Работа с удаленным репозиторием

1. Создание SSH – ключа

ssh-keygen –t rsa

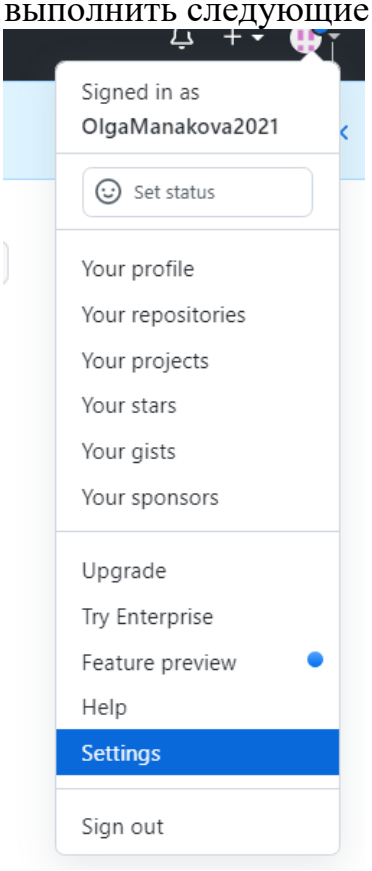
В процессе генерации необходимо ввести:
1. общее имя файла для частного и публичного ключей (например, <test>)
2. для повышения защиты ключей ввести пароль, повторить его еще раз. Процедура не является обязательной, но желательной.

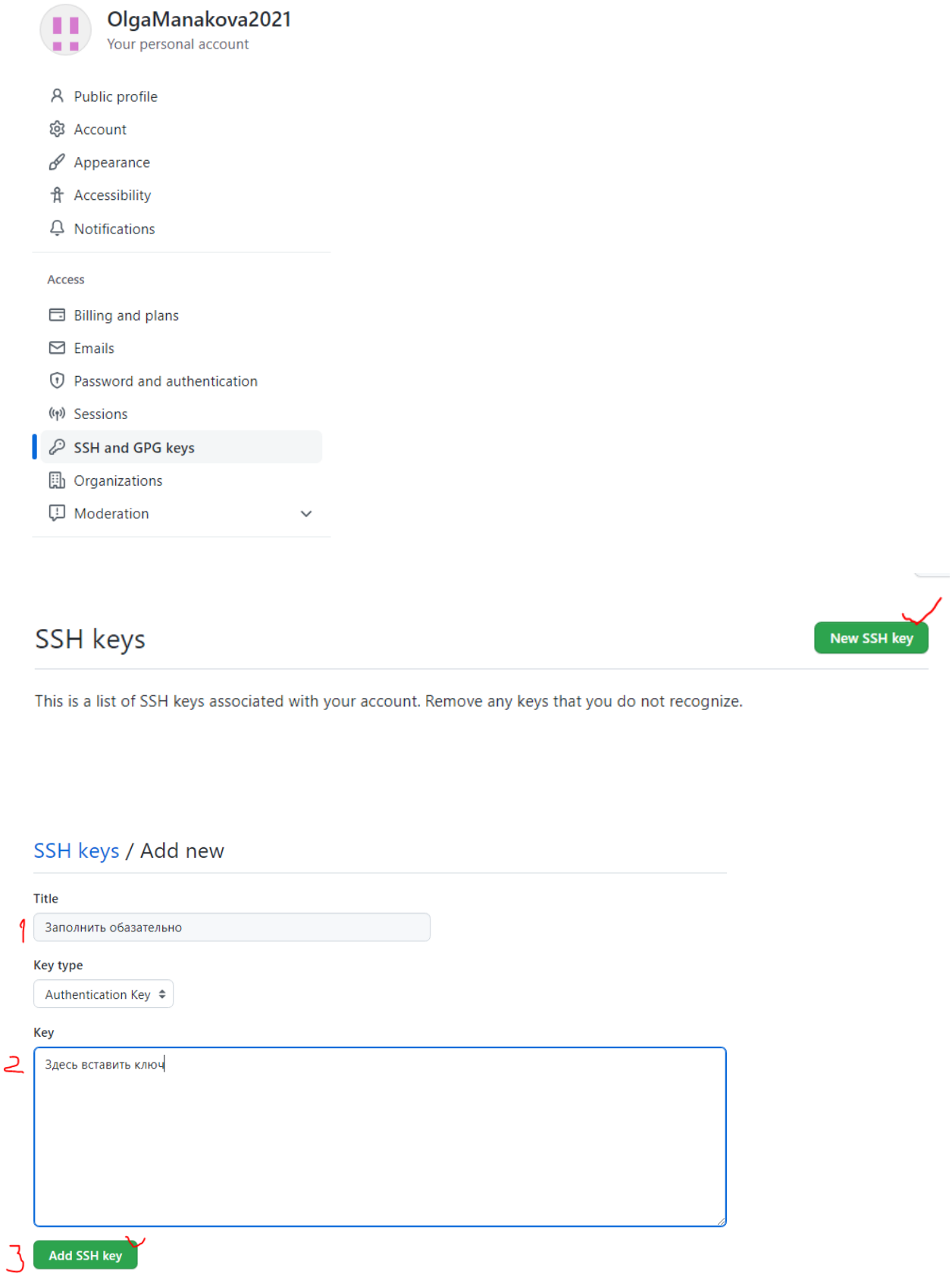
Два ключа появятся в корневой папке пользователя.

2. Зайти в файл публичного ключа:

cat <имя вашего ключа>.pub

Скопировать все содержимое для дальнейшего переноса в настройки аккаунта. Залогиниться в своем аккаунте на GitHub. Затем в настройках аккаунта выполнить следующие действия (см. рисунки):





3. Выполнить проверку подключения по SSH к сайту (в нашем случае это GitHub):

ssh –t git@github.com

После проверки будет два события:

- требование отпечатка ключа, написать yes
- вам будет отказано в доступе. Для предоставления доступа выполнить команду:

ssh-add <имя вашего ключа>

4. Создать репозиторий на GitHub (например, test_git_remote).

5. В папке с рабочим проектом выполнить:

git init

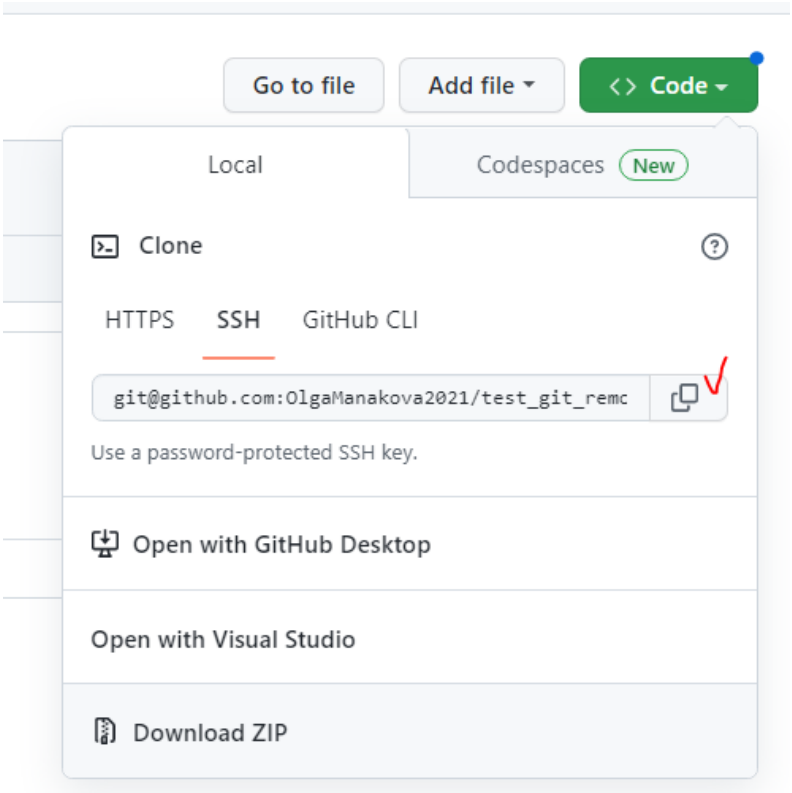
6. Убедиться в наличии файлов (особенно .git) и папок в проекте:

ls –a (-a показывает все файлы, в т.ч. скрытые)

7. Подключиться к созданному репозиторию:

git remote add <имя подключения> <ssh-ссылка на репозиторий>

где <имя подключения> - рекомендуется писать origin (или любое имя), для получения <ssh-ссылка на репозиторий> выполнить:



8. Убедиться, что по подключение состоялось

git remote –v

или

Git remote –verbose

9. По умолчанию в удаленном репозитории будет ветка main, в локальном – master. Ветки main нет в локальном репозитории, но ее можно добавить:

git fetch

Команда позволяет получить изменения из удаленного репозитория, при этом не записывая их в локальный репозиторий.

git fetch «подтянет» все ветки и файлы из удаленного репозитория. Убедимся в этом:

git switch main

git branch

10. Создадим текстовый файл на ветке main в удаленном репозитории на github.

11. Получим все актуальные изменения, ветки, папки, в т.ч. созданный файл в локальный репозиторий из удаленного:

git pull

12. Перенесем весь проект в удаленный репозиторий, т.е. выполним привычный набор команд:

```
git add .
git commit –m "…"
git push
```

при выполнении git push может появится ошибка. Чтобы ее исправить - выполнить копирование предложенной команды с параметрами. Затем выполнить ее.

!!!Внимание!!! По окончанию работы с удаленным репозиторием (в конце урока) необходимо удалить ssh-ключ на github, локальный файл с ключом не удалять.

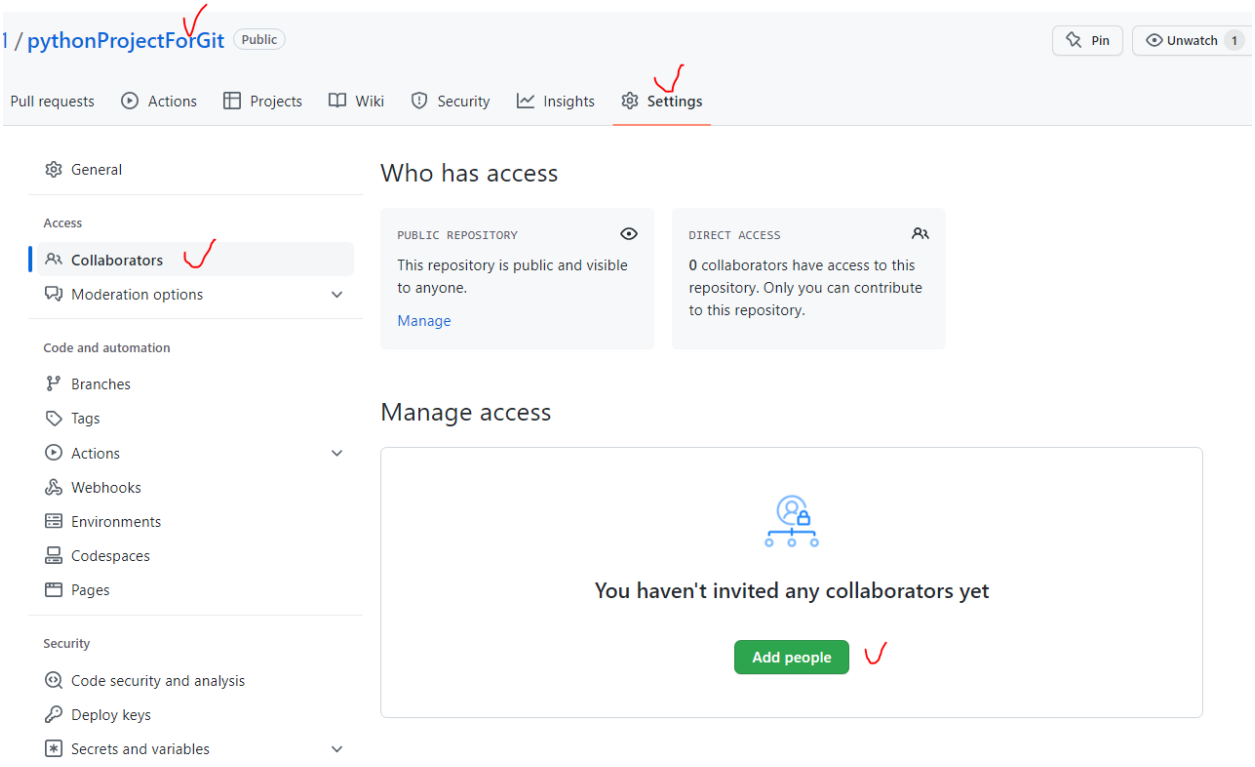
git push --set-upstream <имя подключения> <ветка>

<имя подключения> - origin,
<ветка> - master

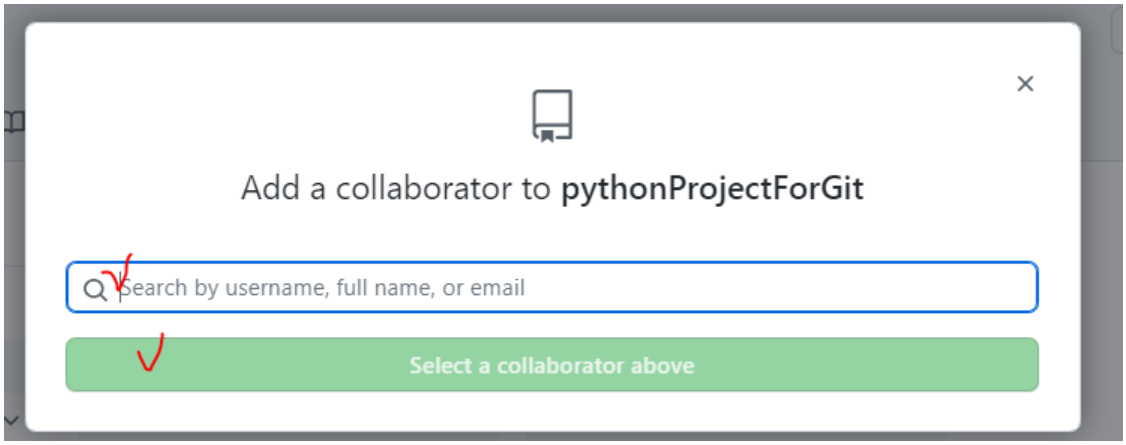
Совместная работа над проектом в git/github
(ssh для этого не нужен)

Перед выполнением нижеприведенных действий убедиться, что все коммиты завершены

1. Пригласим второго участника проекта через github:
   a. Зайти в репозиторий проекта
   b. Выбрать Settings

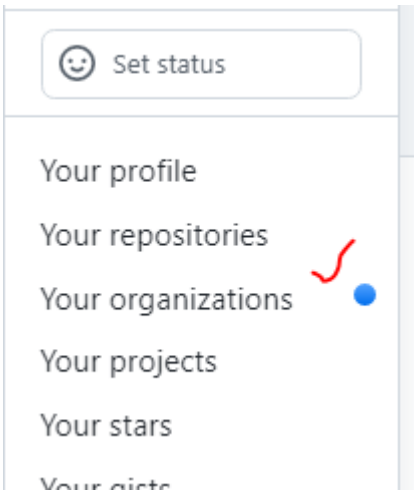


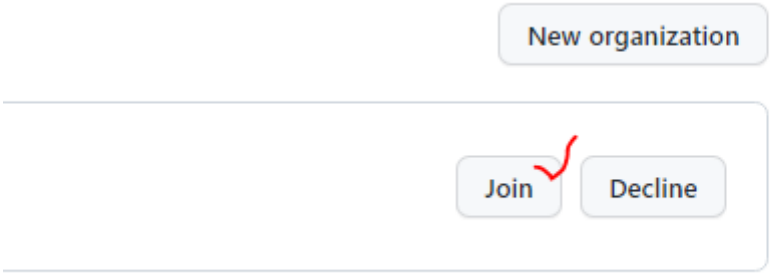
   c. В появившемся окне



   d. Ждем подтверждения приглашения от второго участника.

2. В github второго участника придет приглашение:





После нажатия на *Join* второй участник присоединится к проекту. Затем он должен выполнить клонирование данного проекта на свой ПК:

git clone <http-ссылка на репозиторий>

3. Владелец проекта проекта создает новый файл (test.txt) и пушит его в удаленный репозиторий (команды подаются через git).
4. Второй участник выполняет

git pull

и убеждается (команда ls), что файл test.txt попал в его локальный репозиторий.

5. Второй участник проекта создает новый файл (test1.txt) и пушит его в удаленный репозиторий (команды подаются через git).
6. Владелец проекта выполняет

git pull

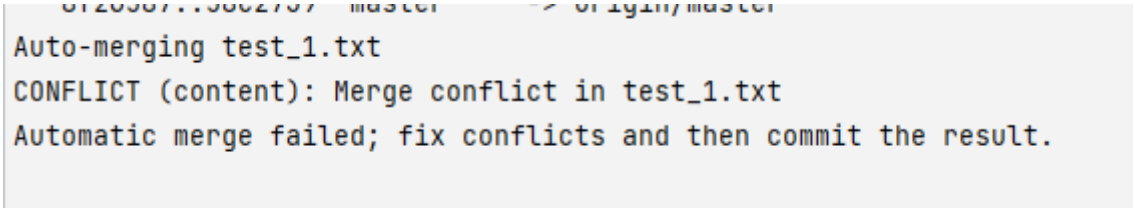
и убеждается (команда ls), что файл test1.txt попал в его локальный репозиторий.

Разрешение конфликтов

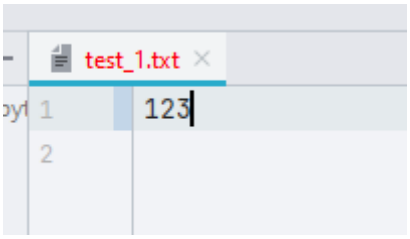
Конфликт возникает, когда участники проекта одновременно вносят разную информацию в один файл.

1. Создадим конфликт на стороне владельца проекта:
    a. Второй участник проекта начинает работу с файлом test.txt и записывает в первой строке – 123. Выполняет add, commit, push.
    b. Владелец проекта в этот же файл (test.txt) вносит в первую строку – 456. Выполняет add, commit, pull и получает конфликт

И в консоли:

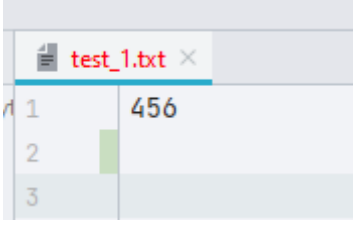


2. Участники проекта теперь должны договорится о том, какая информация является верной:
    a. Если верной оказывается 123, то владелец проекта редактирует файл до состояния



И выполняет add, commit, pull

    a. Если верной оказывается 456, то владелец проекта редактирует файл до состояния



и выполняет add, commit, push

3. Самостоятельно создать и устранить конфликт на стороне второго участника.
4. Поменяется ролями и выполнить все действия по совместной работе сначала.