**Лабораторная работа №1**

Приёмы работы в современных системах контроля версий.

**Цель работы:** познакомиться на практике с основными приемами работы в современных системах контроля версий.

**Задачи для выполнения:**

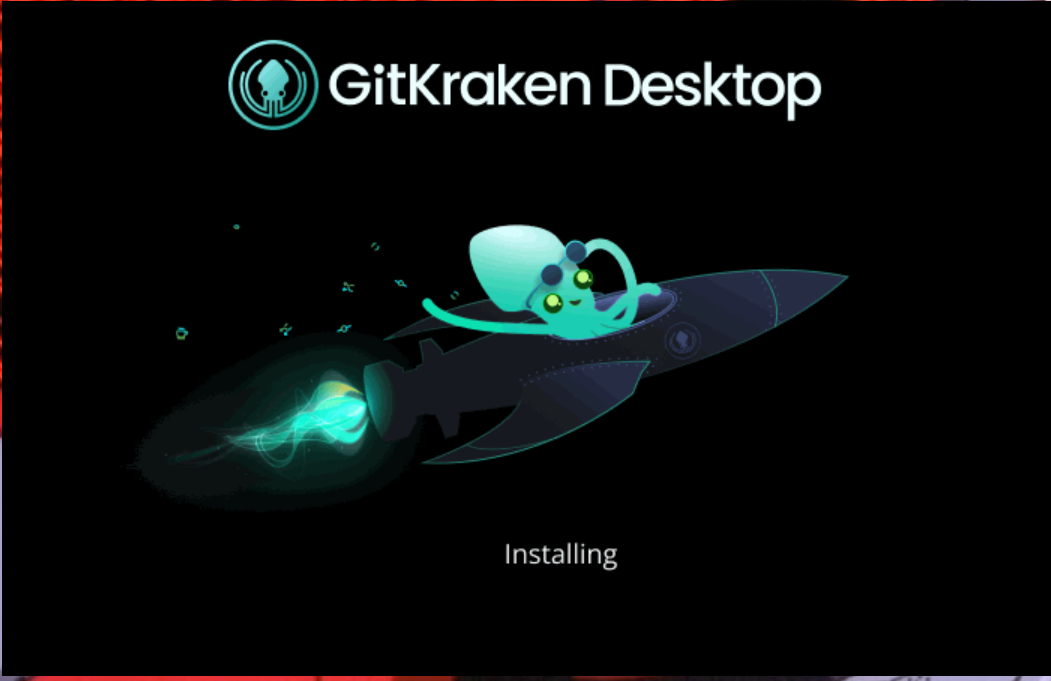
1. Работа с системой контроля версий Git.

2. Работа GUI git клиент GitKraken.

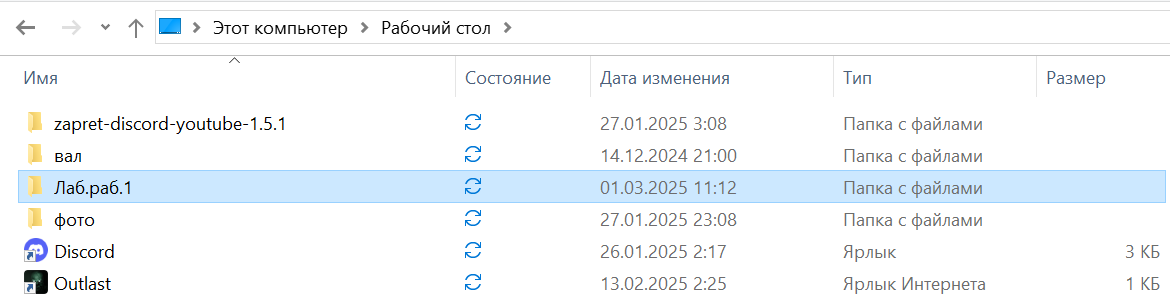
3. Работа с удаленными репозиториями и GitHub.

**Упражнение 1. Работа с системой контроля версий Git.**

1.1. Устанавливаем GitKraken (графический клиент Git).

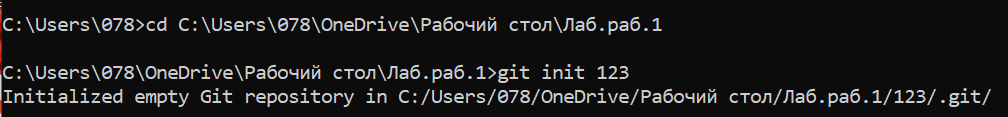


1.2. Создаём в своей домашней папке каталог, который будет содержать файлы нового программного проекта.

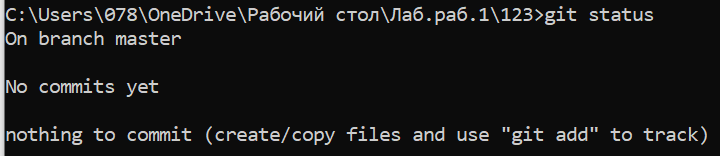


1.3. Выбрала тему <…>.

1.4. Инициализируем в этой директории репозиторий гит при помощи команды git init.

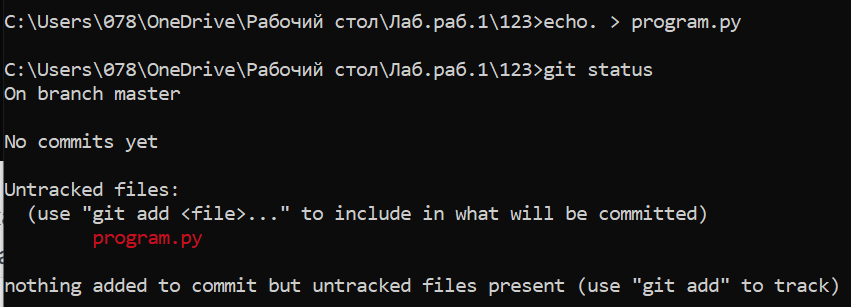


1.5. Выполняем в репозитории команду git status.



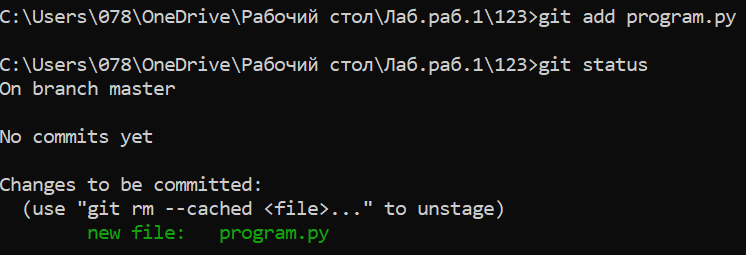
Видим, что пока что нет ни одного коммита

1.6. Создаём файл для исходного текста программы. Выполняем команду git status.



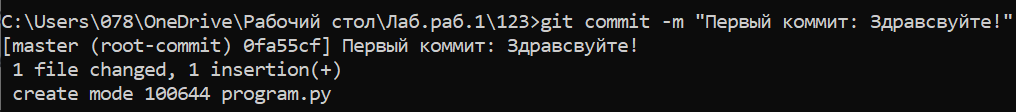
Имя моего файла: program.py

1.7. Добавляем созданный файл под версионный контроль при помощи команды git add. Еще раз выполняем команду git status.

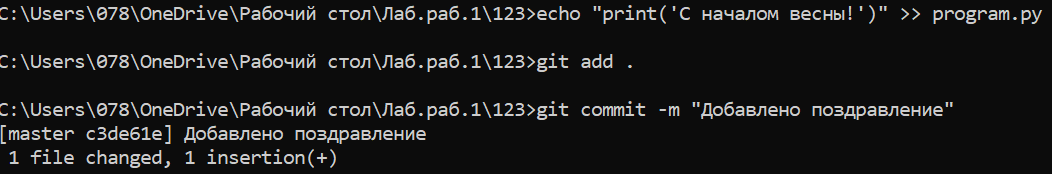


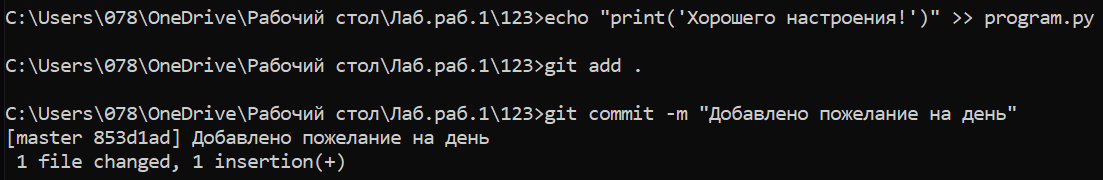
Мы подготовили файл к сохранению с помощью команды git add (при её использовании нужно указать имя файла, судьбу которого мы хотим зафиксировать в текущем виде, в моем случае файл program.py).

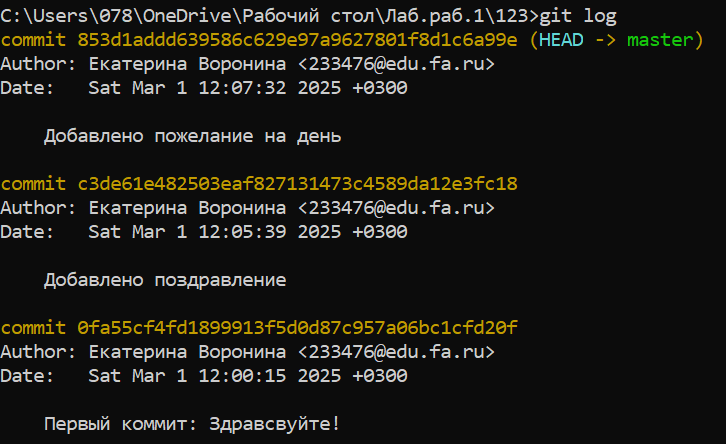
1.8. Когда все файлы готовы к сохранению, сделаем наш первый коммит — зафиксируем все сделанные изменения в «боевой версии». Делается это командой git commit c опцией -m. После -m идёт название коммита в кавычках.



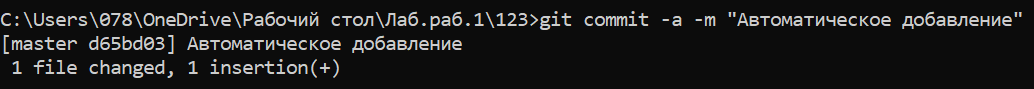
1.9. Сделайте еще несколько коммитов. Выполните команду git log для просмотра истории коммитов.





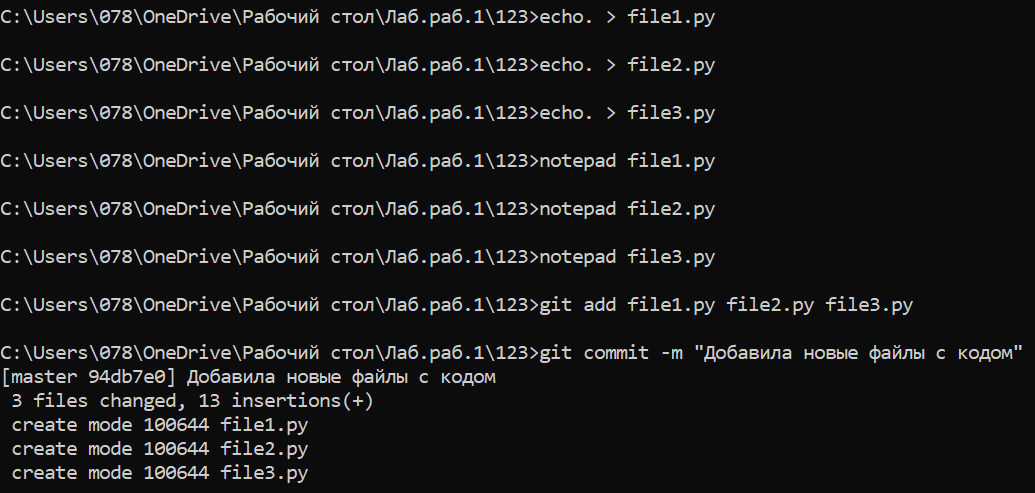


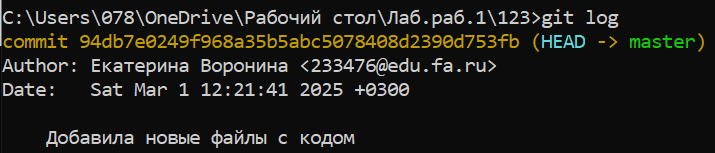
1.10. Сделайте так, чтобы при коммите измененные файлы автоматически добавлялись в коммит.

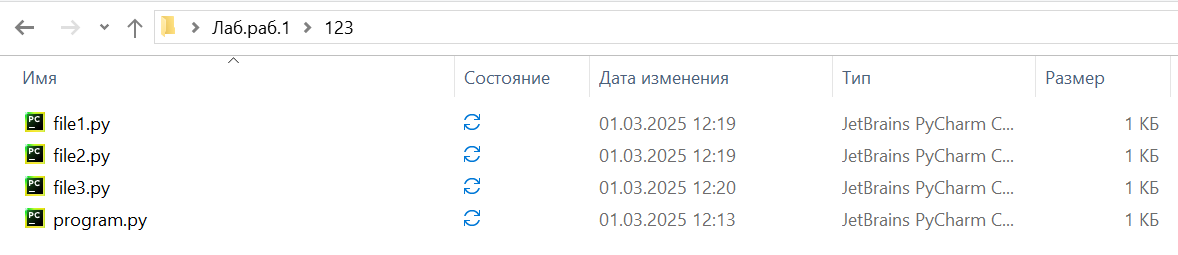


Для этого мы используем -a, т.к. эта опция позволяет Git автоматически добавлять все отслеживаемые файлы, которые были изменены, в индекс перед коммитом.

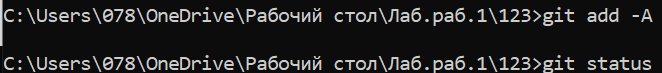
1.11. Добавьте еще несколько файлов с исходным текстом программы.



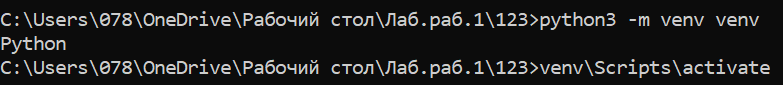




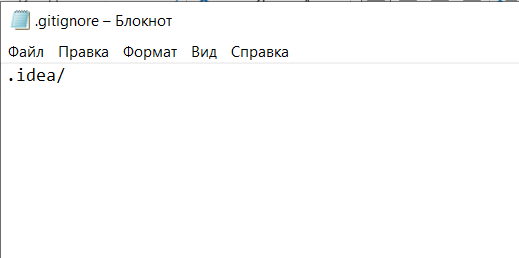
1.12. Добавляем все новые файлы под версионный контроль одной командой.



1.13. Инициализируем в рабочей директории виртуальное окружение.

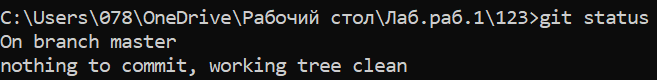


1.14. Добавляем созданную служебную папку в файл.gitignore. Проверяем, что они не добавляются в репозитории при добавлении новых файлов с исходным кодом.

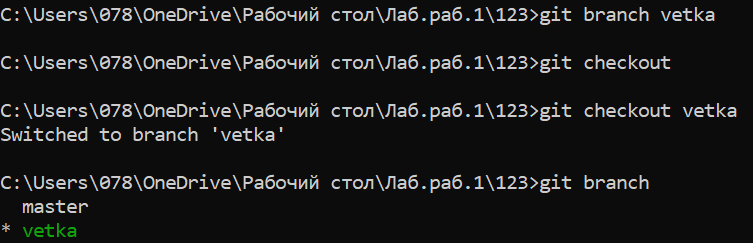






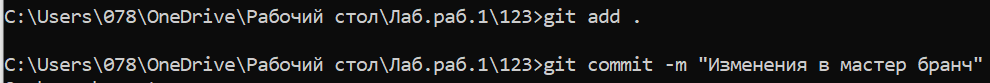


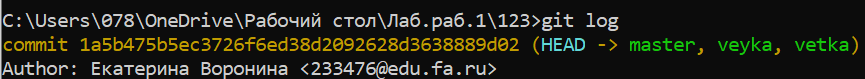
1.15. Создайем новую тематическую ветку git branch. Переходим в нее с помощью git checkout. Выводим на экран список всех веток.

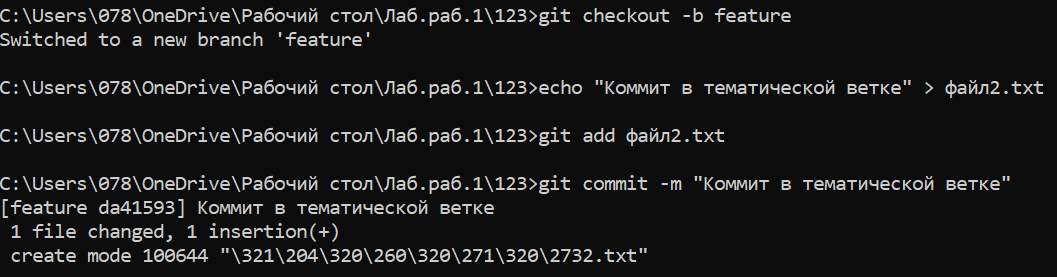


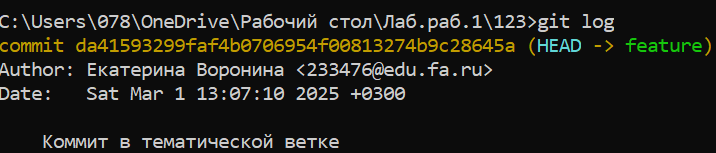
1.16. Делаем несколько коммитов в основную и тематическую ветки.



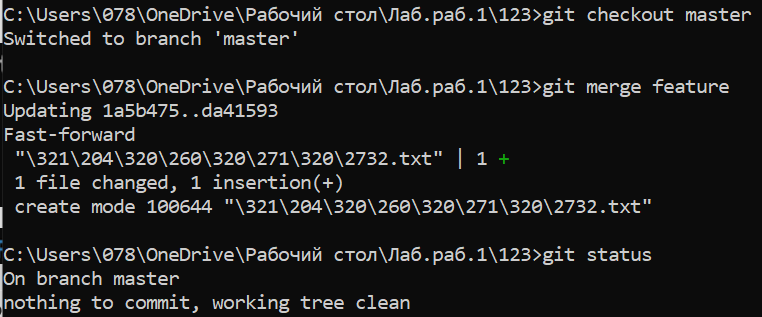




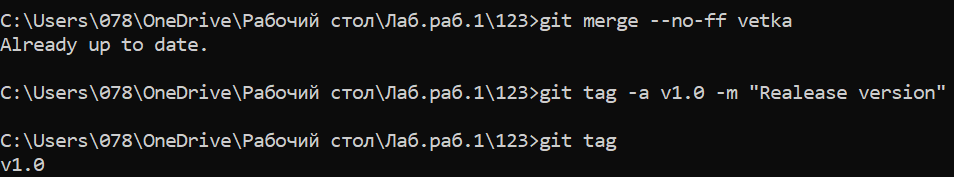




1.17. Сливаем изменения в основную ветку с помощью git merge. Если произошел конфликт слияния, разрешаем его и завершаем слияние с помощью git commit.

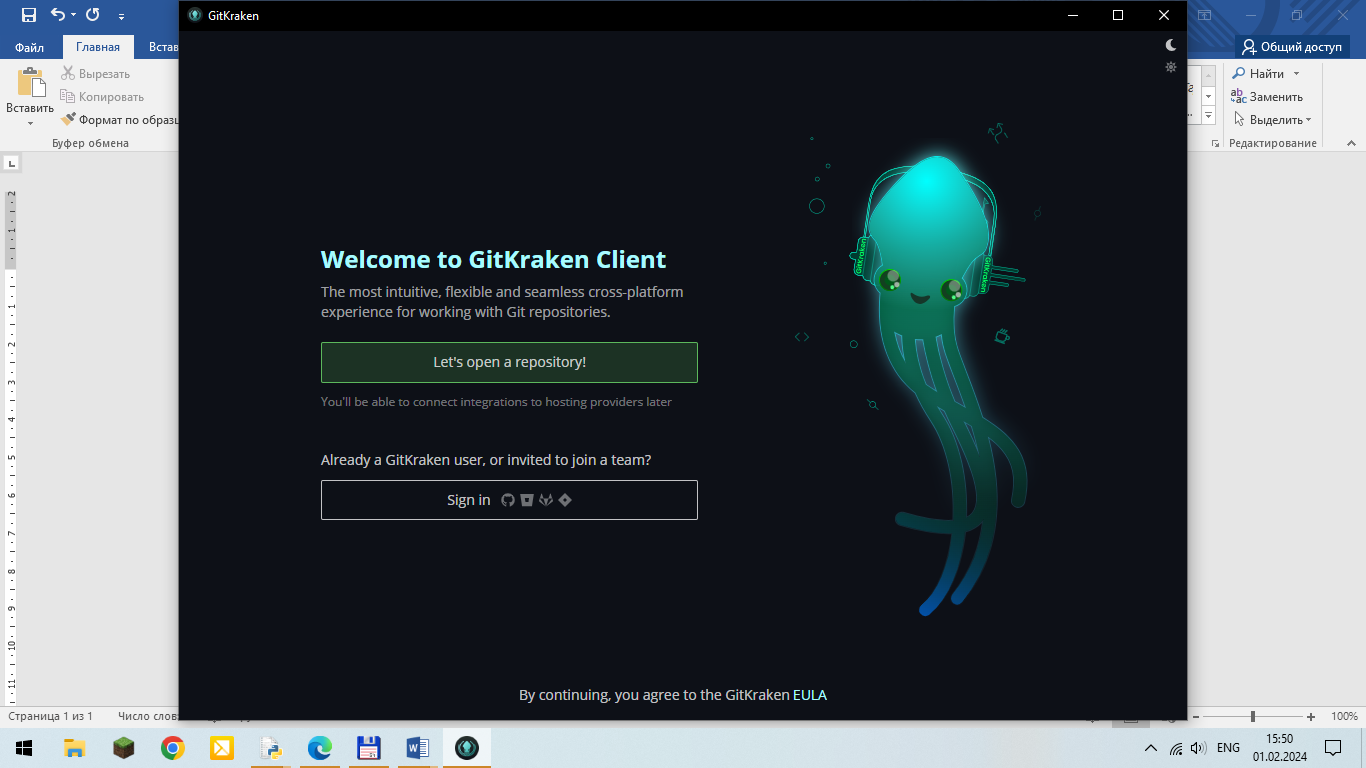


1.18. При получении в процессе разработки программы в стабильно работающем состоянии, сливаем это состояние в основную ветку и добавляем к коммиту слияния пометку с номером релиза.

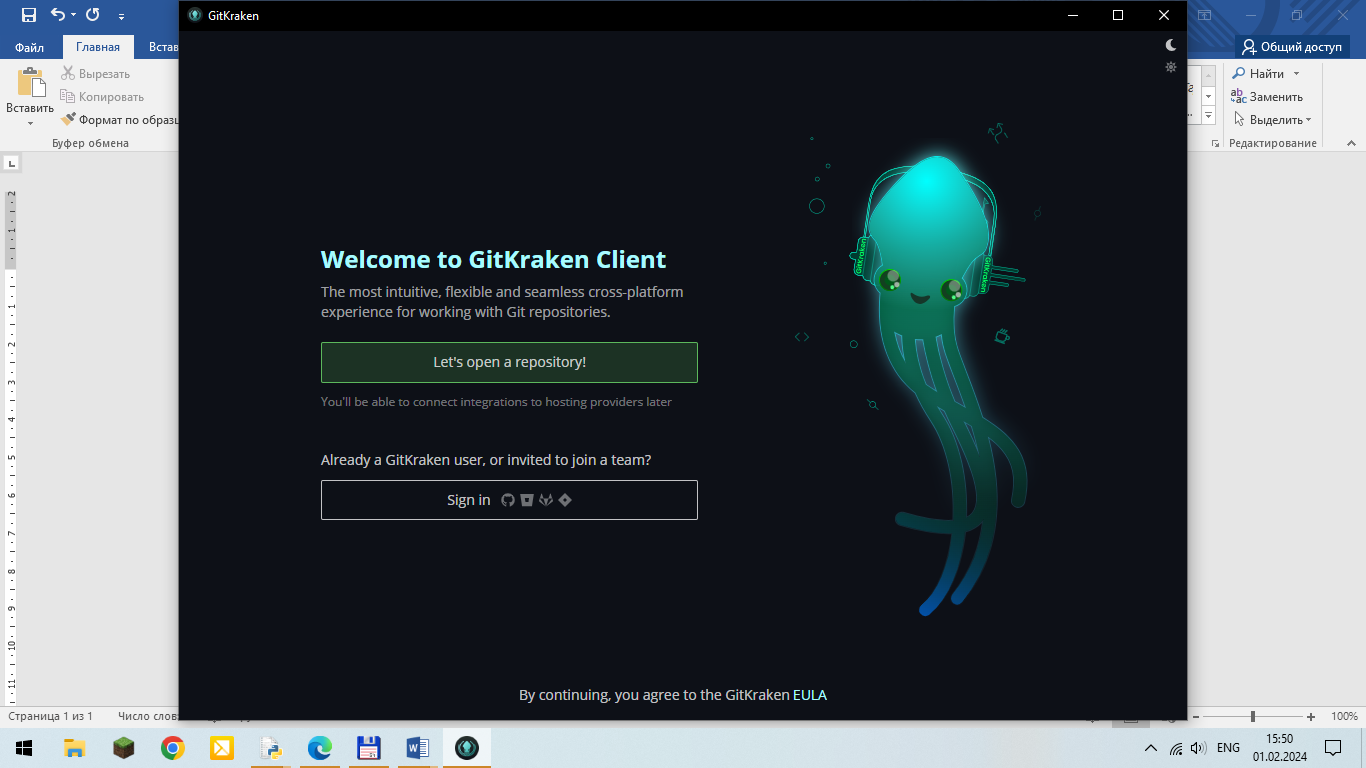


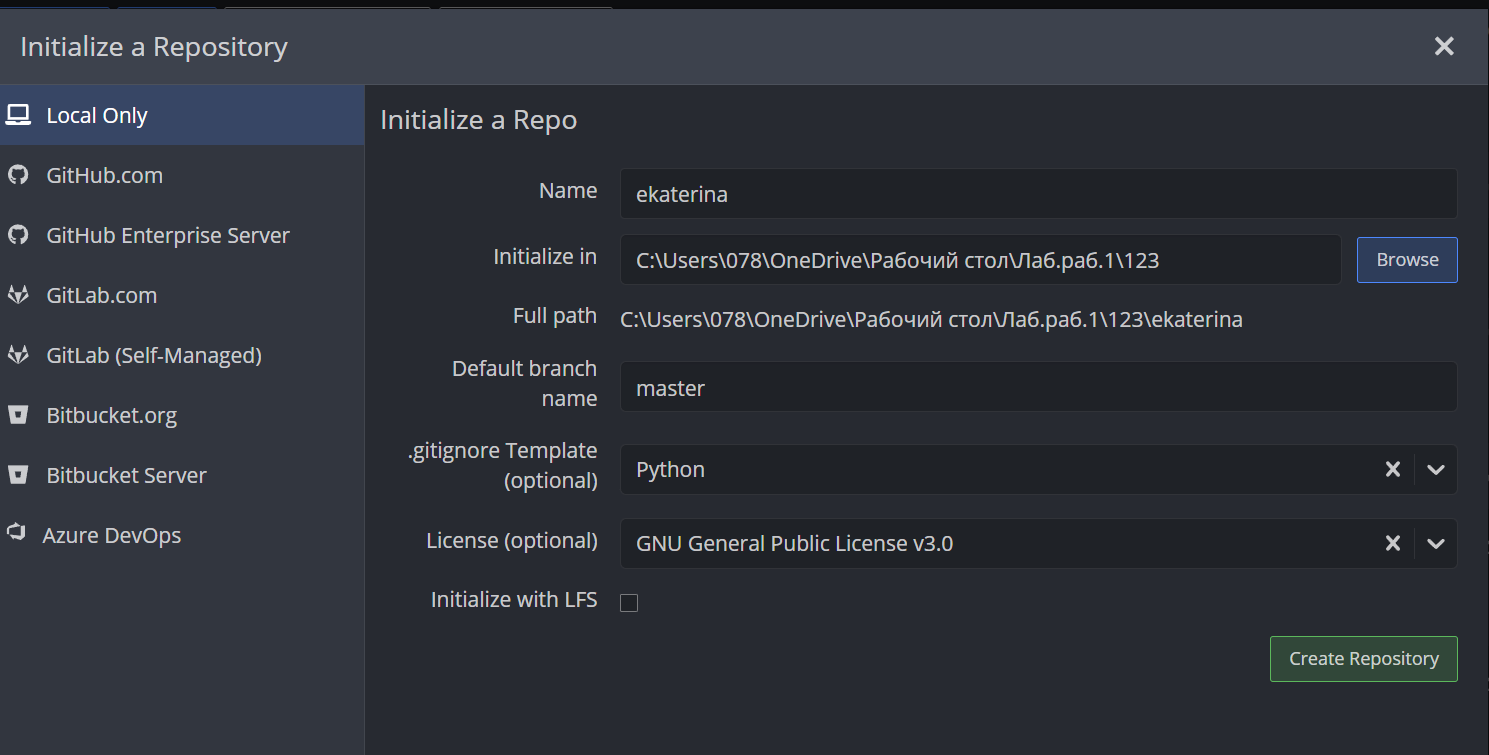
**Упражнение 2. Работа GUI git клиент GitKraken.**

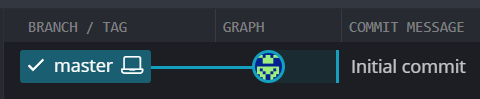
GitKraken – это кроссплатформенный, удобный и высокоэффективный GUI git клиент на Linux, Windows и macOS.



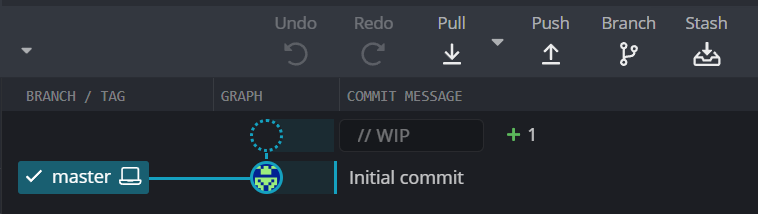
1. Инициализируйте в рабочей директории репозиторий.

****

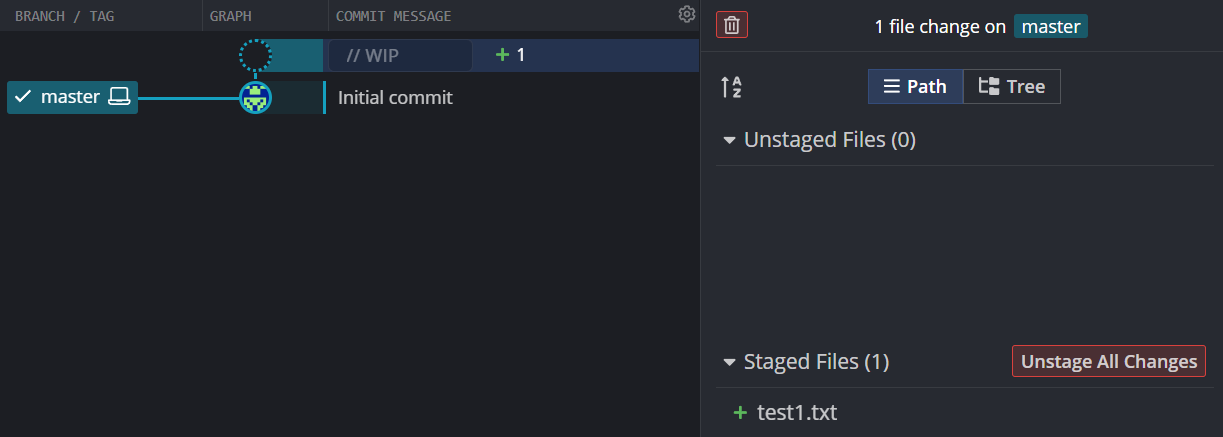
****

****

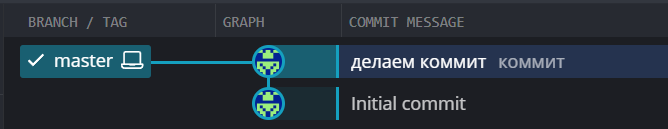
3. Создайте файл для исходного текста программы.



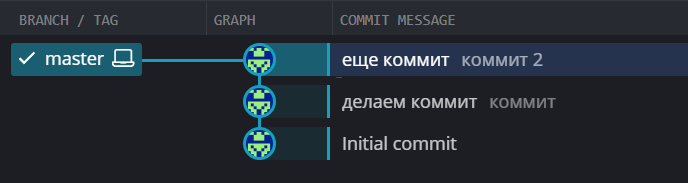
4. Добавьте созданный файл под версионный контроль.



5. Сделайте начальный коммит.

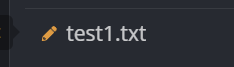


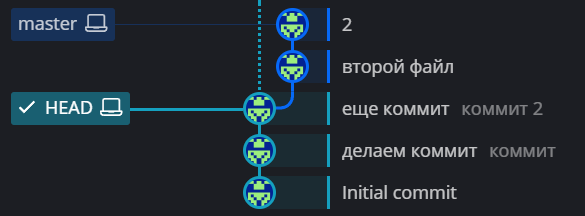
6. Сделайте еще несколько коммитов.



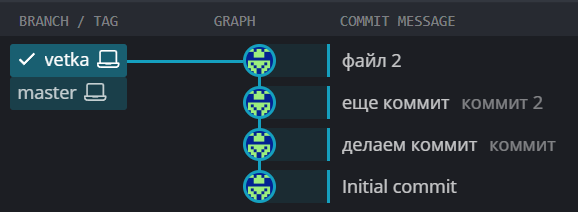
7. Добавьте еще несколько файлов с исходным текстом программы.

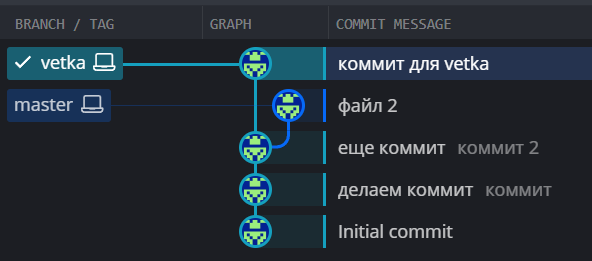


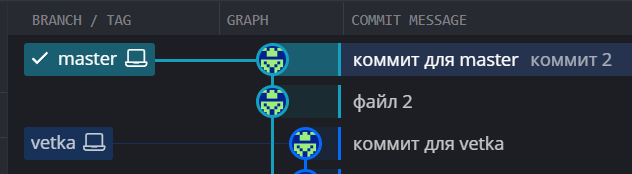


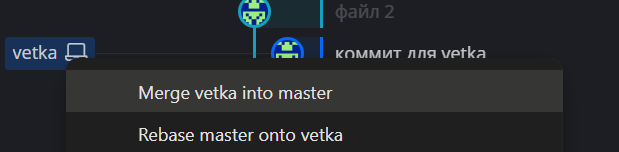


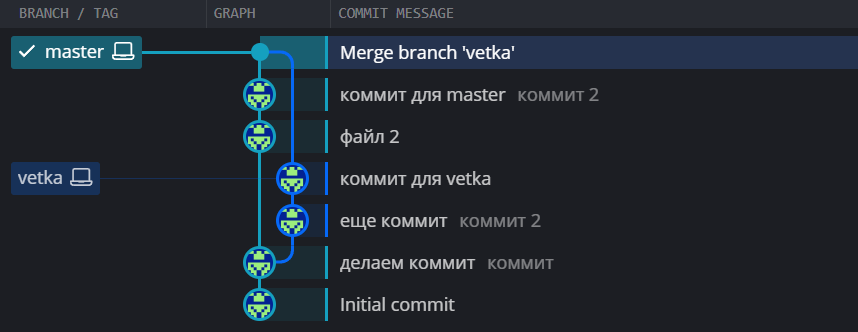
8. Создайте новую тематическую ветку.



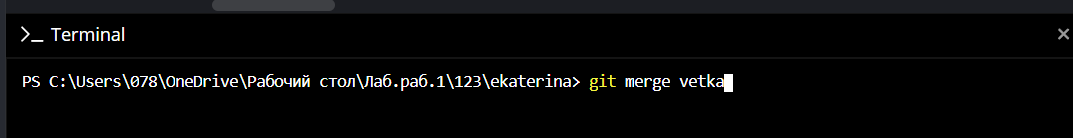
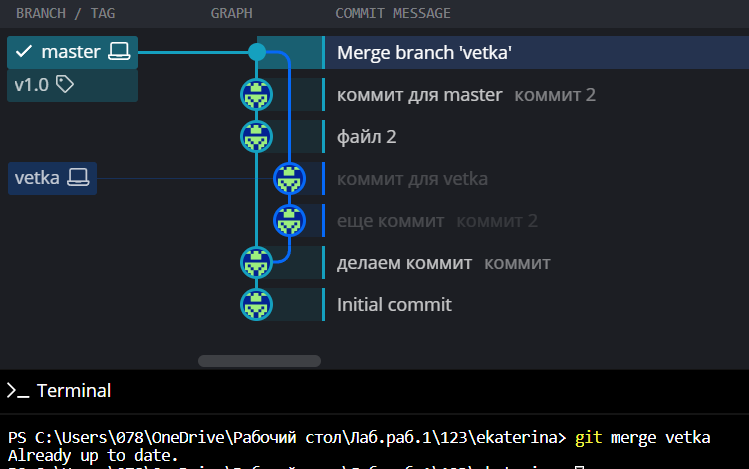
9. Сделайте несколько коммитов в основную и тематическую ветки. 



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



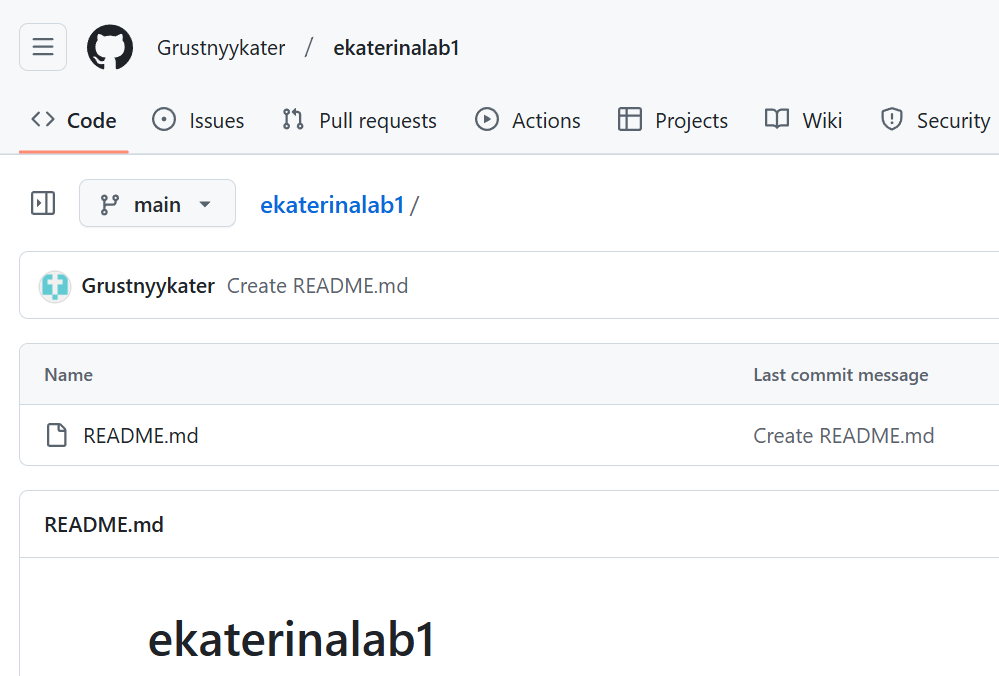
11. При получении в процессе разработки программы в стабильно работающем состоянии, слейте это состояние в основную ветку и добавьте к коммиту слияния пометку с номером релиза.

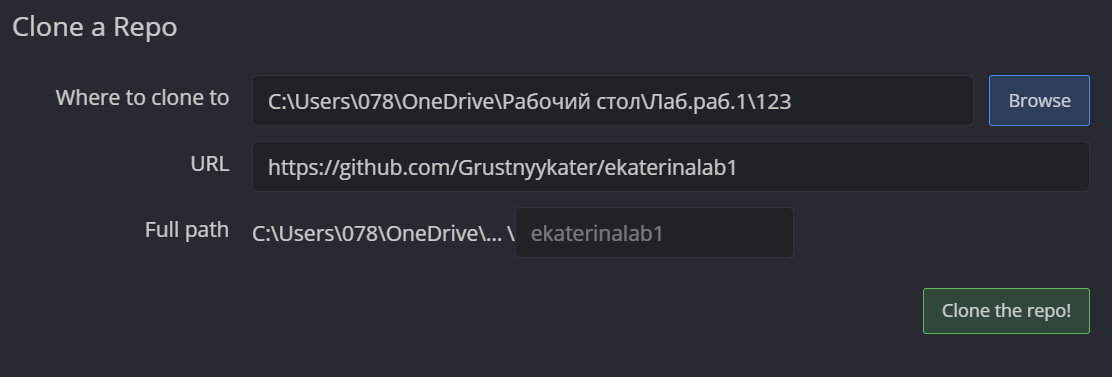
**Упражнение 3. Работа с удаленными репозиториями и GitHub.**

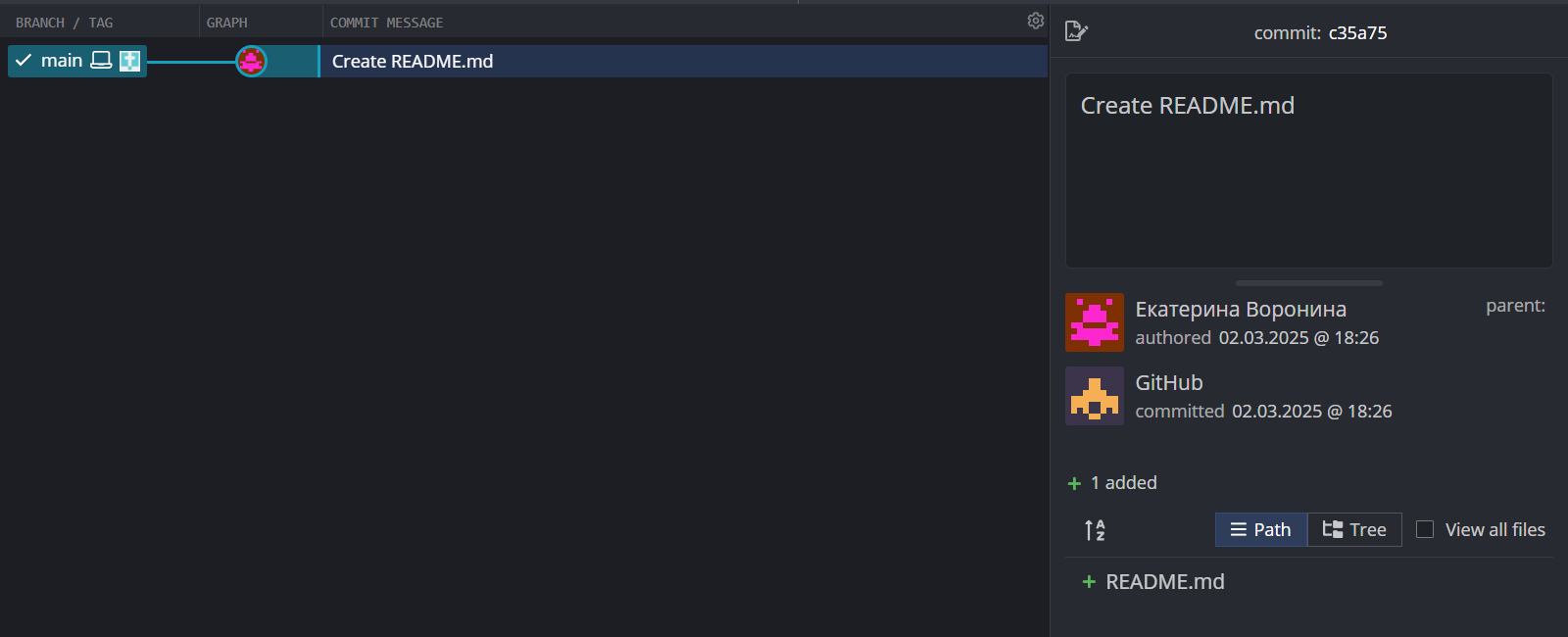
1. Зарегистрироваться на сайте github.com и создать новый пустой репозиторий.

2. Создать в этом репозитории файл Readme.

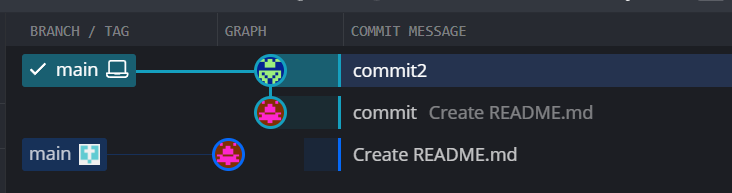


– Склонировать созданный удаленный репозиторий.





– Фиксируем скрипт в репозитории (делаем коммит).



– Обновляем удаленный репозиторий (делаем пуш).

