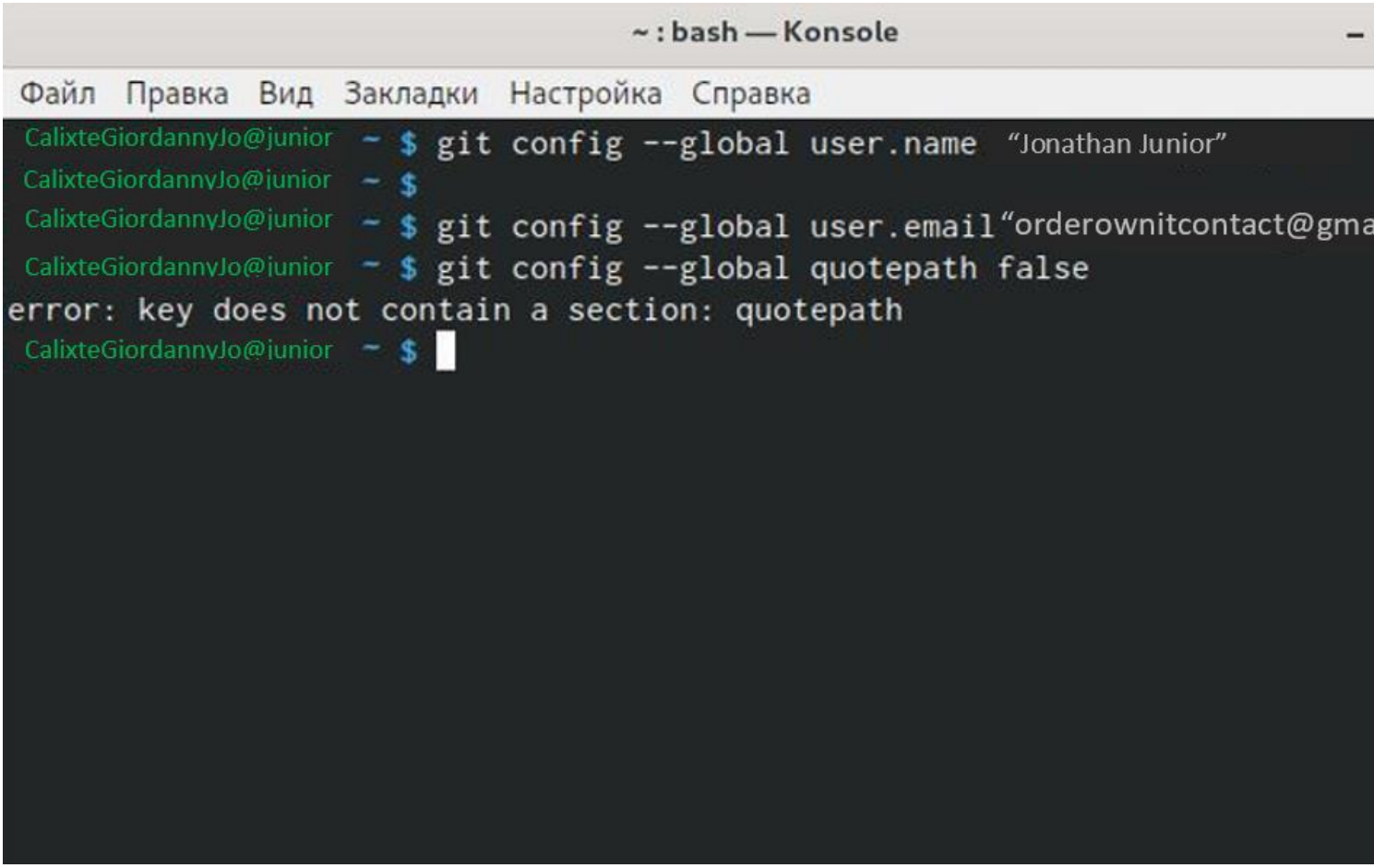
Отчетпо лабораторной работе № 2

Цельработы

Изучитьидеологию и применение средств контроляверсий.

Ход работы

. Создали учетную запись на github.com ,настроили giti создали структуру каталога



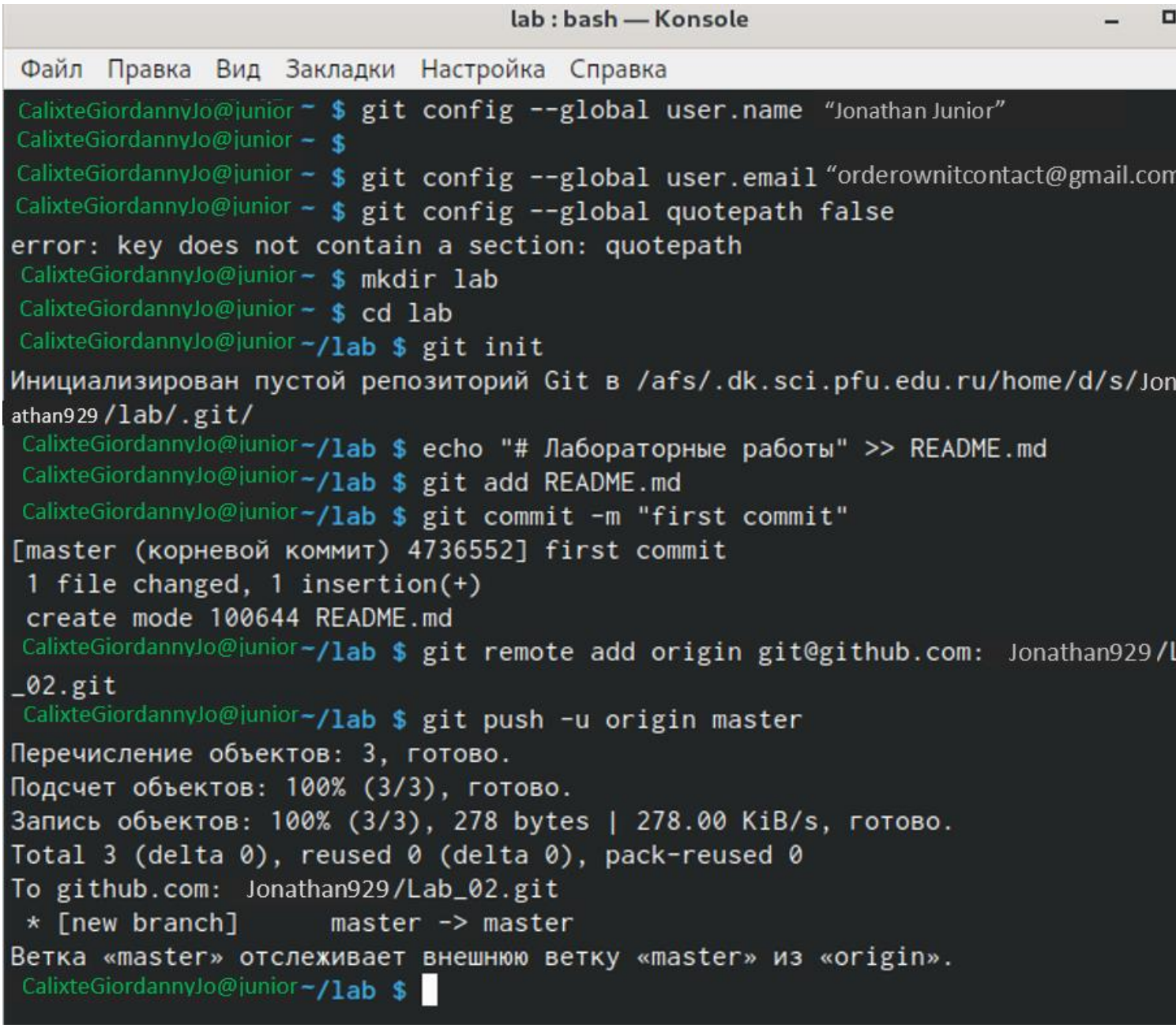
Создали репозиторий на github.com ,назвали его Lab_02

. Переш ли в рабочий каталог `git init`

.Инициализировали систему контроляверсий

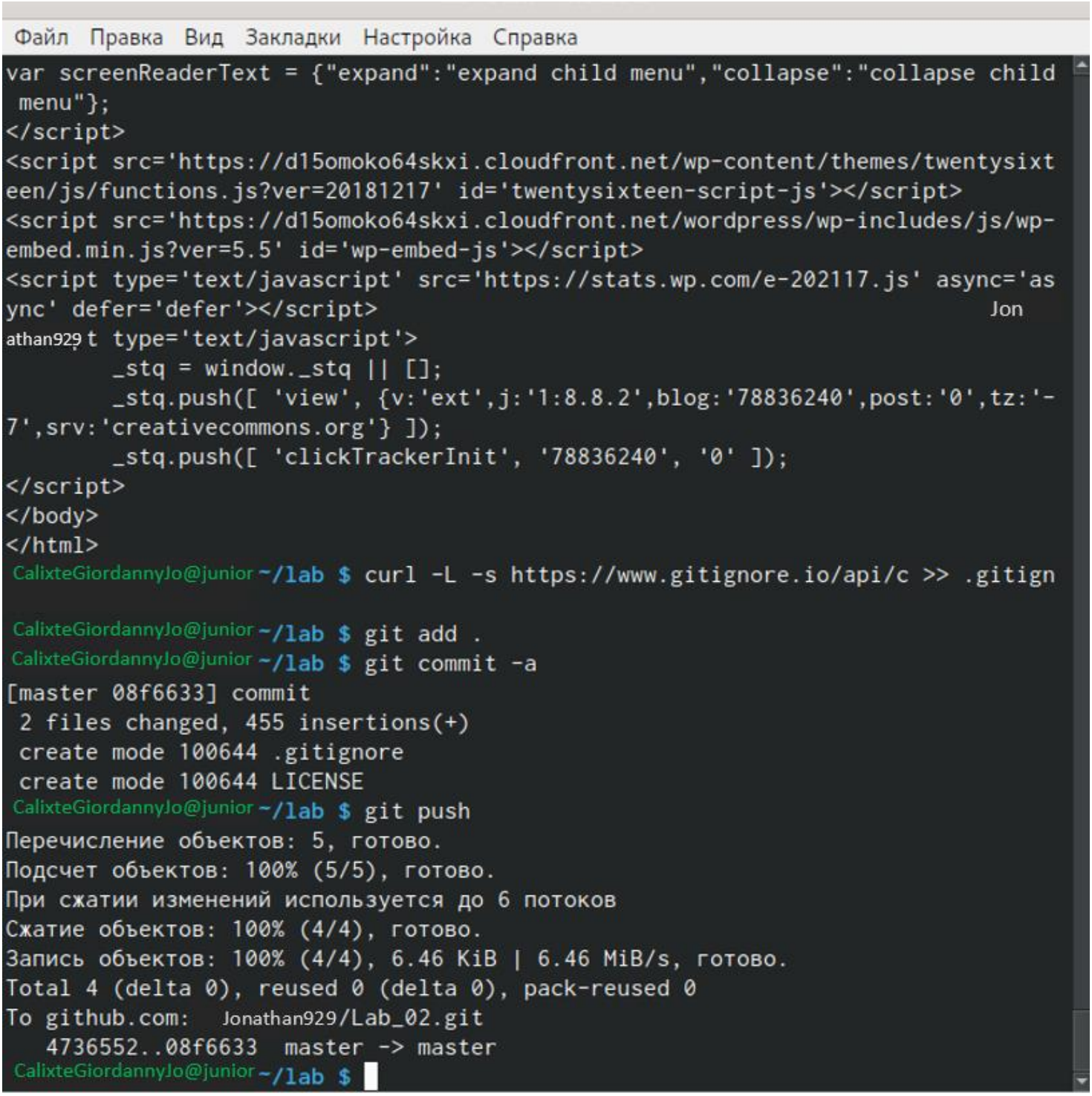
. Создали заготовку файла README.md `echo "Лабораторная работа №2" >> README.md git add README.md`

. Сделали первый коммит и выложили на github

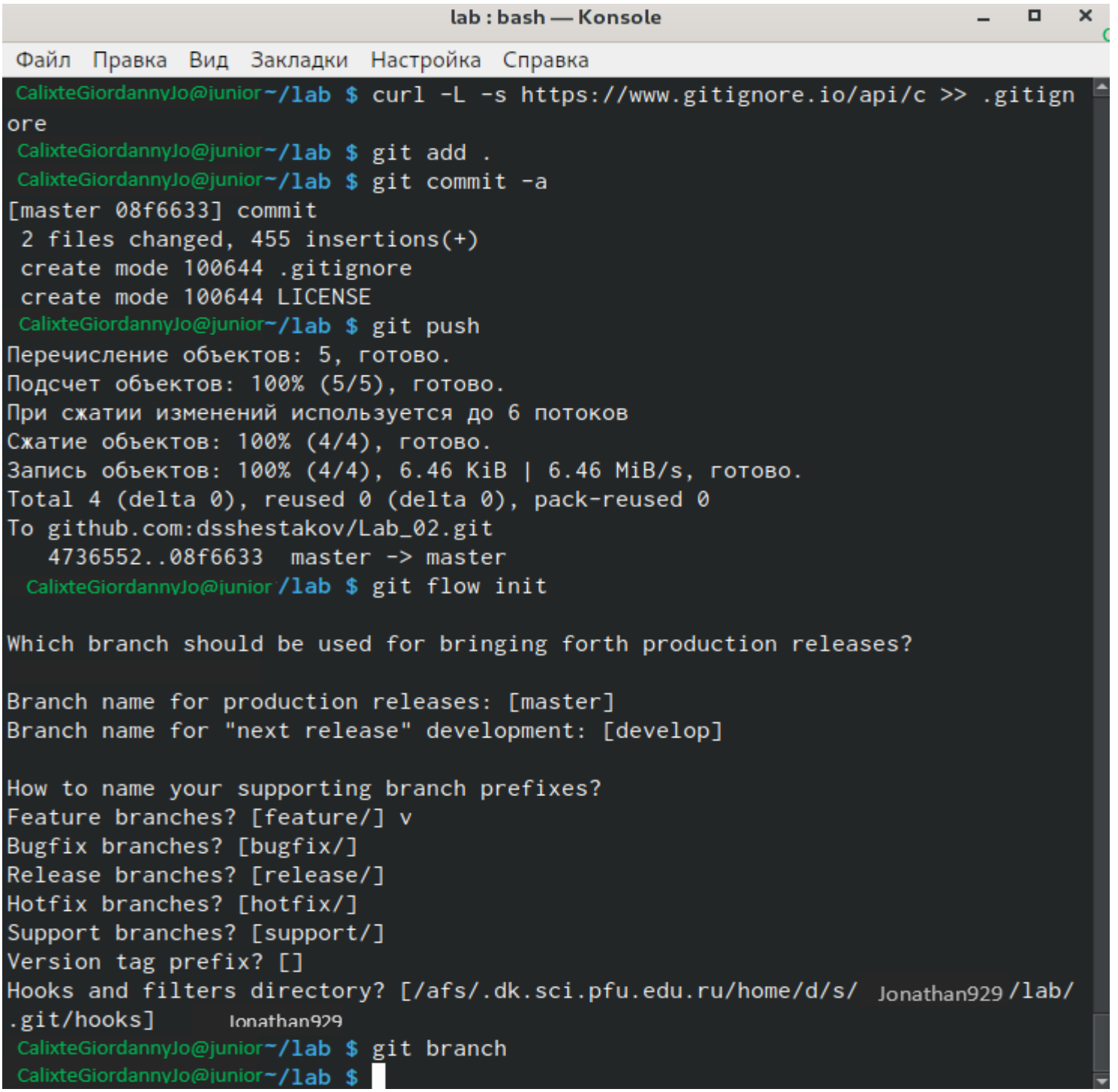


7.Добавили файл лицензии `wget https://creativecommons.org/licenses/by/4.0/legalcode.txt -O LICENSE` **8.Скачали шаблон для C** `curl -L -s https://www.gitignore.io/api/c >> .gitignore` **9.Добавили новые файлы,выполнили коммит,отправили на github**

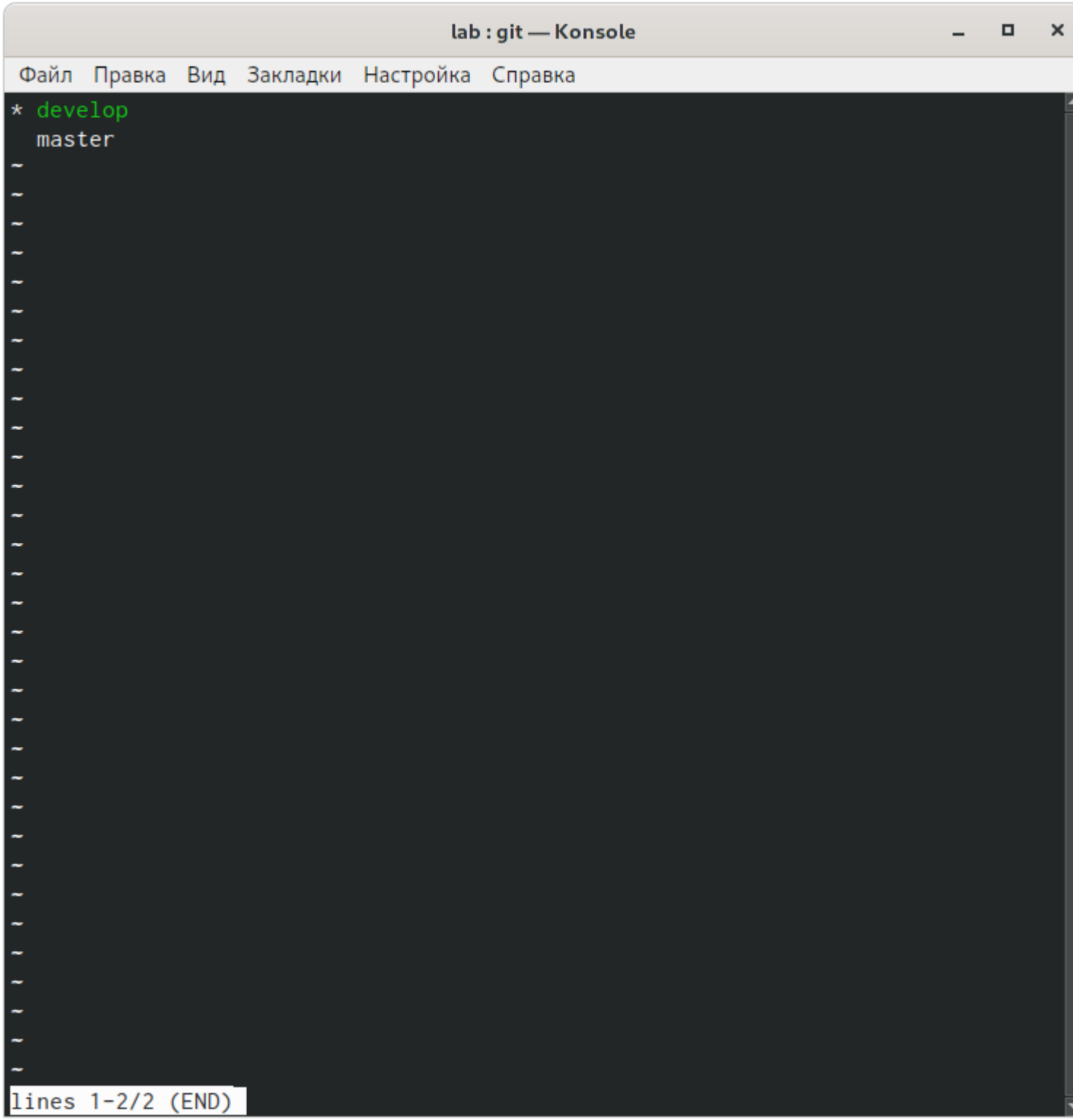
```
git add .
git commit -a
git push
```



10.Инициализировали gitflow `git flow init`



11.Проверили,что мы на ветке develop `git branch`

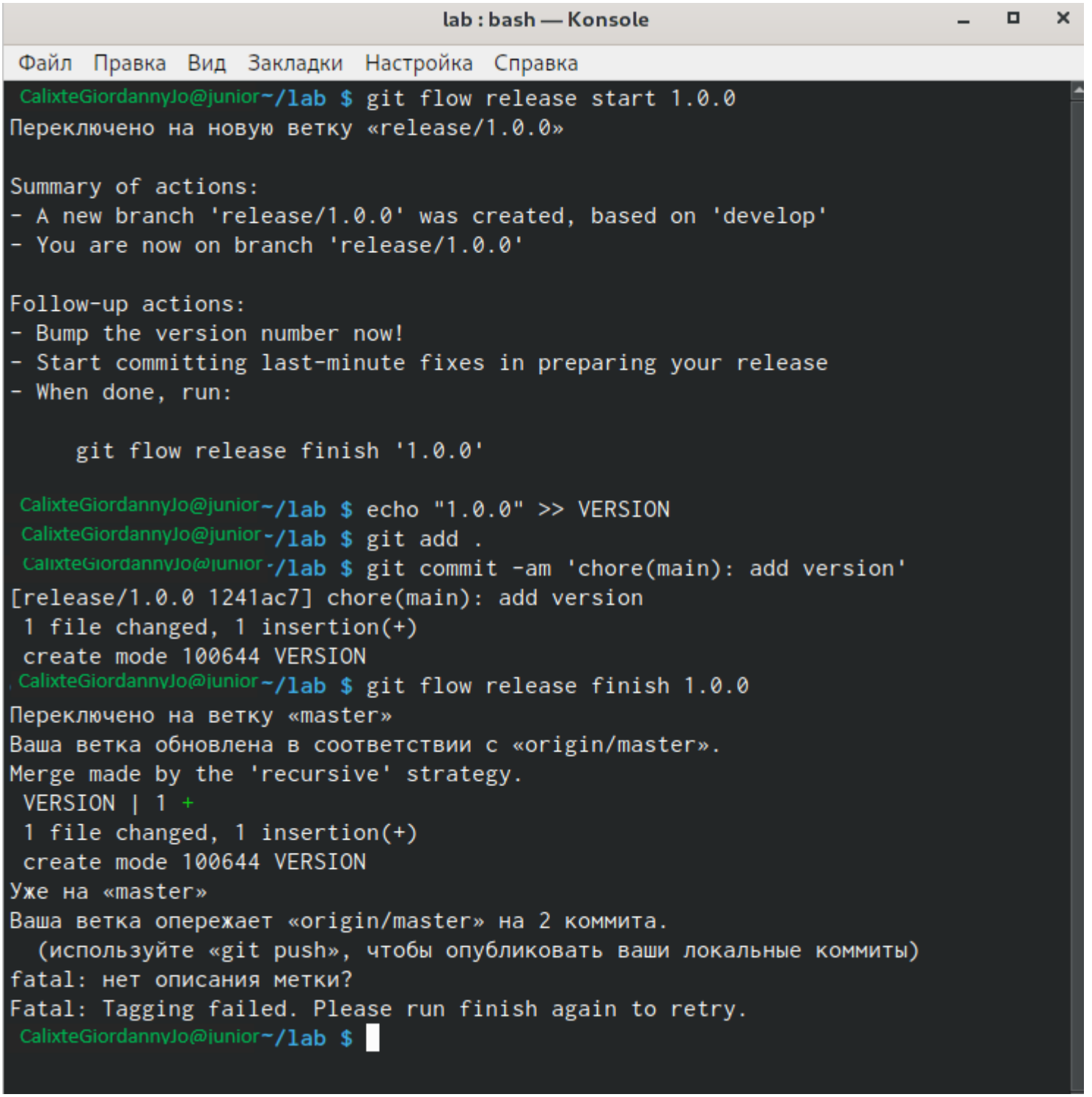


12.Создали релиз 1.0.0 `git flow release start 1.0.0` **13.Добавили индекс**

```
git add .
git commit -am "chore(main): add version"
```

. Залили релизную ветку в основную ветку `git flow release finish 1.0.0` **. Отправили данные на github**

```
git push --all
git push --tags
```



Контрольныевопросы

. Система контроля версий (Version ControlSystem ,VCS) представляет собой программное обеспечение,которое позволяетотслеживатьизмененияв документах,при необходимости производитьихоткат,определять,кто и когда внес исправленияи т.п.

.Репозиторий - пециальное хранилищ е файлов и папокпроекта,измененияв которых отслеживаются.Коммит- операцияотпраки в репозиторий измений,которые поьзователь внес в свою рабочую копию.История - все коммиты соверш енные кданному моменту. Рабочаякопия - версияпроекта,с которой работаетразработчик.

. Централизованная система контроля версий(CVS,Subversion) - репозиторий сущ ествует в единственном экзепляре и хранитсяна сервере.Распределеннаясистема контроляверсий

(Git) - у каждого разработчика есть собственный репозиторий,при этом сущ ествует условно центральный репозиторий,куда будутотправлятьсяизмененияизлокальных

.При работе с общ им хранилищ е,,надо собственную ветку.Затем надо получитьинформацию об изменениях изцентрального репозитория.После можно вноситьсвои изменения.Длятого, чтобы отправитьизменения в центральный репозитория,надо проверитькакие файлы притерпели изменения и при необходимости удалитькакие-то файлы,которые не должны бытьотправлены в центральный репозиторий.Затем создаем коммити загружаем файлы в центарльный репозиторий.

.Возвраткранним версиям проектам,отслеживание истории изменений,устраняетопасность удаление чужихфайлов и даетвозможностьне боятьсяза потерю даных.

. Наиболее часто используемые команды git:– создание основного дерева репозитория:gitinit

– получение обновлений(изменений)текущ его дерева из центрального репозитория:gitpull–

отправка всех произведённых изменений локального дерева в центральный репози-торий:git

push – просмотр списка изменённых файлов в текущ ей директории:gitstatus – просмотр

текущ их изменения:gitdiff – сохранение текущ их изменений:– добавить все изменённые и/

или созданные файлы и/или каталоги:gitadd .– добавить конкретные изменённые и/или

созданные файлы и/или каталоги:gitadd имена_файлов – удалить файл и/или каталог из

индекса репозитория(при этом файл и/или каталог остаётся в локальной директории):gitrm

имена_файлов – сохранение добавленныхизменений:– сохранитьвсе добавленные

изменения и все изменённые файлы:gitcom m it-am 'Описание коммита'– сохранить

добавленные изменения с внесением комментария через встроенный редактор:gitcom m it–

создание новой ветки,базирующ ейся на текущ ей:gitcheckout-b имя_ветки – переключение на

некоторую ветку:gitcheckoutимя_ветки (при переключении на ветку,которой ещ ё нет в локальном репозитории,она будетсоздана и связана с удалённой)– отправка изменений конкретной ветки в центральный репозиторий:gitpush origin имя_ветки – слияние ветки с текущ им деревом:gitm erge --no-ff имя_ветки – удаление ветки:– удаление локальной уже слитой с основным деревом ветки:gitbranch -d имя_ветки – принудительное удаление локальной ветки:gitbranch -D

имя_ветки – удаление ветки с центрального репозитория:git push origin :имя_ветки

.При помощ и веток в VCS можно:

Реализоватьфичу,не меш аяостальным.Проводитьмодерацию (кодревью)нового кода перед непосредственным добавлением в кодовую базу.Отвлечьсяотреализации фичи и починитьбагв другом месте.Вовсе отложитьначатую фичу до лучш ихвремен.Получитьзапрос на доработку старой версии программы отзаказчика и поддерживатьдалее несколько версий ПО.

Поэкспериментироватьс кодом безстраха сломатьбилд.Организоватьпроцесс поэтапного выпуска программы (разработка - тестирование - релиз),не блокируяразработку следующ ей версии.Организовать работу с open source сообщ еством или подрядчиком.Запилить постоянную автоматическую сборку с рабочей ветки с прогоном тестов и ручную авторизованную сборку релиза с релиз-ветки.