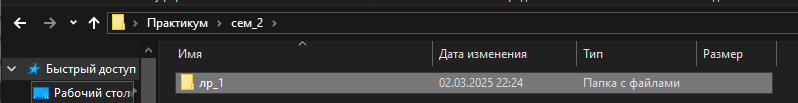
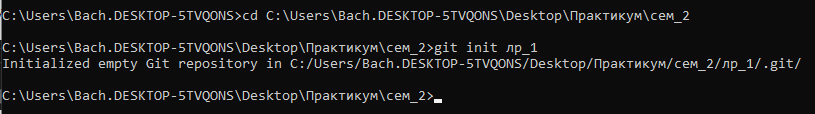
* 1. Установить на компьютер графический клиент Git.



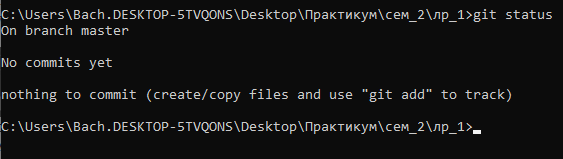
* 1. Создайте в своей домашней папке (или в любой другой на ваш выбор) каталог, который будет содержать файлы нового программного проекта.



* 1. Инициализируйте в этой директории репозиторий гит при помощи команды git init. Обратите внимание на появление в этой папке скрытой подпапки с названием .git. Если вы ее не видите, то скорее всего, у вас отключено отображение скрытых папок.

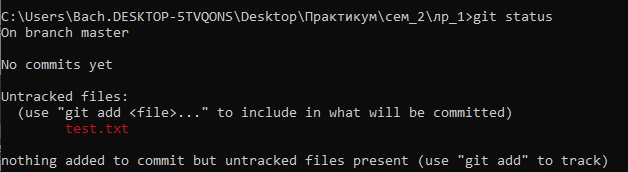


* 1. Выполните в репозитории команду git status. Проинтерпретируйте полученное сообщение.



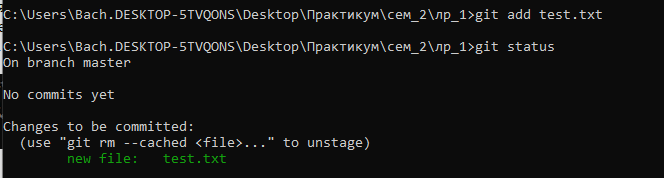
Пока что нет ни одного коммита

* 1. Создайте файл для исходного текста программы. Выполните команду git status.



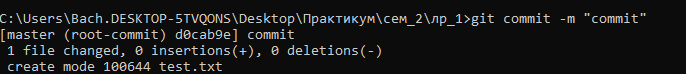
Прописываем команду git status, теперь у нас есть неотслеживаемый файл test.py – «Гит» видит его, но, если попытаться сохранить их версию сейчас, «Гит» этого не сделает. Для этого файлы нужно подготовить к сохранению.

* 1. Добавьте созданный файл под версионный контроль при помощи команды git add. Еще раз выполните git status.



Прописываем команду git add и вносим созданный файл под версионный контроль. Еще раз прописываем git status – теперь файл test.py индексирован.

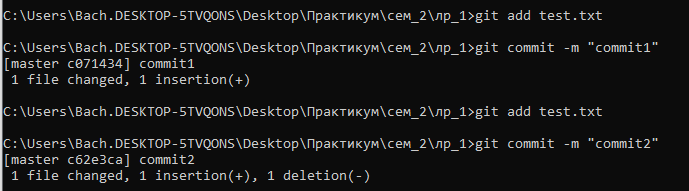
* 1. Когда все файлы готовы к сохранению, сделаем наш первый коммит — зафиксируем все сделанные изменения в «боевой версии». Делается это командой git commit c опцией -m. После -m идёт название коммита в кавычках.

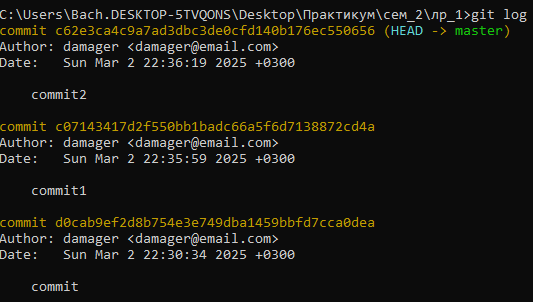


Делается это командой git commit c опцией -m. После -m идёт название коммита в кавычках.

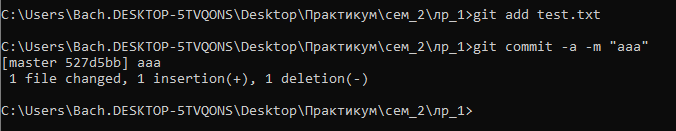
* 1. Сделайте еще несколько коммитов. Выполните команду git log для

просмотра истории коммитов.



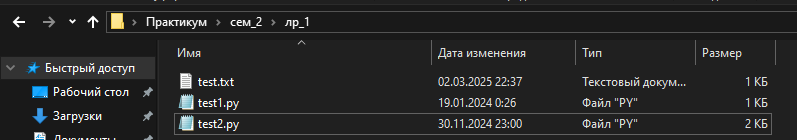


* 1. Сделайте так, чтобы при коммите измененные файлы автоматически добавлялись в коммит.

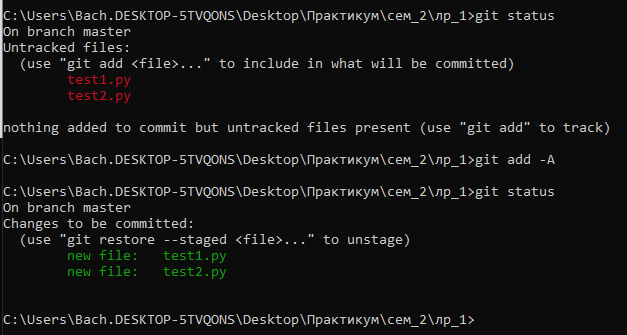


Чтобы при коммите измененные файлы автоматически добавлялись в коммит, используется команда git commit с опцией -a, которая автоматически индексирует все измененные и удаленные файлы, которые уже отслеживаются Git, но не добавляет новые файлы, о которых Git еще не знает.

* 1. Добавьте еще несколько файлов с исходным текстом программы.

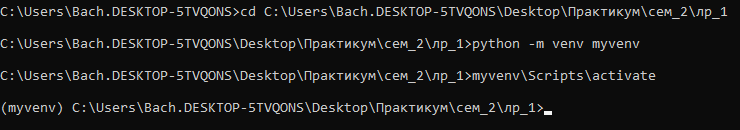


* 1. Добавьте все новые файлы под версионный контроль одной командой.



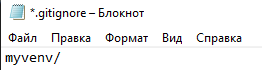
У нас появилось два неотслеживаемых файла, мы вносим их под версионный контроль с помощью команды git add с опцией -А, которая используется для добавления всех изменений, включая новые файлы, измененные файлы и удаленные файлы.

* 1. Инициализируйте в рабочей директории виртуальное окружение

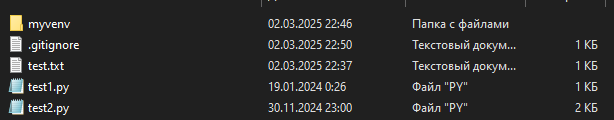


* 1. Добавьте созданную служебную папку в файл .gitignore. Проверьте,

что они не добавляются в репозитории при добавлении новых файлов с исходным кодом.

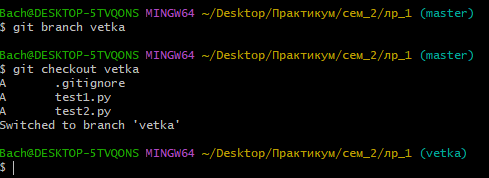


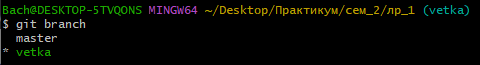
Выполняем команду git add -A и видим, что файлы виртуального окружения не добавляются в репозитории.



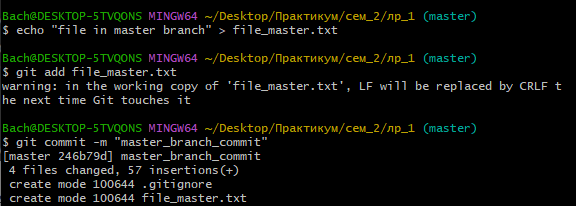
* 1. Создайте новую тематическую ветку git branch. Перейдите в нее с

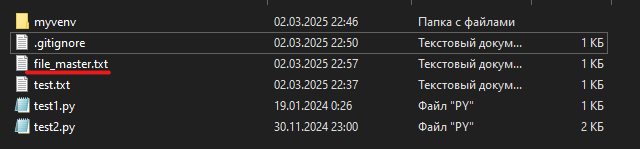
помощью git checkout. Выведите на экран список всех веток.

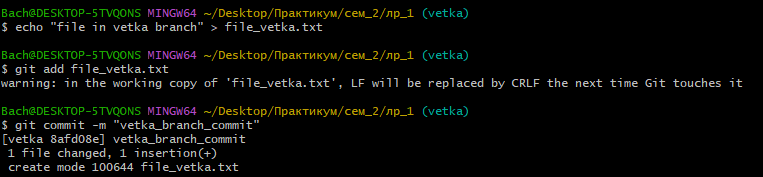


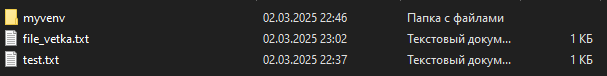


* 1. Сделайте несколько коммитов в основную и тематическую ветки









При переключении между ветками в папке отображаются только файлы, созданные в той ветке, в которой мы находимся.

* 1. Слейте изменения в основную ветку с помощью git merge. Если

произошел конфликт слияния, разрешите его и завершите слияние с

помощью git commit.

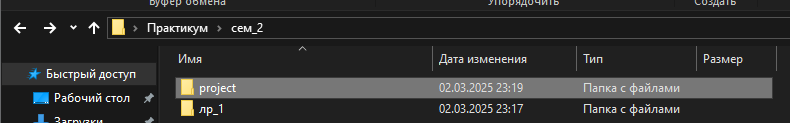


* 1. При получении в процессе разработки программы в стабильно работающем состоянии, слейте это состояние в основную ветку и добавьте к коммиту слияния пометку с номером релиза.

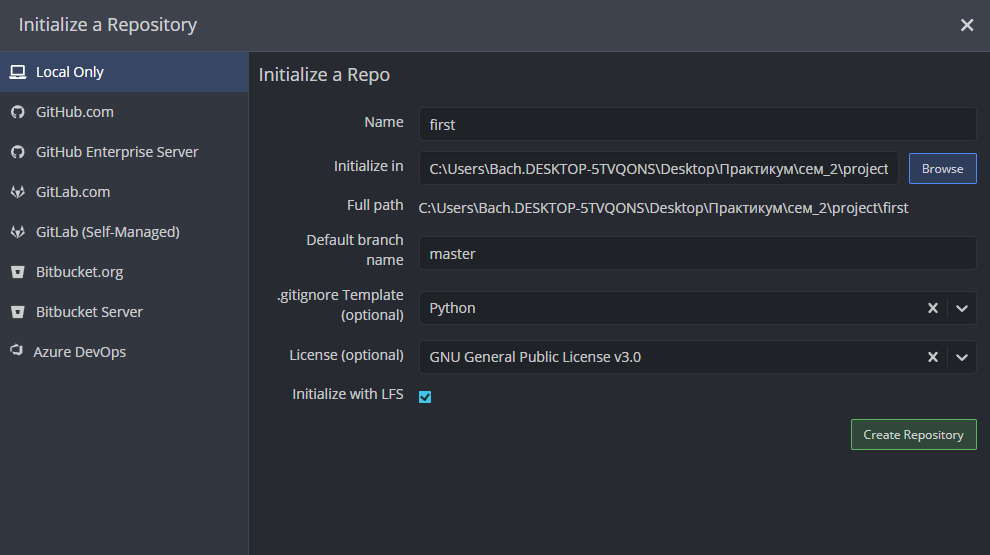
**Упражнение 2.** Работа GUI git клиент GitKraken.

GitKraken – это кроссплатформенный, удобный и высокоэффективный GUI git клиент на Linux, Windows и macOS.

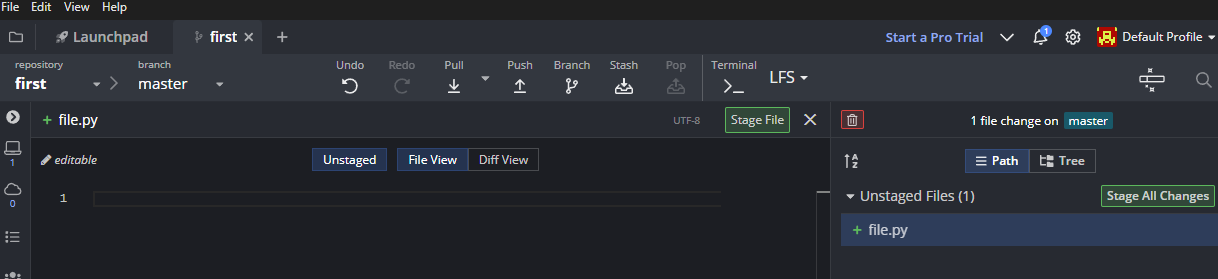
1. Выберите тематику программы, которую собираетесь написать. Создайте для нее рабочую директорию



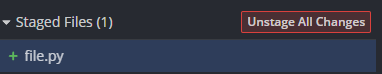
1. Инициализируйте в рабочей директории репозиторий.



1. Создайте файл для исходного текста программы.

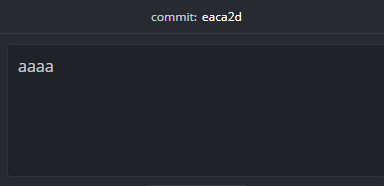


1. Добавьте созданный файл под версионный контроль.

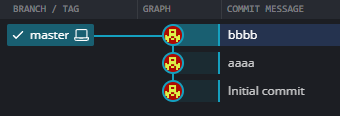




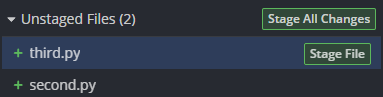
1. Сделайте начальный коммит.



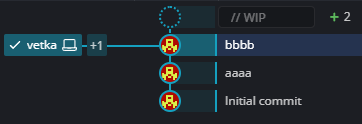
1. Сделайте еще несколько коммитов.



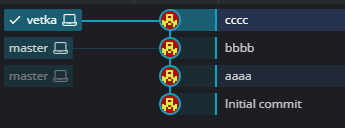
1. Добавьте еще несколько файлов с исходным текстом программы.



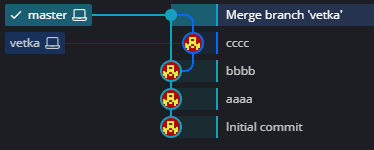
1. Создайте новую тематическую ветку.



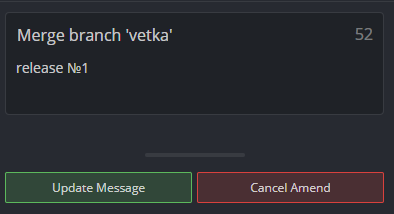
1. Сделайте несколько коммитов в основную и тематическую ветки.

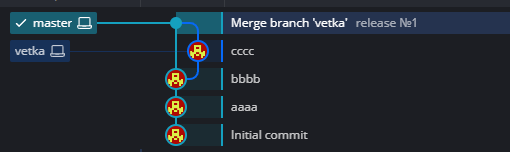


1. Слейте изменения в основную ветку. Если произошел конфликт слияния, разрешите его и завершите слияние.



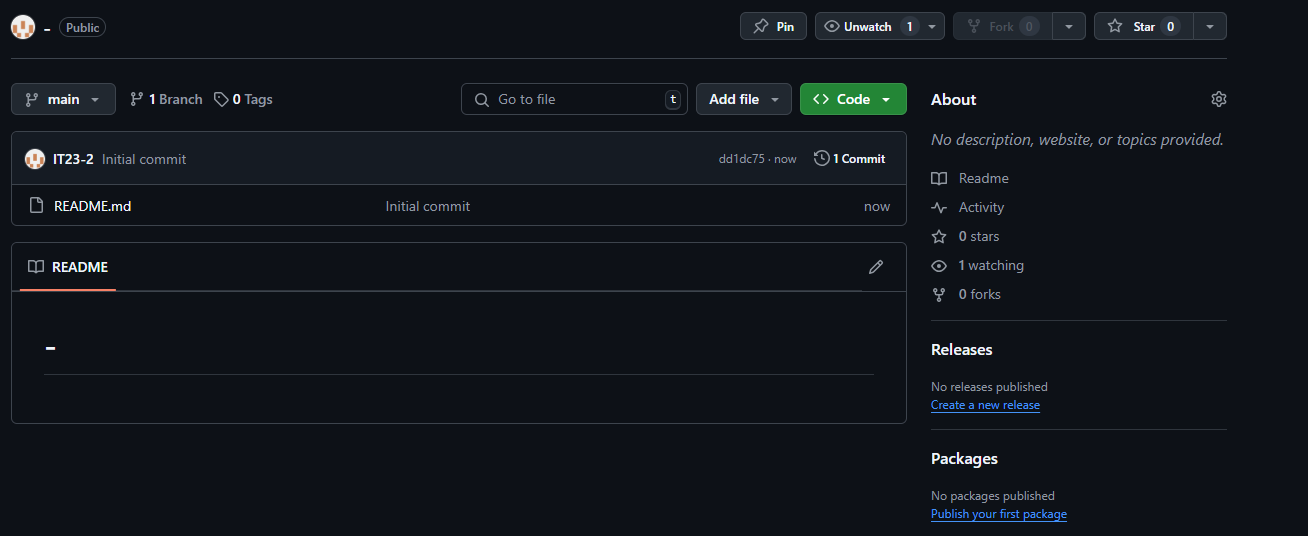
1. При получении в процессе разработки программы в стабильно работающем состоянии, слейте это состояние в основную ветку и добавьте к коммиту слияния пометку с номером релиза.





**Упражнение 3.** Работа с удаленными репозиториями и GitHub.

1. Зарегистрироваться на сайте github.com и создать новый пустой репозиторий



1. Создать в этом репозитории файл Readme

Если вы работаете с git:

– Склонировать созданный удаленный репозиторий в директорию ~/git/test

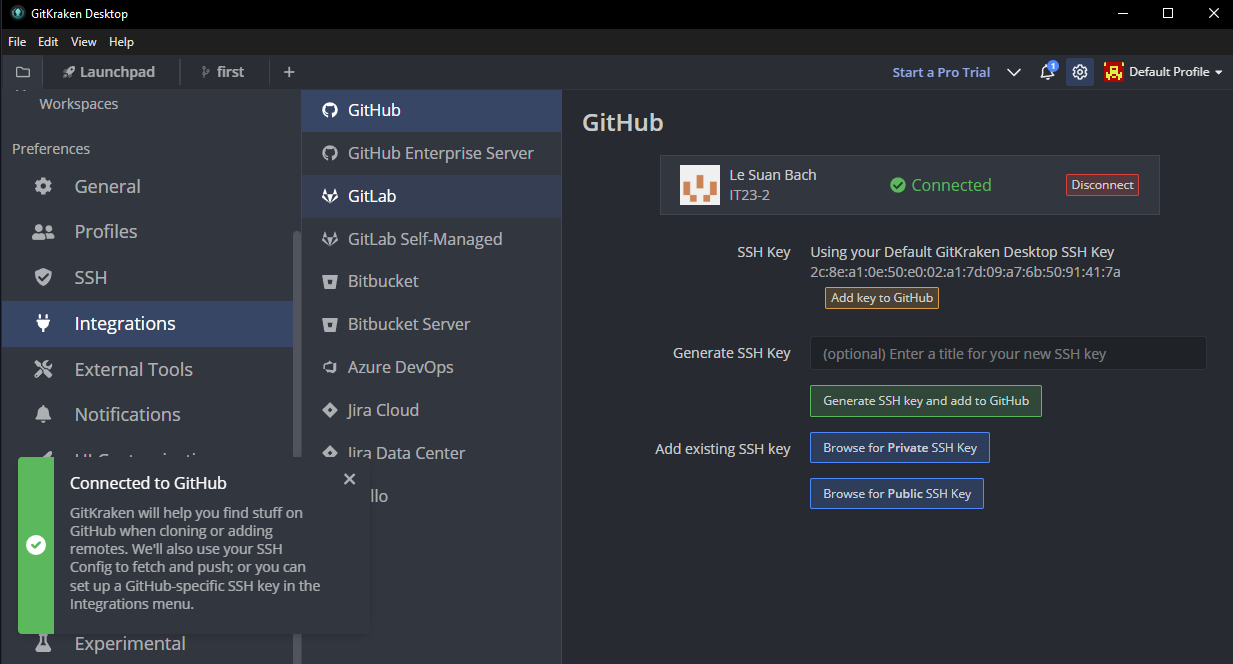
– На локальной машине пишем скрипт ~/git/test/backup.sh, с произвольным содержанием

– Фиксируем скрипт в репозитории (делаем коммит)

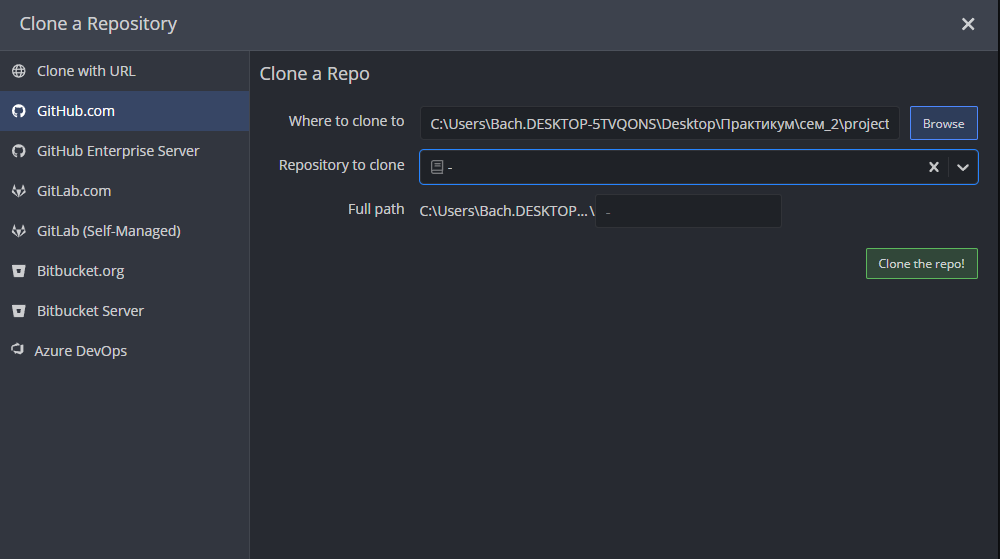
– Обновляем удаленный репозиторий репозиторий (делаем пуш)

Если вы работаете с gitKraken:

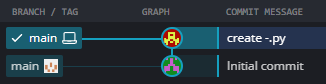
Привязываем свой аккаунт GitHub к GitKraken:



Клонируем заранее созданный репозиторий -:



Фиксируем скрипт в репозитории (делаем коммит)



Обновляем удаленный репозиторий (делаем пуш)

