Научное программирование

Лабораторная работа №1.

Подмогильный Иван Александрович.

Содержание

# 1 Цель работы

Освоить на практике использование системы контроля версий Git.

# 2 Задание

* Создать базовую конфигурацию для работы с git.
* Создать ключ SSH.
* Создать ключ PGP.
* Настроить подписи git.
* Зарегистрироваться на Github.
* Создать локальный каталог для выполнения заданий по предмету.

# 3 Выполнение лабораторной работы

## 3.1 Результаты выполнения лабораторной работы. Установил git-flow в Ubuntu

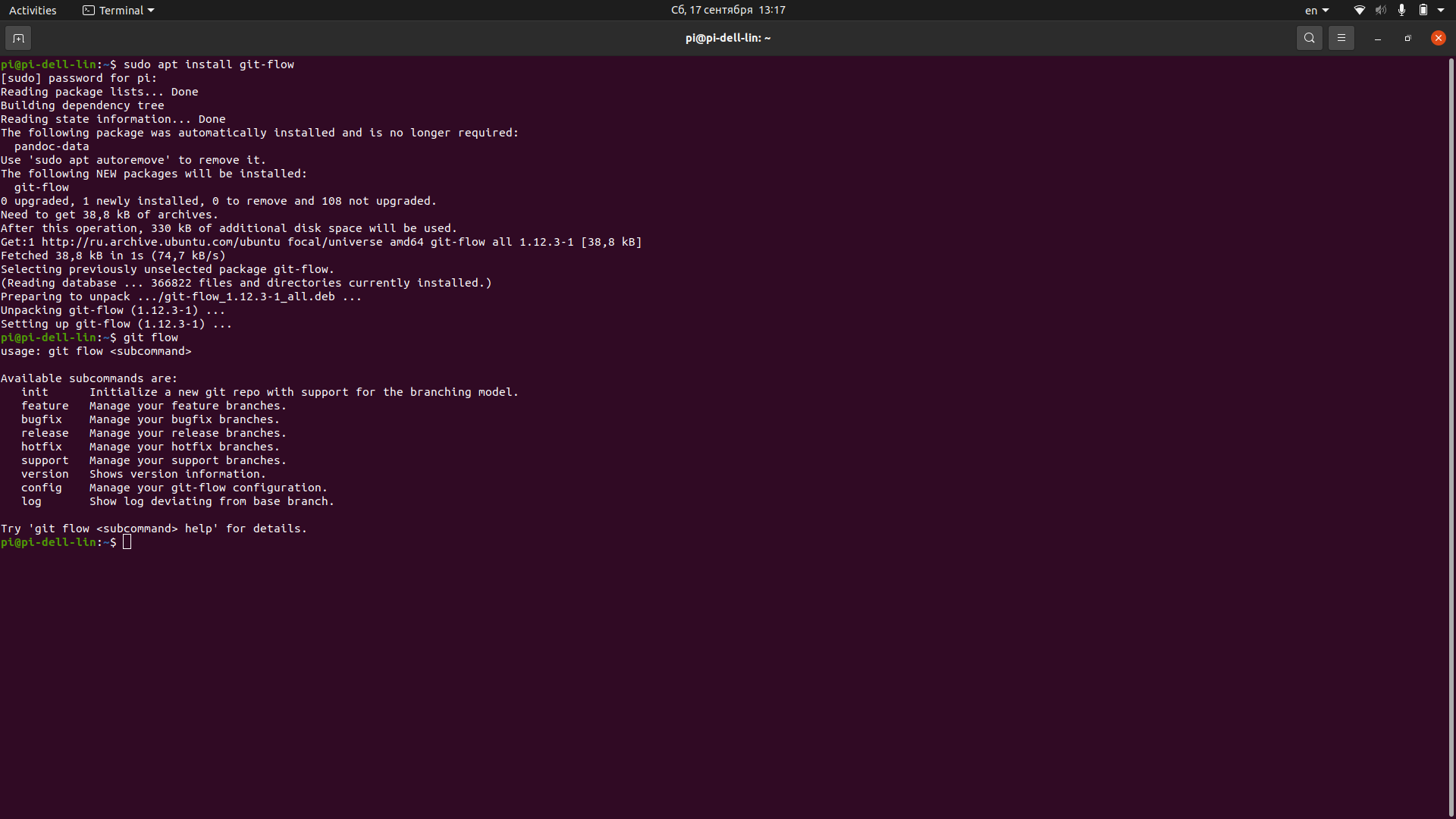


Figure 1: 1

## 3.2 Установил gh

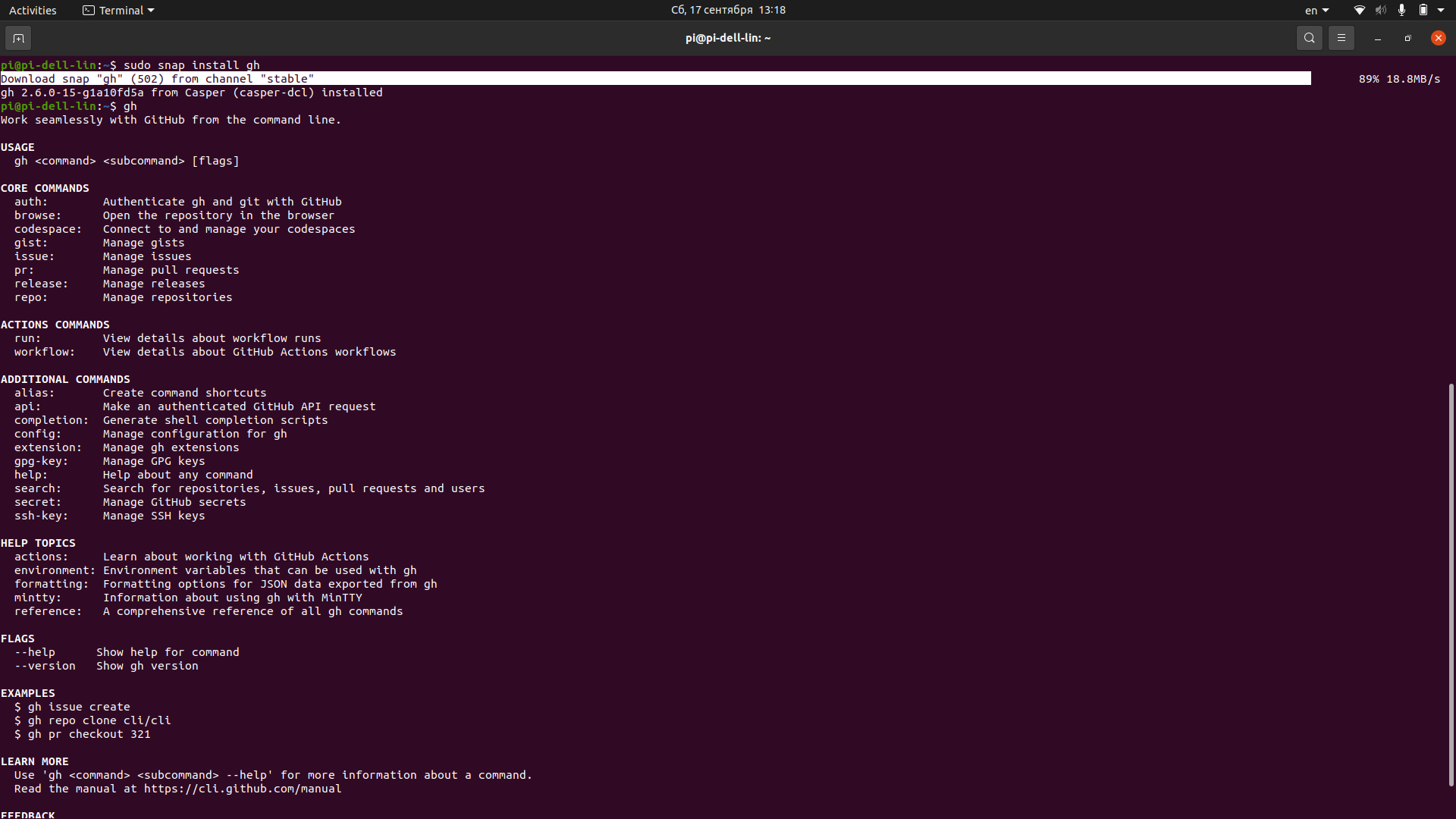


Figure 2: 2

## 3.3 Задал базовую настройку git. Настроил утф-8 в выводе сообщений гит.

Зададим имя начальной ветки (будем называть её master). Настроил верификацию и подписание коммитов git.

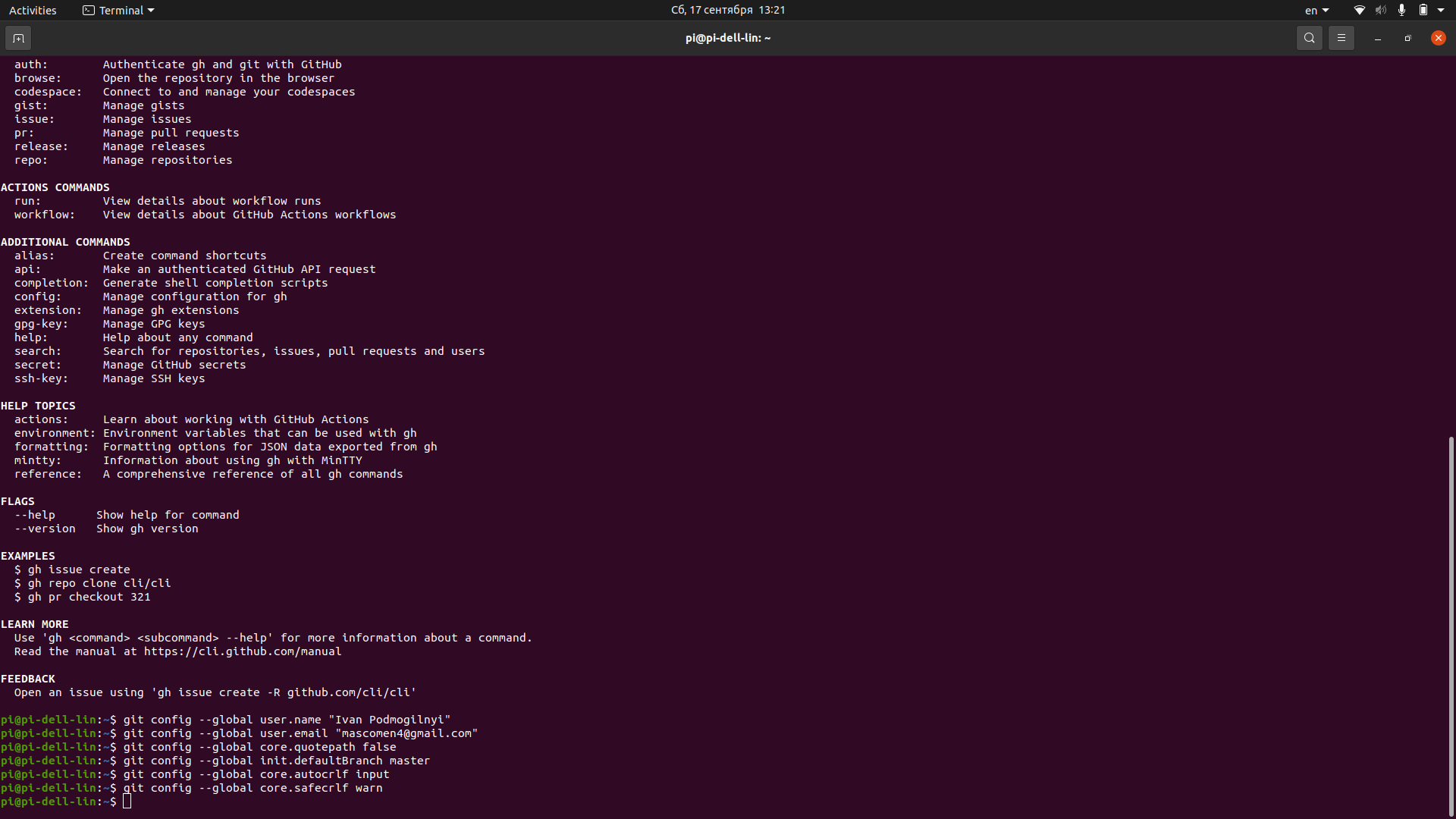


Figure 3: 3

## 3.4 Создайте ключи ssh по алгоритму rsa с ключём размером 4096 бит по алгоритму ed25519

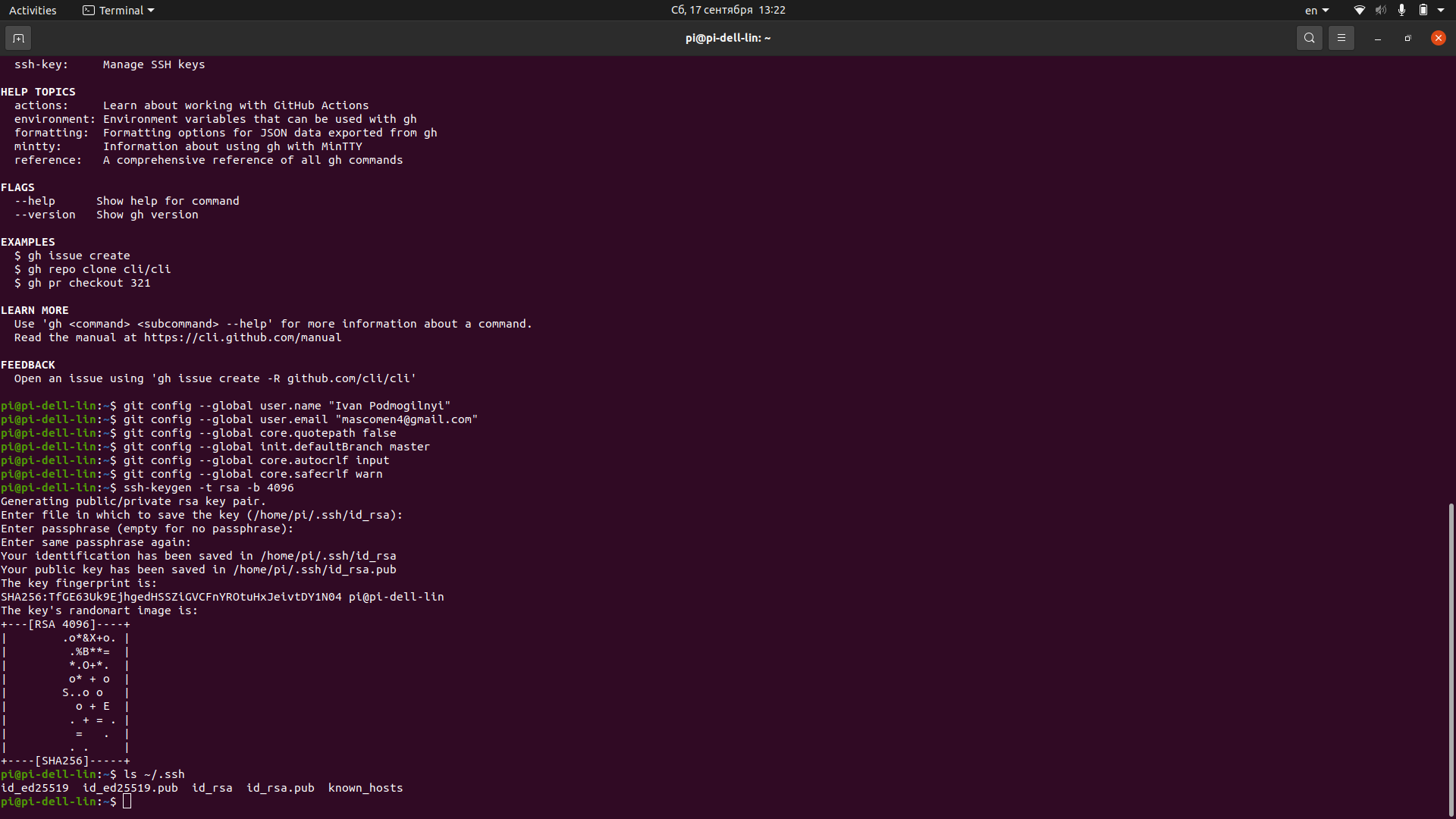


Figure 4: 4

## 3.5 Создайте ключи pgp Генерируем ключ

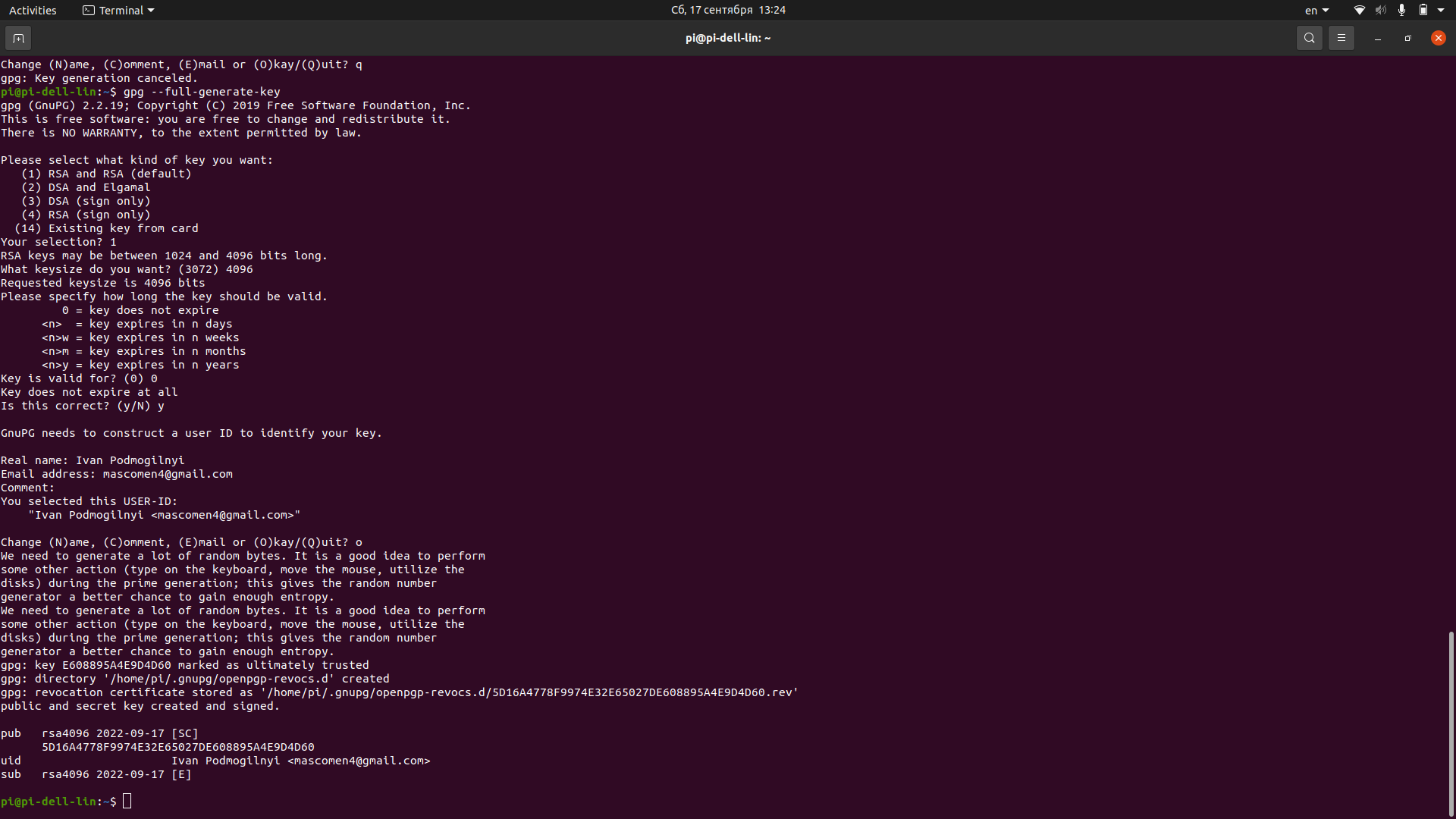


Figure 5: 5

## 3.6 Добавление PGP ключа в GitHub

Выводим список ключей и копируем отпечаток приватного ключа Cкопируйте ваш сгенерированный PGP ключ в буфер обмена

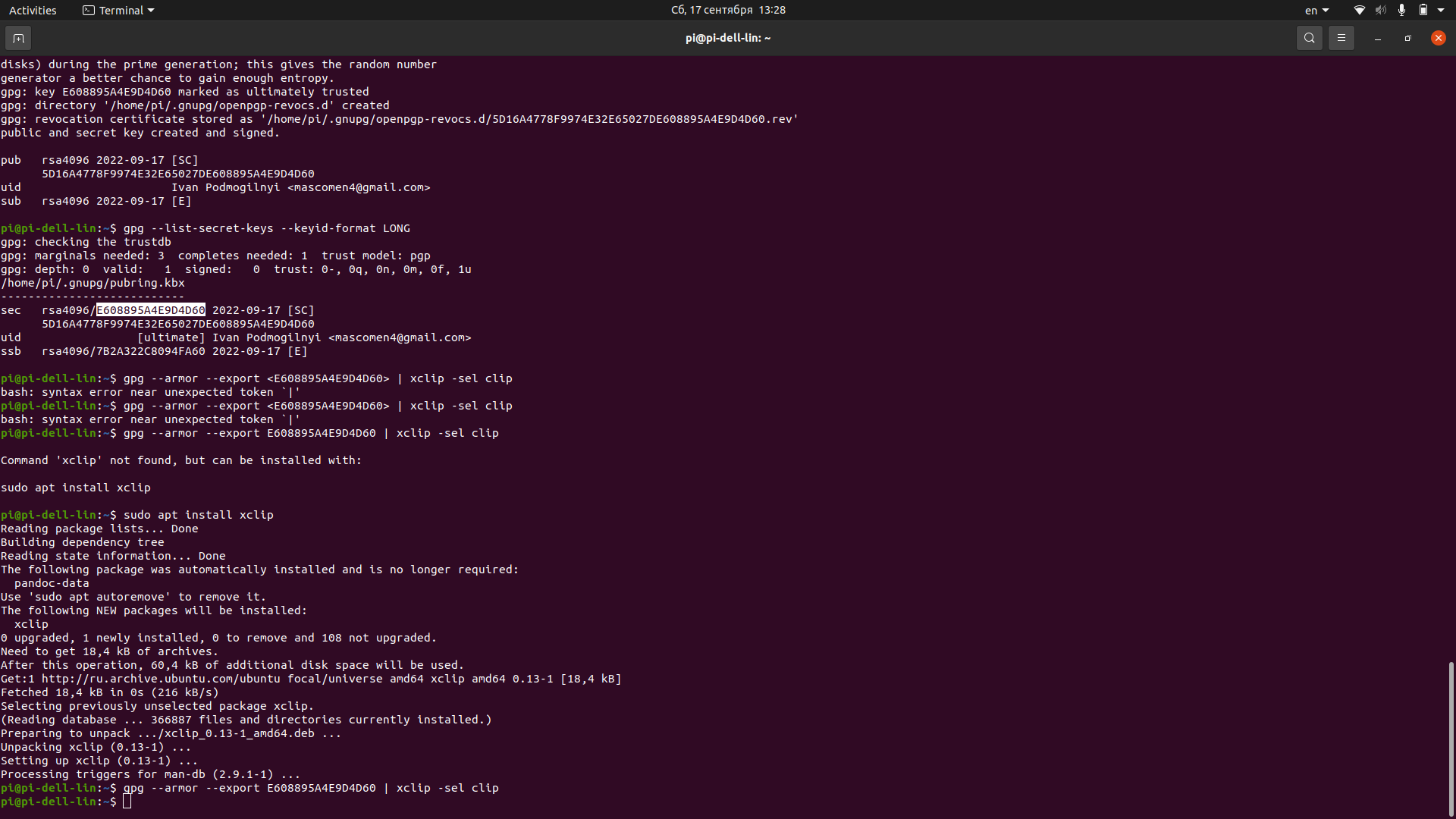


Figure 6: 6

## 3.7 Добавил скопированный ключ в Github аккаунт

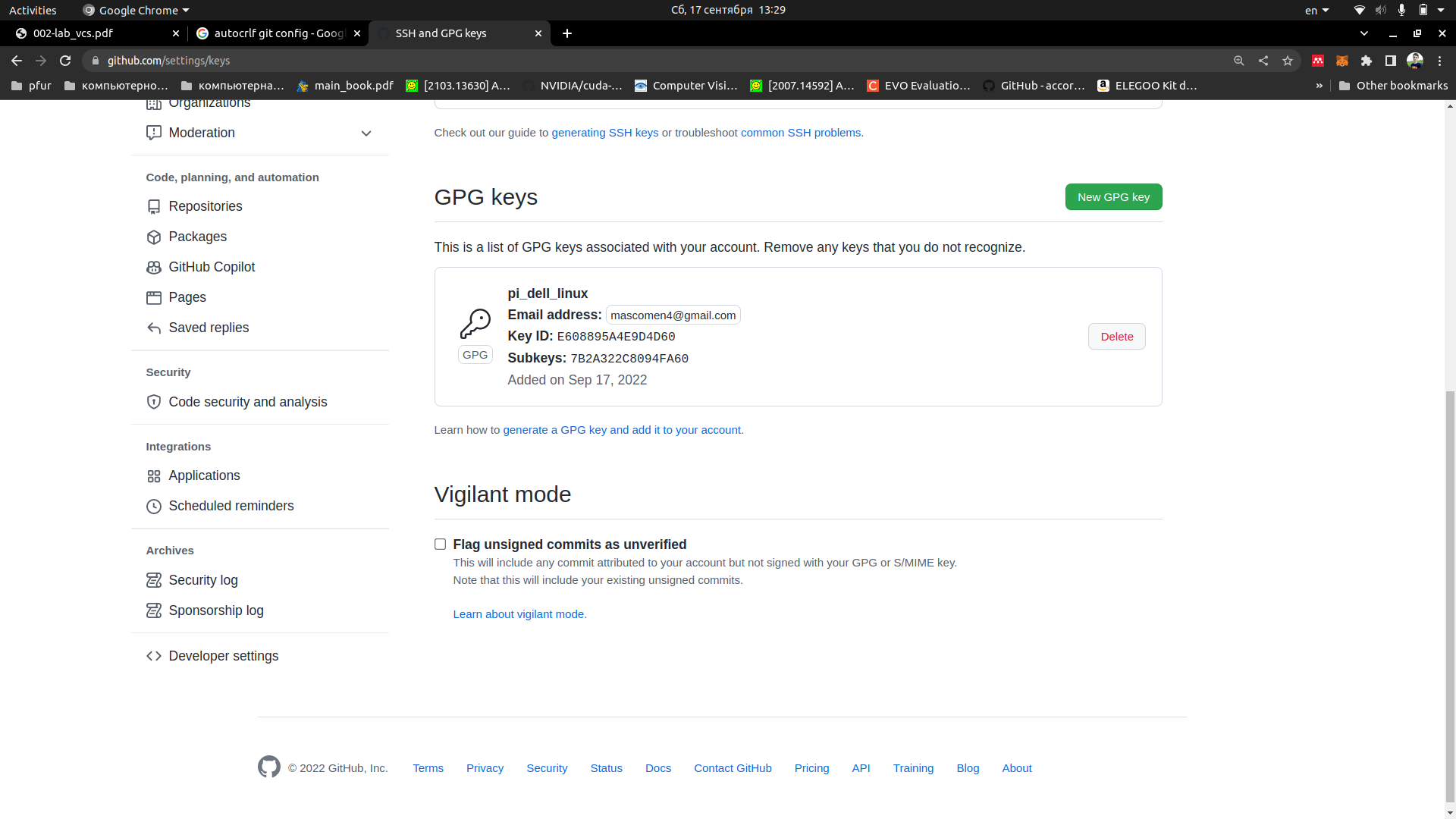


Figure 7: 7

## 3.8 Настройка автоматических подписей коммитов git

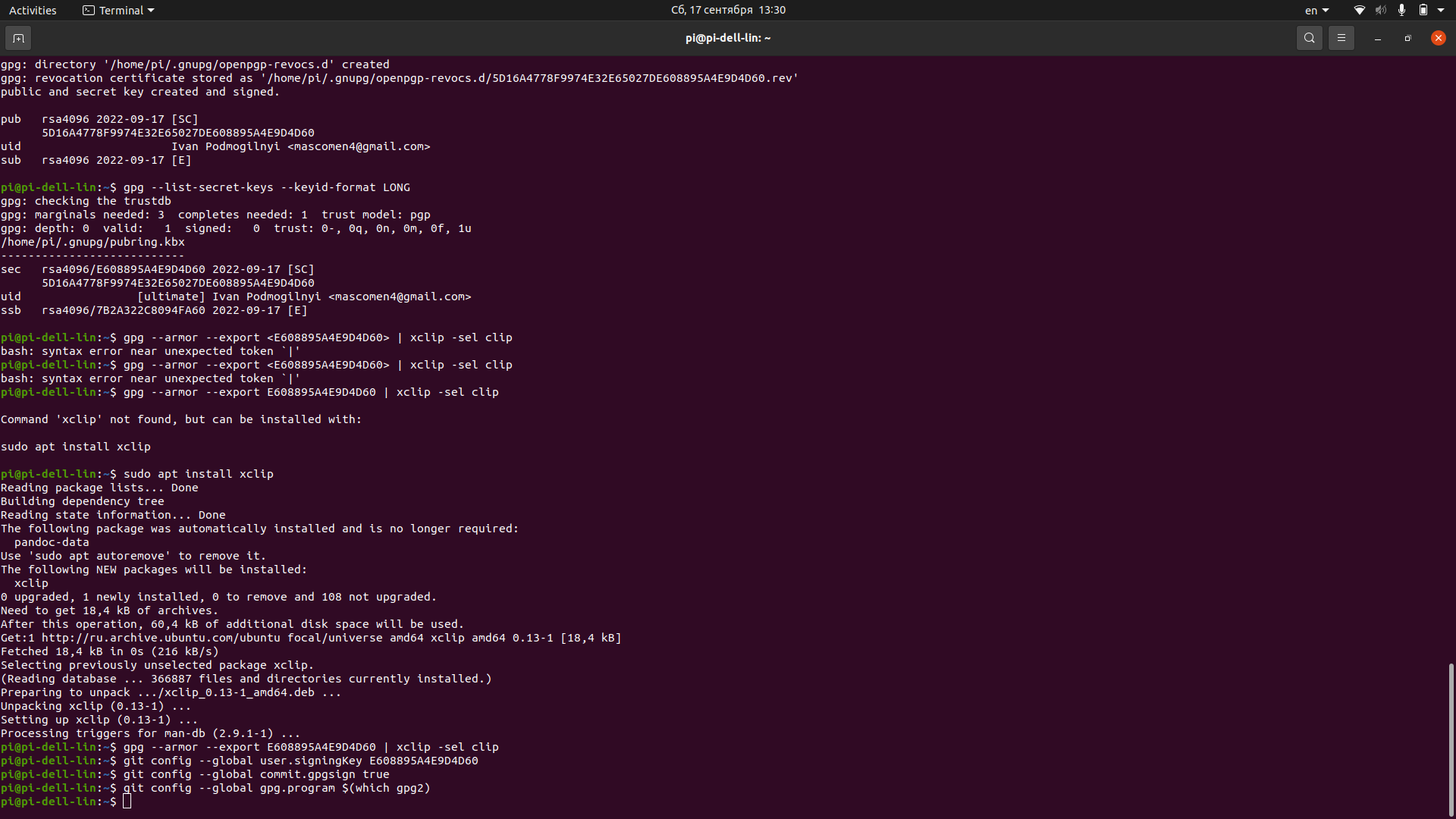


Figure 8: 8

## 3.9 Настройка gh

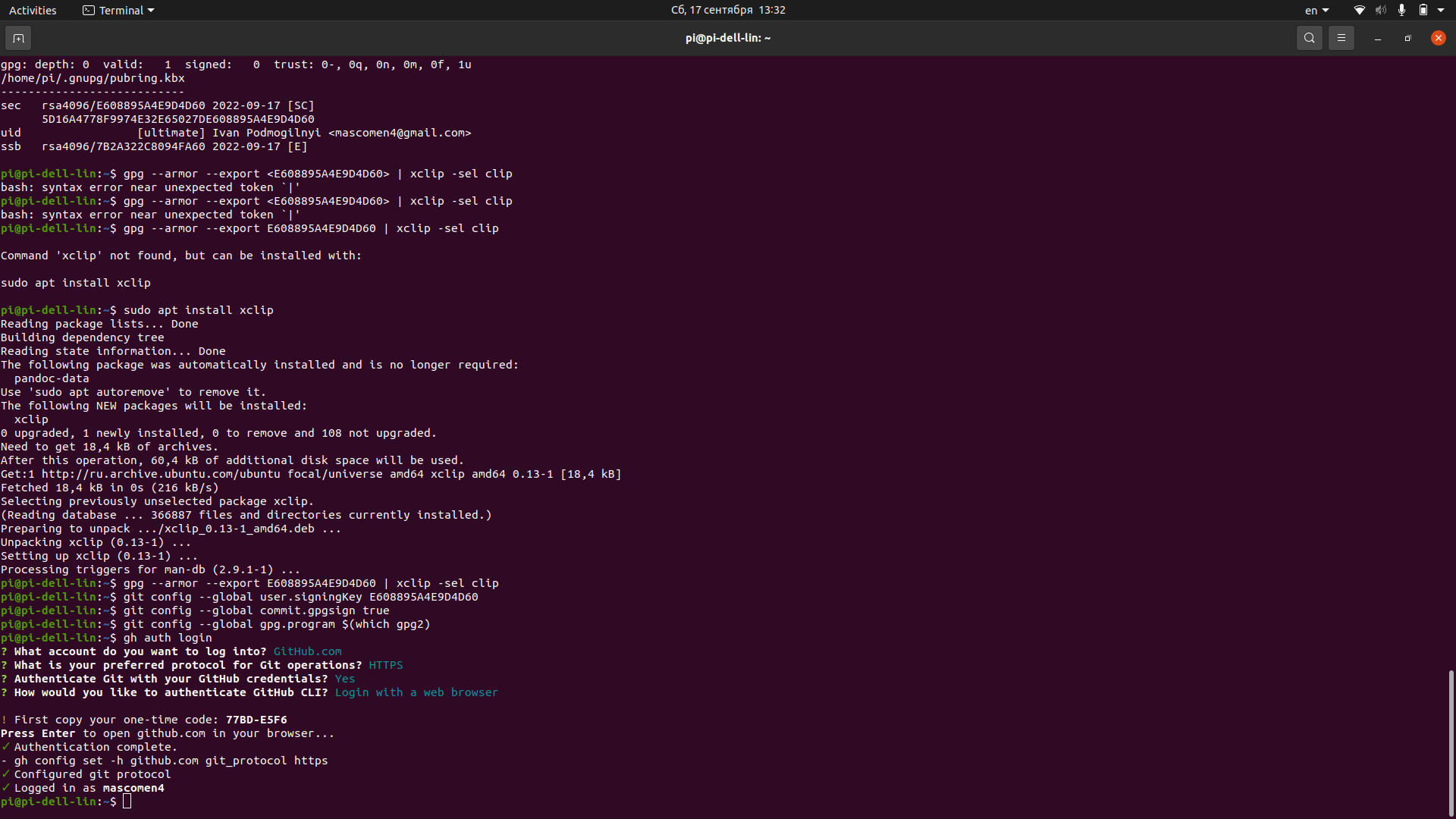


Figure 9: 9

## 3.10 Необходимо создать шаблон рабочего пространства.

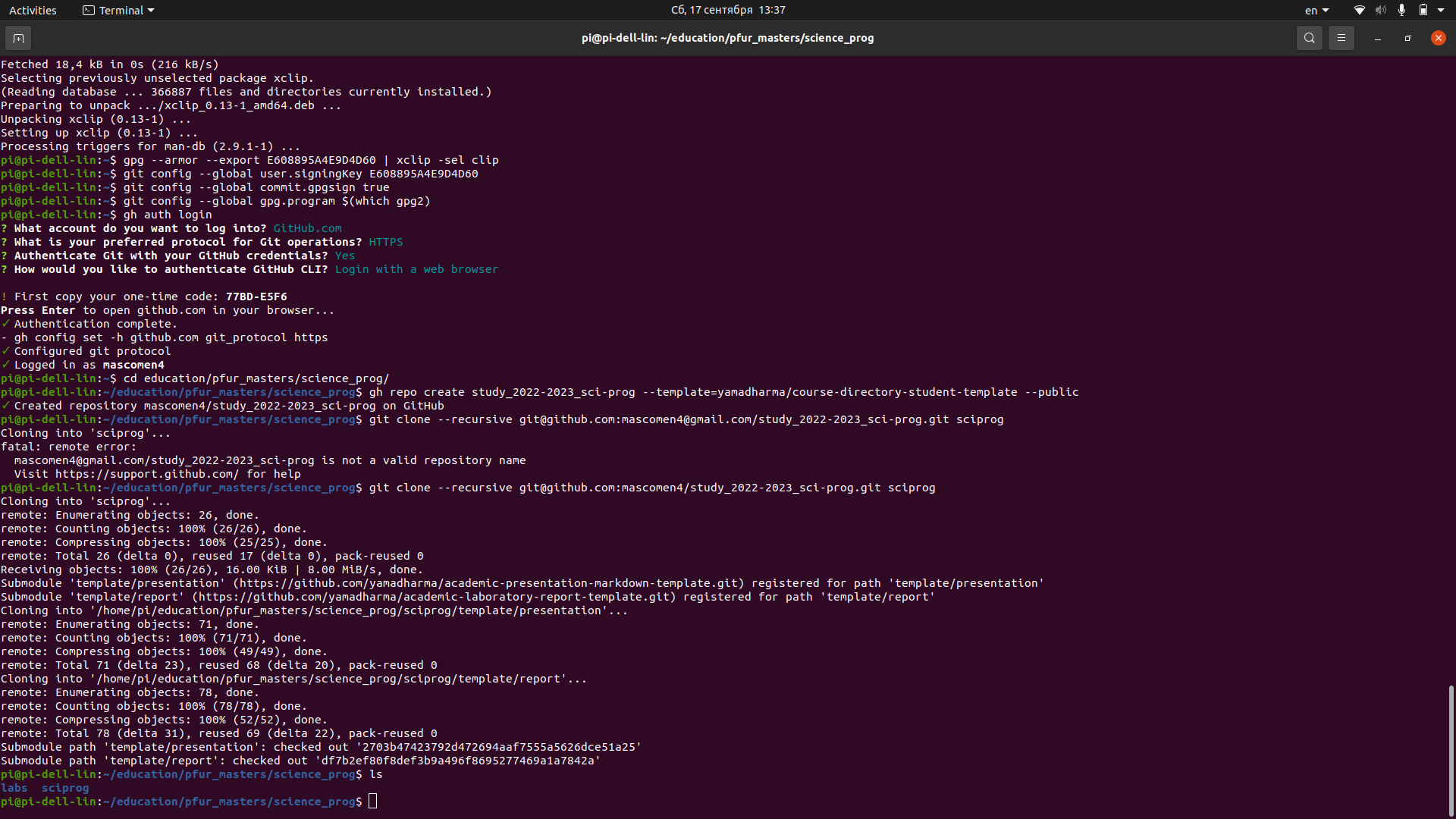


Figure 10: 10

## 3.11 Перейдите в каталог курса:

Удалите лишние файлы. Создайте необходимые каталоги

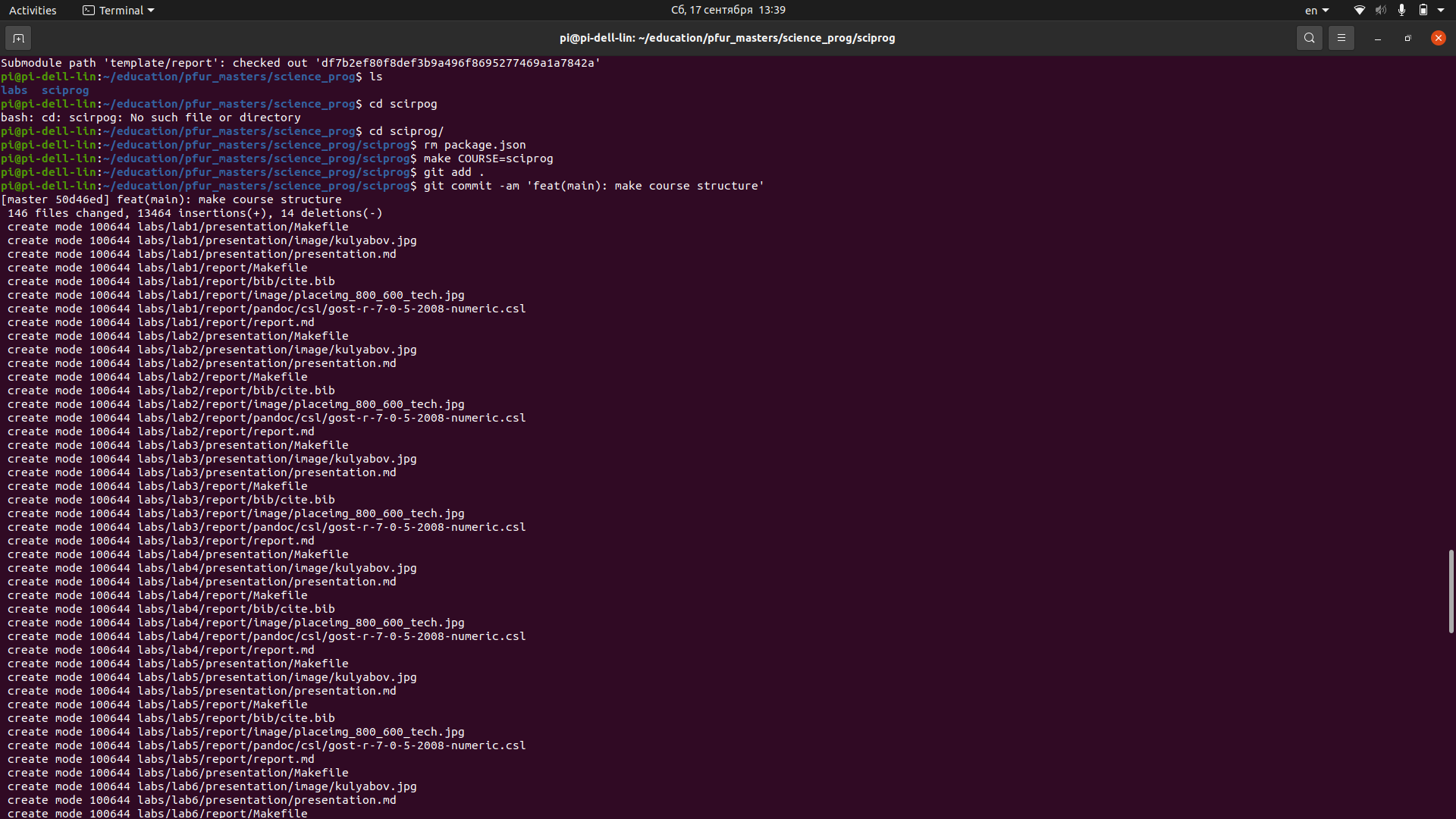


Figure 11: 11

## 3.12 Отправьте файлы на сервер

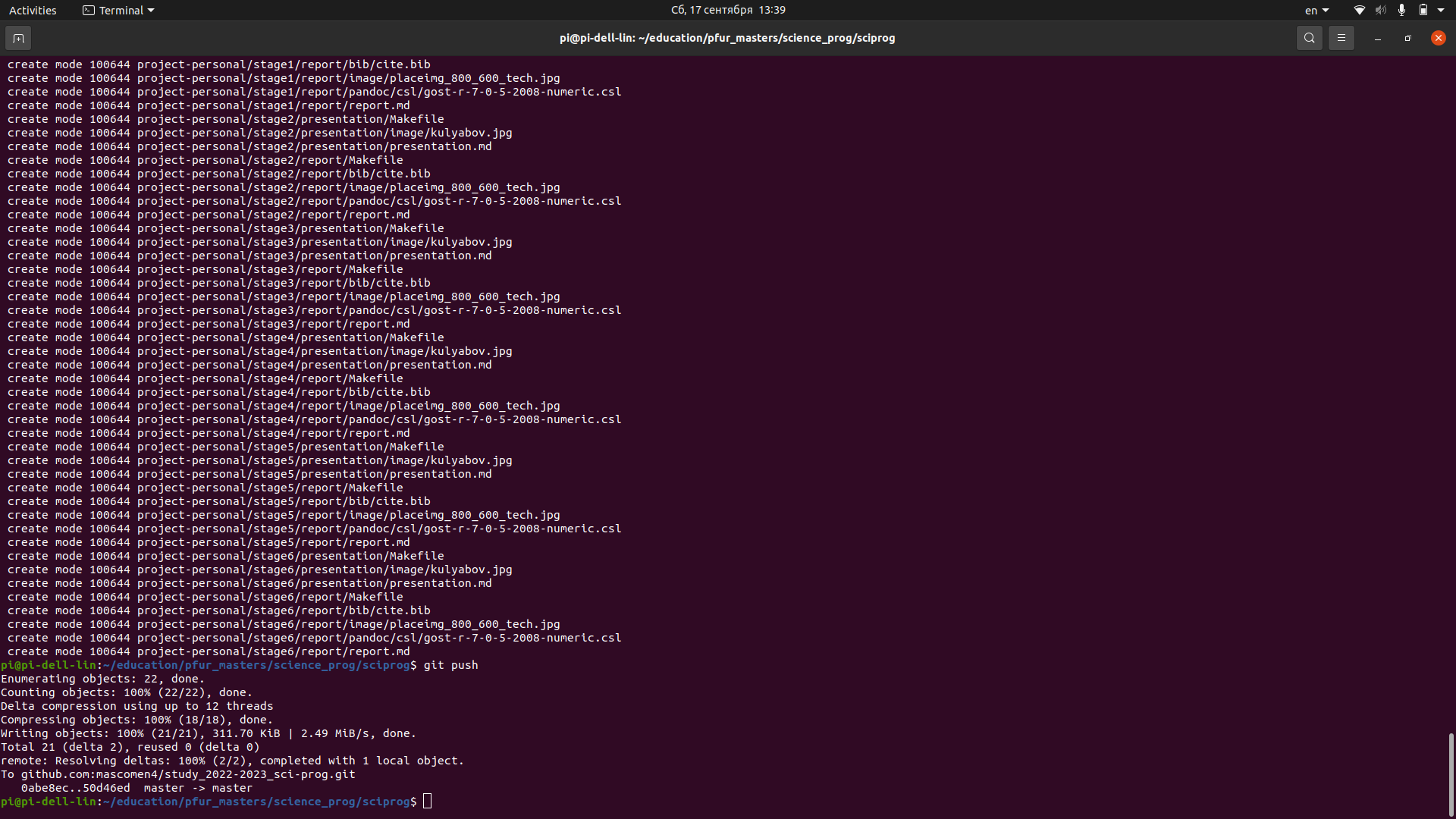


Figure 12: 12

Контрольные вопросы:

1. Система контроля версий предназначена для ведения истории изменений. Каждое изменение добавляется через коммиты, и составляется дерево коммитов. В любой момент времени можно вернуться на любую ноду дерева
2. Хранилище - удаленный сервер, на котором хранится проект с гит файлами, commit - изменение в проект, которре затем должно быть подтверждено командой git push. История - дерево всех коммитов. Рабочая копия - создается с помощью git clone, копия на локальной машине, в которую вносятся изменения. Они могут быть загружены на сервер через коммиты.
3. Централизованные системы используют единственный сервер, содержащий все версии файлов, и некоторое количество клиентов, которые получают файлы из этого централизованного хранилища. Примеры: CVS, Subversion и Perforce
4. При единоличной работе с хранилищем применяются такие же правила как и при работе с общим хранилищем (см. пункт 5)
5. При работе с общим хранилищем необходимо для каждой функции строго добавлять новую ветку feature, реализовывать её и слиять с веткой develop
6. Защищает исходный код от потери, обеспечивает командную работу, помогает отменить изменения, распределённая работа
7. git add, git commit, git push, git remote, git clone, git flow, git branch, git merge, git checkout, git pull, git init, git config
8. Если нужно вести систему контроля, но в целом мы не делимся кодом с командой, и нам не нужно иметь доступ к коду с разных устройств, которые практически никак не связаны с нашей локальной сетью, то можно использовать локальные репозитории. В противном случае нужно использовать удаленные репозитории
9. Ветви - отдельные истории в СКВ, которые позволяют вести разработки параллельно. Над двумя ветками могут работать две разные комманды, а затем их можно слить в одну
10. Через файл .gitignore. Тутор: https://git-scm.com/docs/gitignore Зачем? Потому что некоторые файлы могут быть слишком большими: бинарники, словари, видео, изображения

# 4 Выводы

Освоил на практике применение методов шифрования Цезаря и Атбаша.