прислать изменения в ваш репозиторий через пулреквест.
* **Обновиться из апстрима** — обновить свою локальную версию форка до последней версии основного репозитория, от которого сделан форк.
* **Обновиться из ориджина** — обновить свою локальную версию репозитория до последней удалённой версии этого репозитория.
* **Клонирование (Clone)** — скачивание репозитория с удалённого сервера на локальный компьютер в определённый каталог для дальнейшей работы с этим каталогом как с репозиторием.
* **Ветка (Branch)** — это параллельная версия репозитория. Она включена в этот репозиторий, но не влияет на главную версию, тем самым позволяя свободно работать в параллельной. Когда вы внесли нужные изменения, то вы можете объединить их с главной версией.
* **Мастер (Master)** — главная или основная ветка репозитория.
* **Коммит (Commit)** — фиксация изменений или запись изменений в репозиторий. Коммит происходит на локальной машине.
* **Пул (Pull)** — получение последних изменений с удалённого сервера репозитория.
* **Пуш (Push)** — отправка всех неотправленных коммитов на удалённый сервер репозитория.
* **Пулреквест (Pull Request)** — запрос на слияние форка репозитория с основным репозиторием. Пулреквест может быть принят или отклонён вами, как владельцем репозитория.
* **Мёрдж (Merge)** — слияние изменений из какой-либо ветки репозитория с любой веткой этого же репозитория. Чаще всего слияние изменений из ветки репозитория с основной веткой репозитория.
* **Кодревью** — процесс проверки кода на соответствие определённым требованиям, задачам и внешнему виду.

# Список иллюстраций

[Рис 1. Создание репозитория на гитхаб](https://docs.google.com/document/d/1onBq9kpazvDpFYvZHfNWI5tFz8S5tAiSjOI4Npoj0S4/edit" \l "bookmark=id.k7dv2nvo0uhm)[Рис 2. Инициализация git в рабочем каталоге](https://docs.google.com/document/d/1onBq9kpazvDpFYvZHfNWI5tFz8S5tAiSjOI4Npoj0S4/edit" \l "bookmark=id.qq7s4qbuejl3)[Рис 3. Создание README.md](https://docs.google.com/document/d/1onBq9kpazvDpFYvZHfNWI5tFz8S5tAiSjOI4Npoj0S4/edit" \l "bookmark=id.b3wtzqzdm8b5)[Рис 4. Первый коммит и отправка на гитхаб](https://docs.google.com/document/d/1onBq9kpazvDpFYvZHfNWI5tFz8S5tAiSjOI4Npoj0S4/edit" \l "bookmark=id.u195r38ubwtj)[Рис 5. Файл лицензии](https://docs.google.com/document/d/1onBq9kpazvDpFYvZHfNWI5tFz8S5tAiSjOI4Npoj0S4/edit" \l "bookmark=id.u8k7tbz0am5b)[Рис 6. Файл .gitignore](https://docs.google.com/document/d/1onBq9kpazvDpFYvZHfNWI5tFz8S5tAiSjOI4Npoj0S4/edit" \l "bookmark=id.smwvgvb9n9yj)[Рис 7. Коммит и отправка результата на гитхаб](https://docs.google.com/document/d/1onBq9kpazvDpFYvZHfNWI5tFz8S5tAiSjOI4Npoj0S4/edit" \l "bookmark=id.hlinc43x98dv)[Рис 8. Инициализация git flow](https://docs.google.com/document/d/1onBq9kpazvDpFYvZHfNWI5tFz8S5tAiSjOI4Npoj0S4/edit" \l "bookmark=id.ng7n97f865wy)[Рис 9. Создание нового релиза и файла с номером версии](https://docs.google.com/document/d/1onBq9kpazvDpFYvZHfNWI5tFz8S5tAiSjOI4Npoj0S4/edit" \l "bookmark=id.e2rpor6lubu)[Рис 10. Добавление изменений в индекс](https://docs.google.com/document/d/1onBq9kpazvDpFYvZHfNWI5tFz8S5tAiSjOI4Npoj0S4/edit" \l "bookmark=id.f8zbc8rv3q76)[Рис 11. Отправка данных на гитхаб](https://docs.google.com/document/d/1onBq9kpazvDpFYvZHfNWI5tFz8S5tAiSjOI4Npoj0S4/edit" \l "bookmark=id.6ukoqs9u4txa)[Рис 12. Окно создания релиза Рис 13. Окно с готовым релизом](https://docs.google.com/document/d/1onBq9kpazvDpFYvZHfNWI5tFz8S5tAiSjOI4Npoj0S4/edit" \l "bookmark=id.tval6lxc009n)

# Теоретическое введение:

*Установка Git в Windows*

Скачайте exe-файл инсталлятора с [сайта Git](https://git-scm.com/download/win) и запустите его. Это Git для Windows, он называется msysGit. Установщик спросит добавлять ли в меню проводника возможность запуска файлов с помощью Git Bash (консольная версия) и GUI (графическая версия). Подтвердите действие, чтобы далее вести работу через консоль в Git Bash. Остальные пункты можно оставить по умолчанию.

Для начала определим, что такое репозиторий. Это рабочая директория с вашим проектом. По сути, это та же папка с HTML, CSS, JavaScript и прочими файлами, что хранится у вас на компьютере, но находится на сервере GitHub. Поэтому вы можете работать с проектом удалённо на любой машине, не переживая, что какие-то из ваших файлов потеряются — все данные будут в репозитории при условии, что вы их туда отправите. Но об этом позже.

Если над проектом трудится команда разработчиков, как правило, создаётся общий репозиторий, в котором находится рабочая версия проекта (назовём его мастер-репозиторий). При этом каждый пользователь клонирует себе в профиль оригинальный репозиторий и работает именно с копией. Такая копия называется форком. Так как форк — ваша персональная версия мастер-репозитория, в нём вы можете пробовать разные решения, менять код и не бояться что-то сломать в основной версии проекта.

Работа с git описана в частности в [1].

*Как сделать форк мастер-репозитория?*

Заходим в нужный репозиторий, нажимаем на «вилку» с надписью fork. Форк репозитория создан и находится в вашем профиле на GitHub [2].

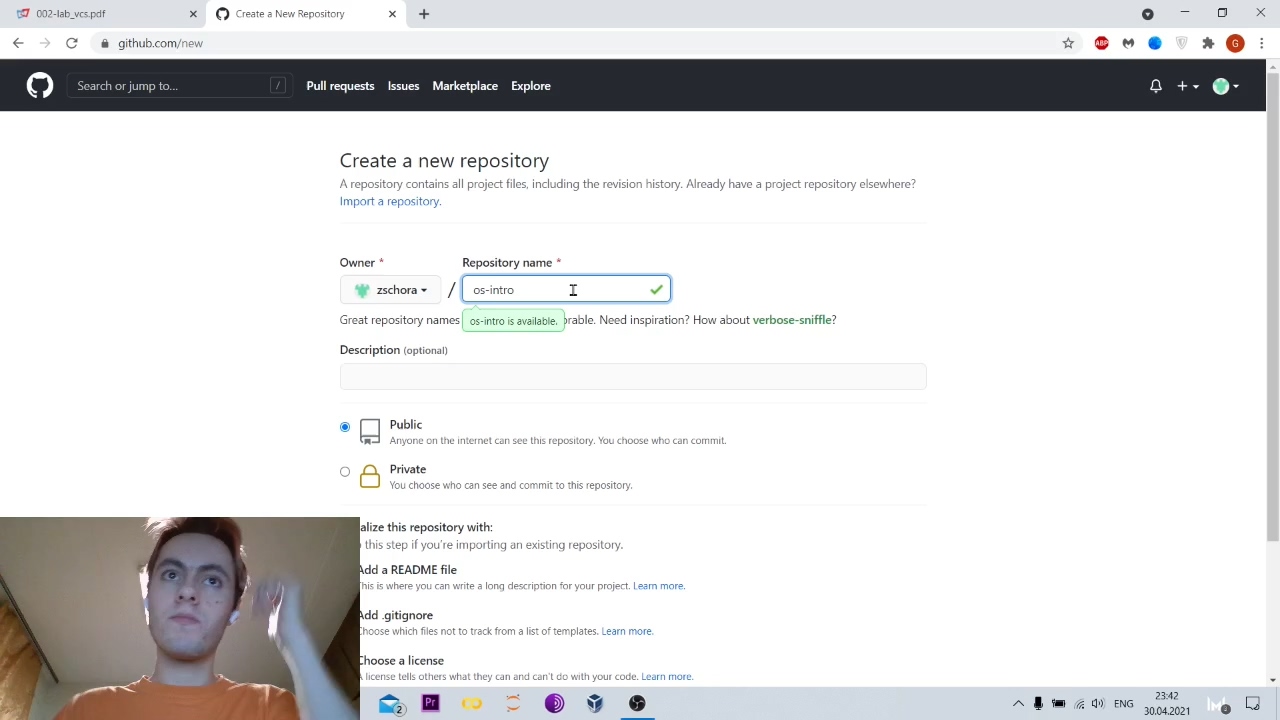
Теперь нужно склонировать форк себе на компьютер, чтобы вести работу с кодом локально. Тут нам и пригодится SSH.

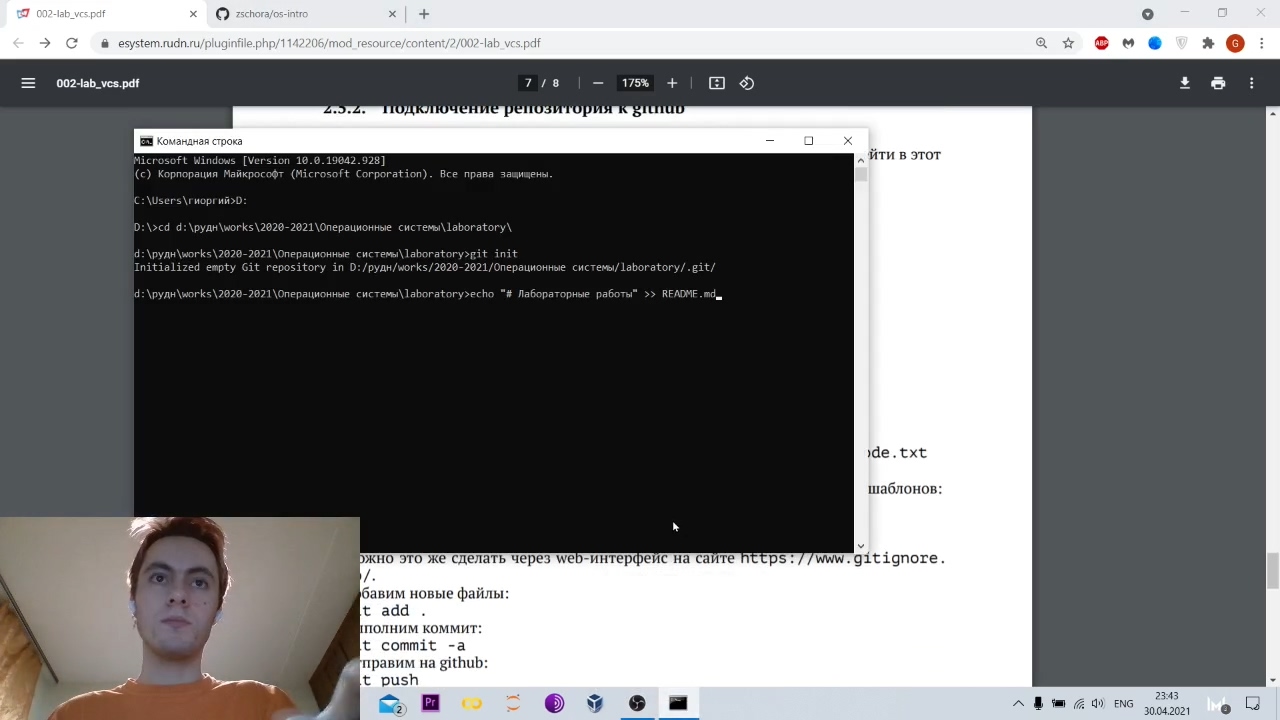
Открываем консоль, переходим в директорию, где хотим сохранить папку с проектом, и вводим команду:

*git clone*

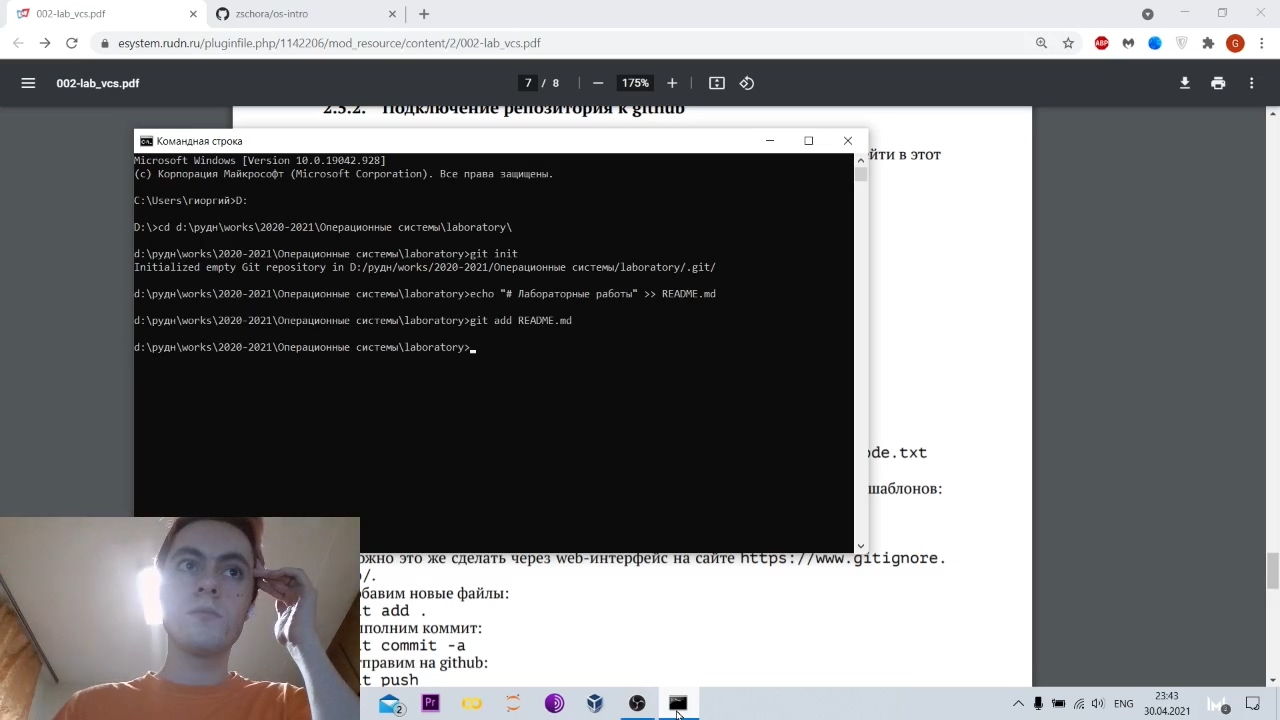
[*git@github.com*](mailto:git@github.com)*:your-nickname/your-project.git*

# Выполнение лабораторной работы

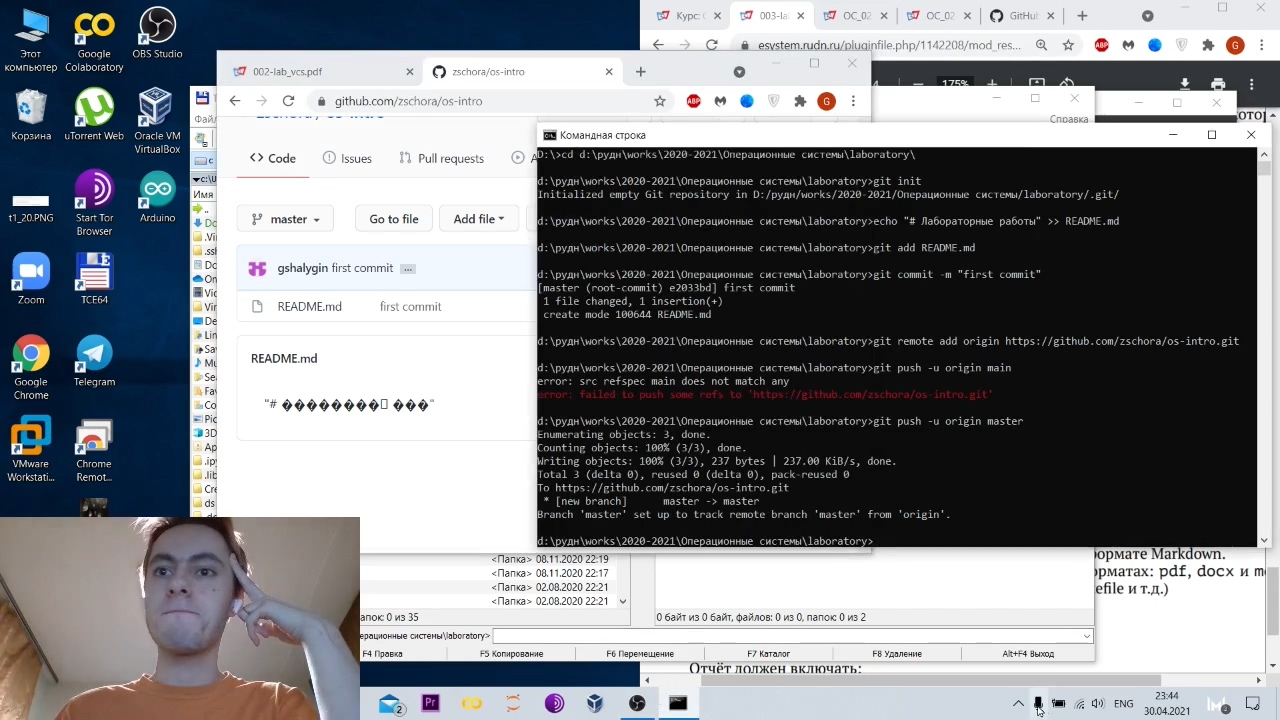
1. Создадим учётную запись на [https://github.com](https://github.com/) и новый репозиторий на git-hub с названием os-intro.
2. Перейдем в рабочий каталог командой cd d:\рудн\works\2020-2021\Операционные системы\laboratory и инициализируем в нем систему git

* 

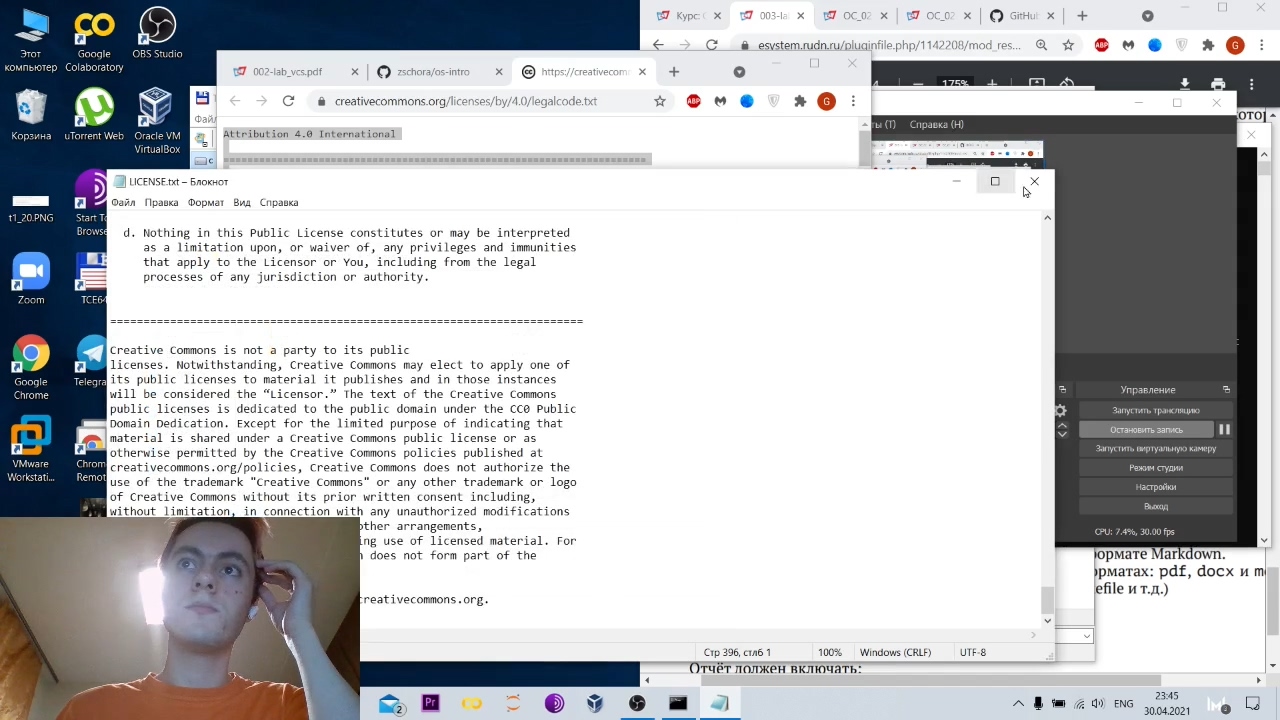
1. Создаём заготовку для файла README.md с помощью команд:echo "# Лабораторные работы" >> README.mdgit add README.md

* 

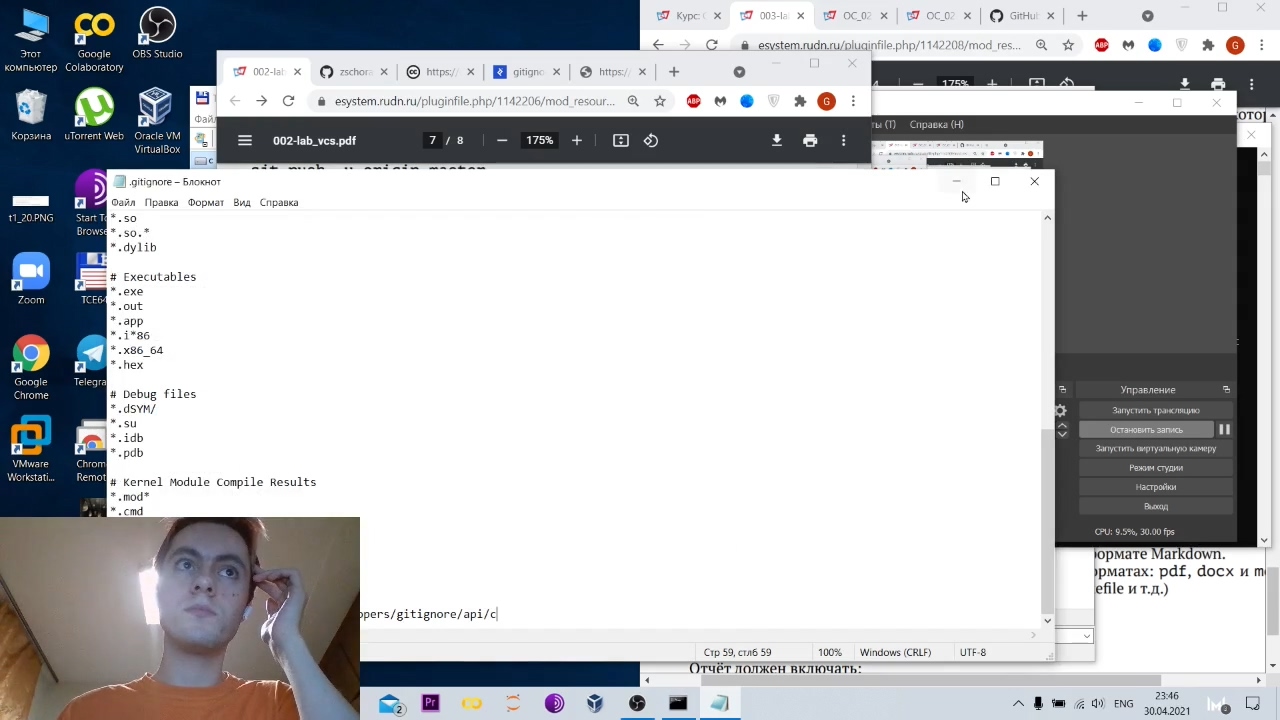
1. Делаем первый коммит и выкладываем на github:

* *git commit -m "first commit"*  
  *git remote add origin* [*https://github.com/zschora/os-intro.git*](https://github.com/zschora/dick.git)  
  *git push -u origin master*
* 

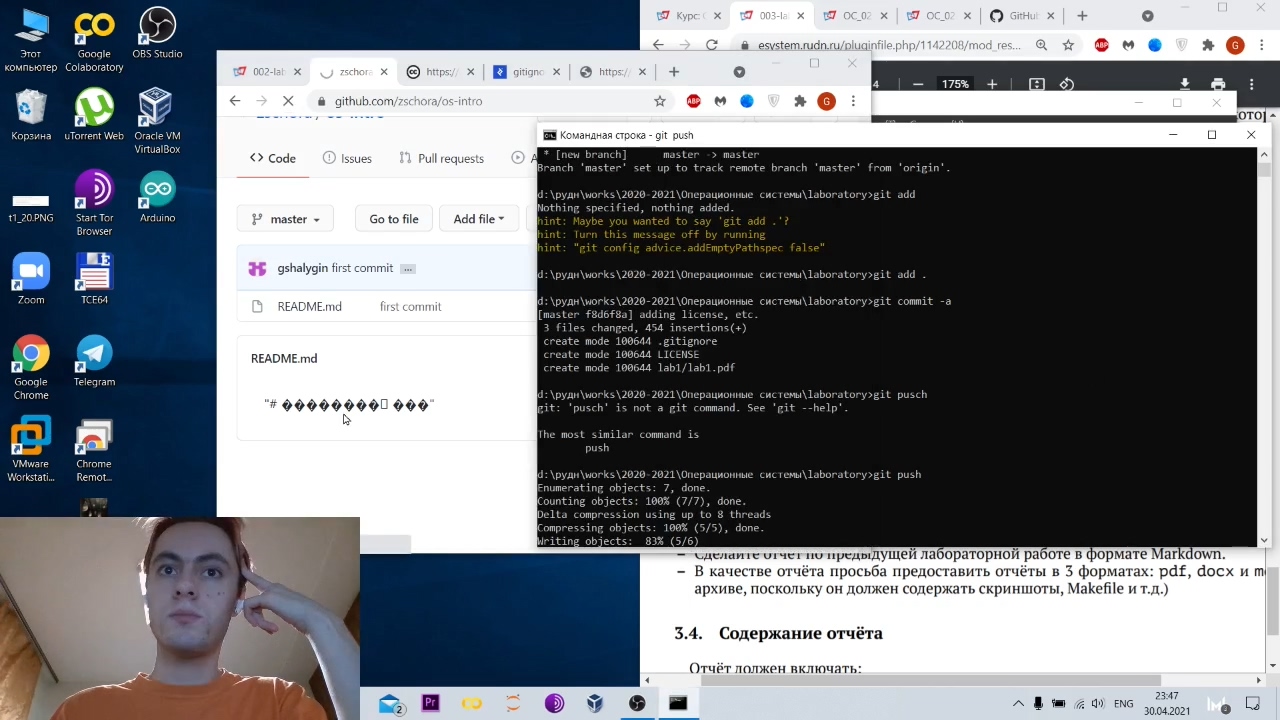
1. Добавим файл лицензии <https://creativecommons.org/licenses/by/4.0/legalcode.txt>

* 

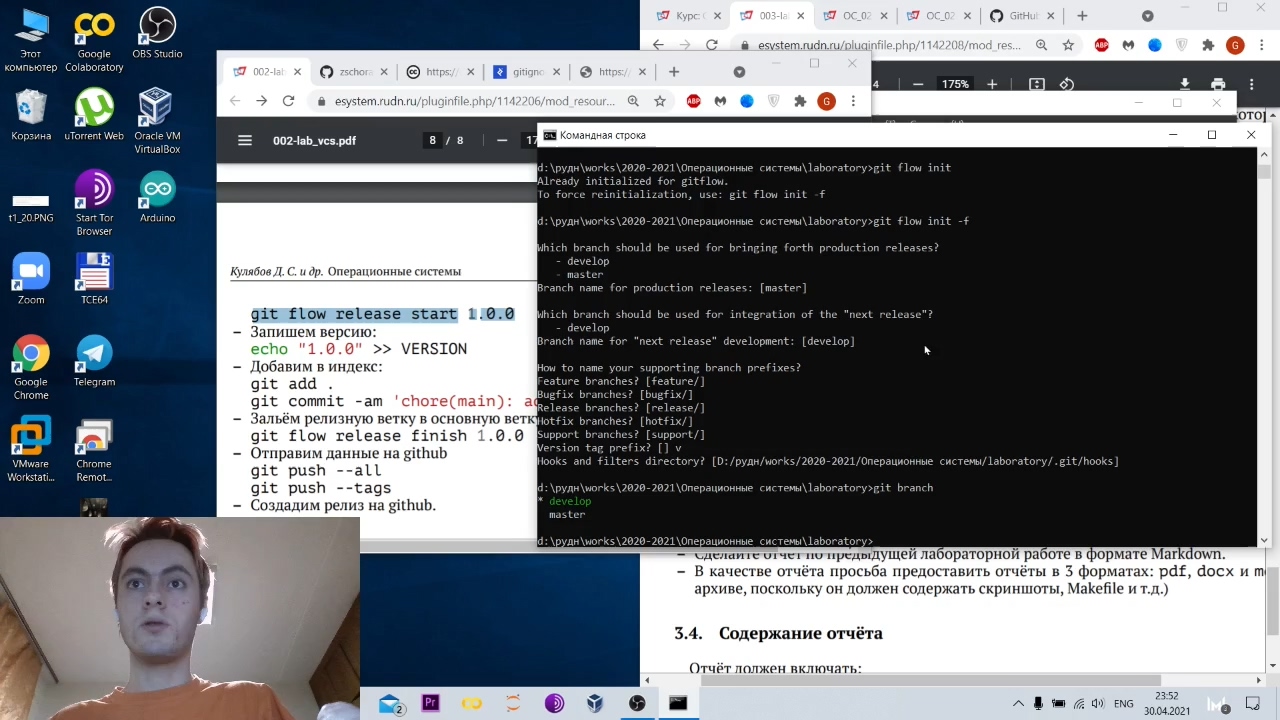
1. Добавим шаблон игнорируемых файлов .gitignore для языка C.

* 

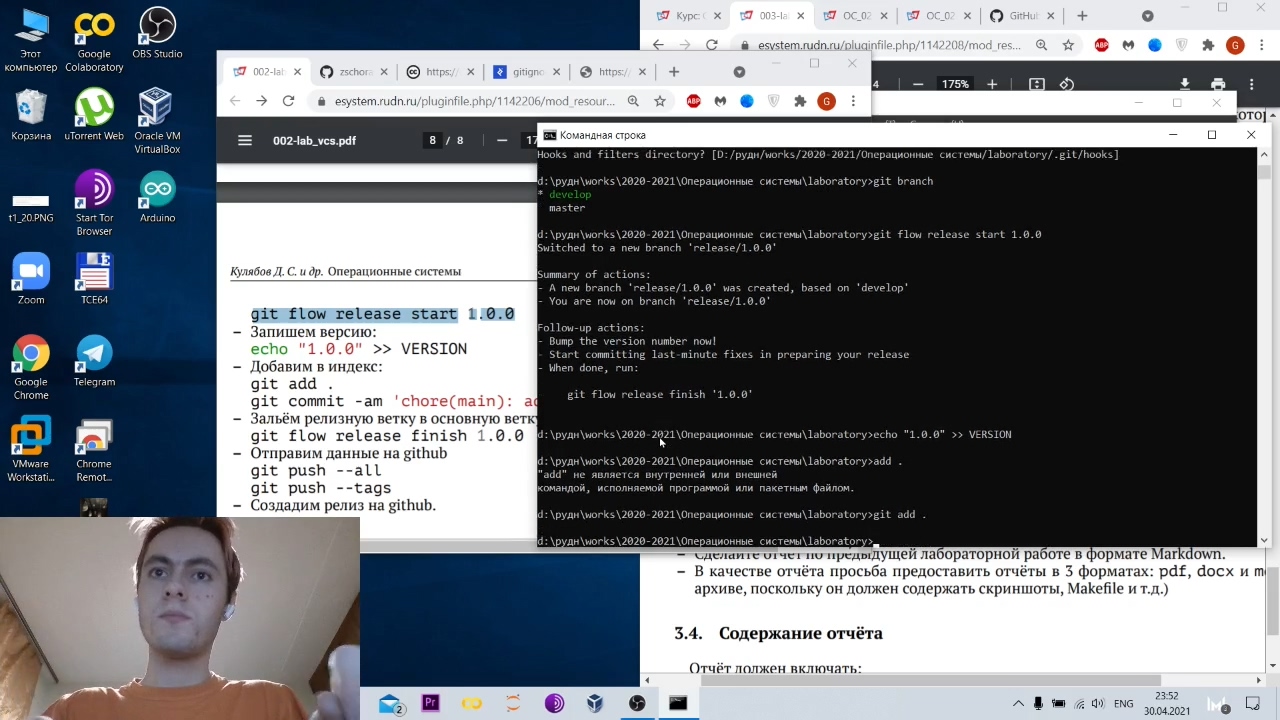
1. Добавим новые файлы:

* *git add .*
* Выполним коммит:
* *git commit -a*
* Отправим на github:
* *git push*
* 

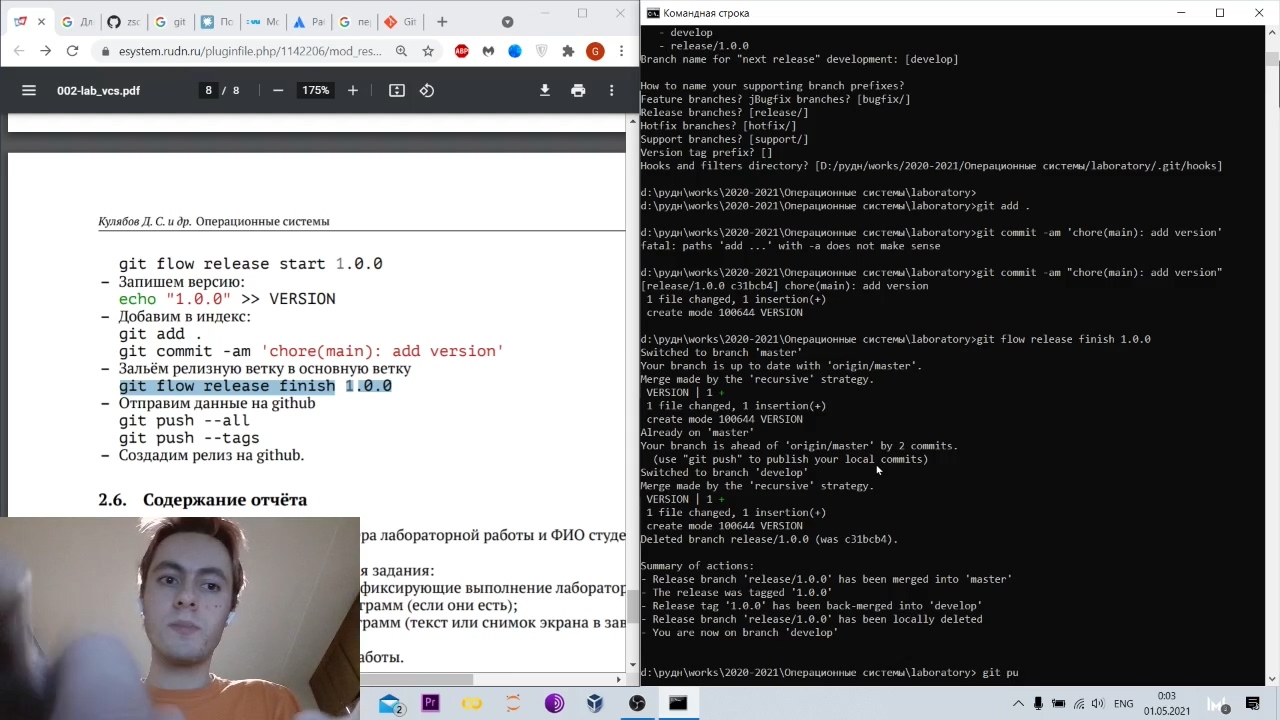
1. Инициализируем git-flow командой *git flow init*

* Префикс для ярлыков установим в v. Проверьте, что Вы на ветке develop: *git branch*
* 

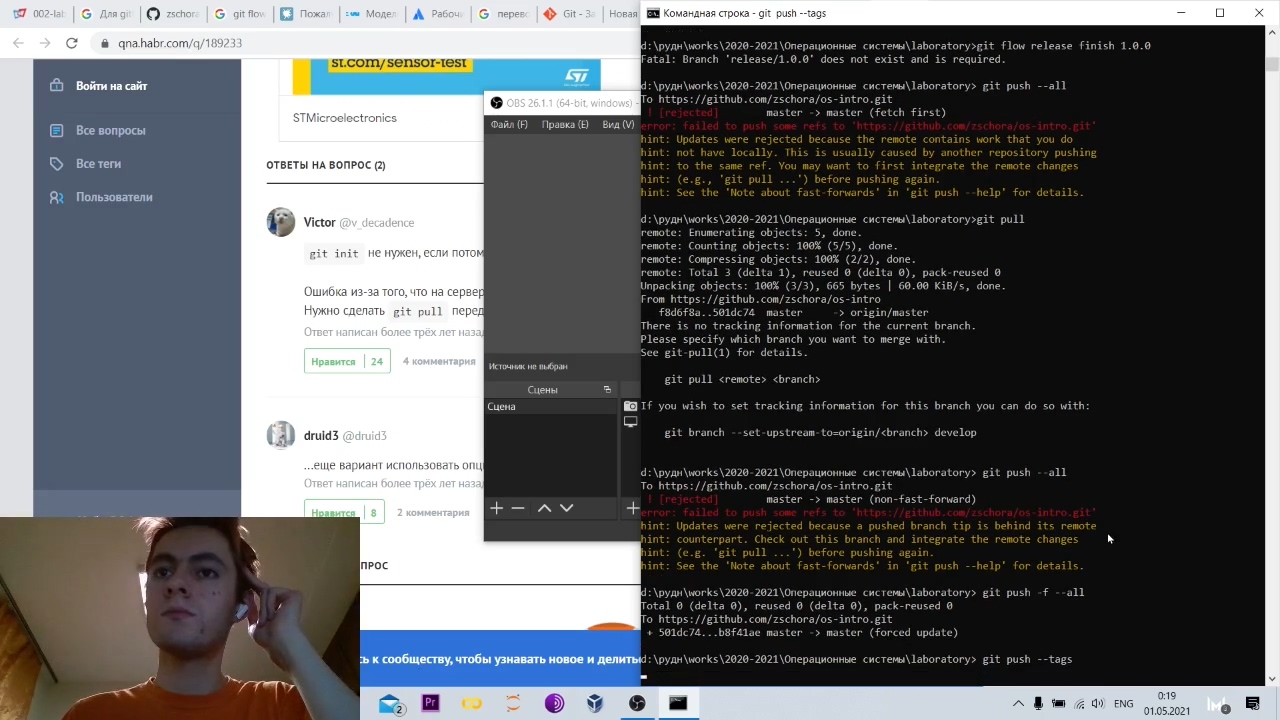
1. Создадим релиз с версией 1.0.0: *git flow release start 1.0.0*

* И запишем версию в файл VERSION: *echo "1.0.0" >> VERSION*
* 

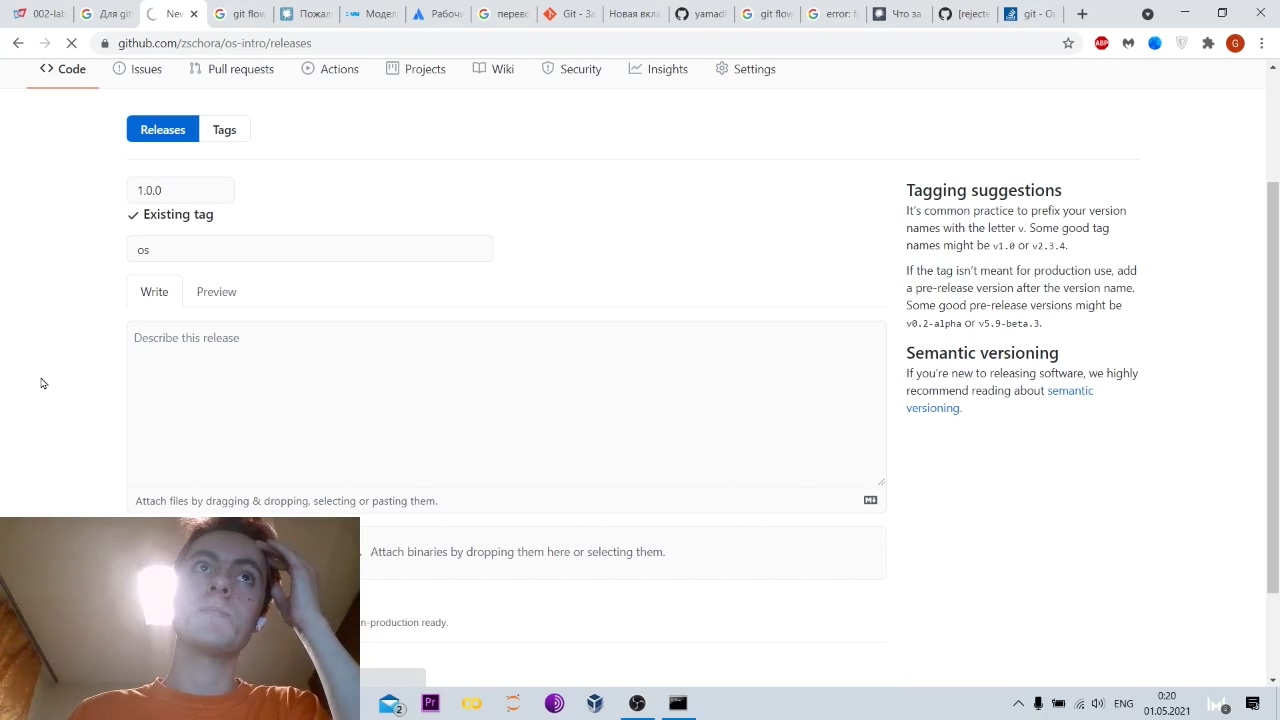
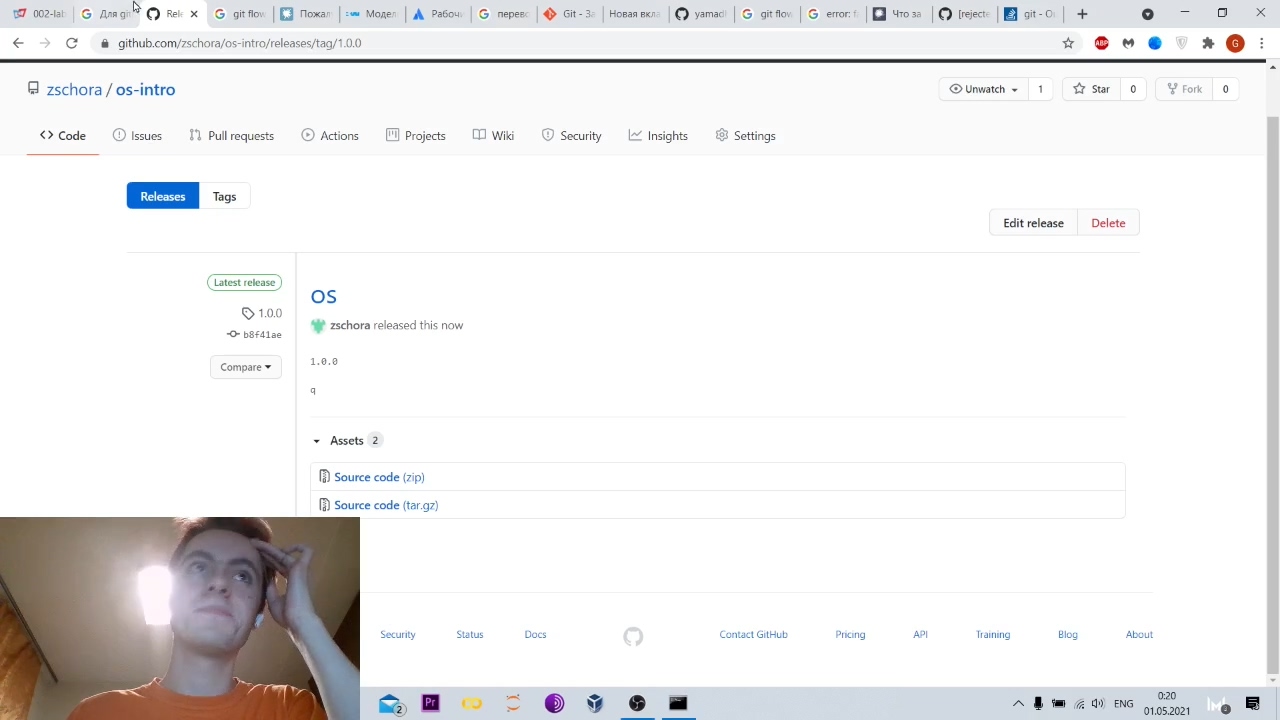
1. Добавим в индекс:

* *git add .*
* *git commit -am “chore(main): add version”*
* 

1. Отправим данные на github: *git push --allgit push --tags*

* 

1. Создадим релиз на github:

* 
* 

# Выводы

В процессе работы над лабораторной работы были получены навыки использования git в связке с сайтом github.com, также освоена конфигурация git flow.

# Библиография

1. <https://proglib.io/p/git-cheatsheet>
2. [https://www.github.com](https://www.virtualbox.org/wiki/Downloads)