*Алиева Хадижат ПИ20-1*

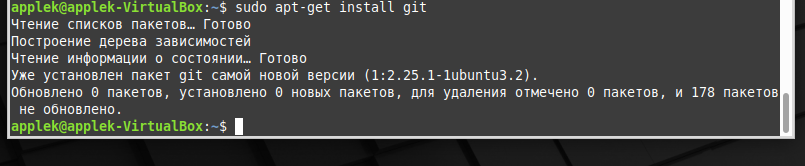
**1. Основы работы с СКВ в графическом режиме**

**Цель работы**

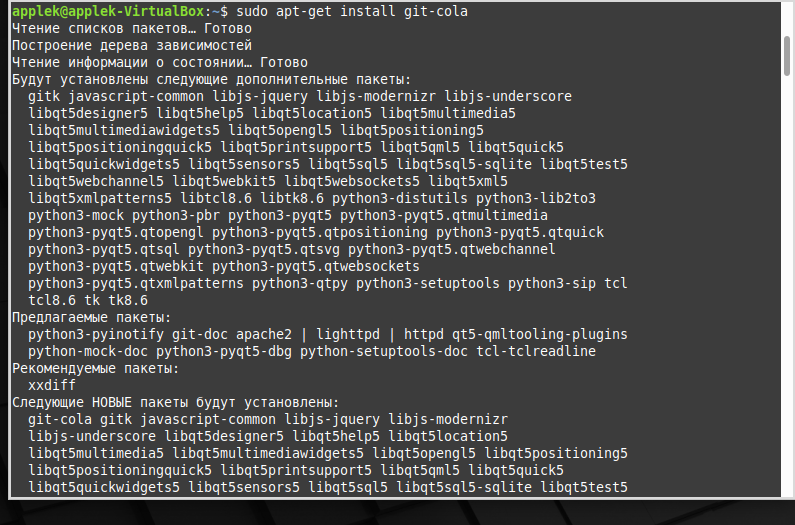
Познакомиться на практике с основными приемами работы в современных системах контроля версий.

**Задания для выполнения**

1. Установить на компьютер графический клиент Git.



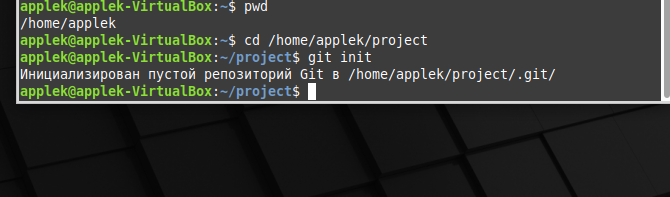
Так как git был уже ранее установлен, установим в терминале git-cola, что посоветовали в методических указаниях:



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



1. Выберите тематику программы, которую собираетесь написать. Язык программирования и используемые инструменты разработки сейчас не важны.
2. Инициализируйте в этой директории репозиторий гит.
3. 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 пазпок.

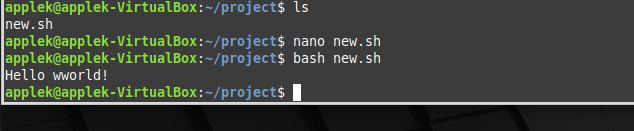


1. Создайте новый файл для исходного текста программы. Если вы используете программный фреймворк, инициализируйте его в рабочий каталог.



1. Напишите несколько строк вашей программы.

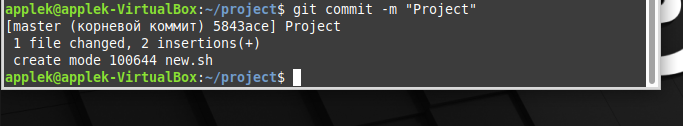




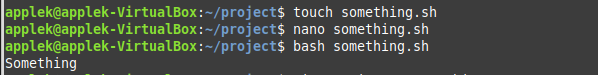
1. Добавьте файл с исходным текстом (несколько файлов, если необходимо) в индекс вашего репозитория.

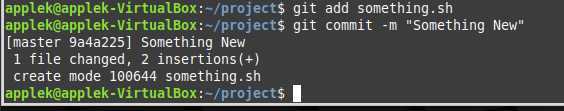


1. Совершите ваш первый коммит. Напишите осмысленное сообщение коммита.

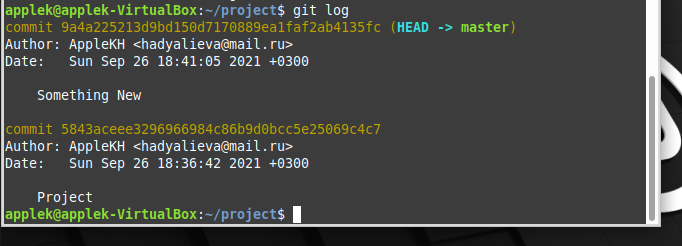


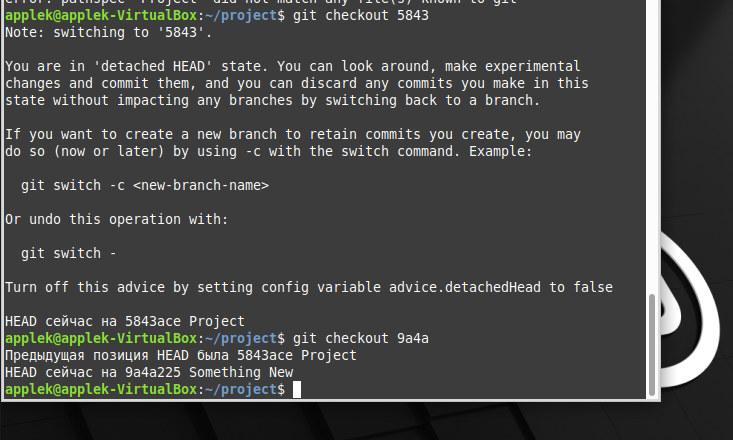
1. Повторите несколько раз. Каждый раз, завершая определенный этап работы, выполняйте коммит и описывайте проделанные изменения в сообщении коммита.





1. Просмотрите историю коммитов. Попробуйте перейти на один из прошлых коммитов. Вернитесь в актуальное состояние программы.





**Контрольные вопросы**

1. Опишите своими словами значение следующих терминов:
   1. **рабочий каталог**

Она представляет собой директорию (папку), которая может хранить файлы проекта и в которой может работать система контроля версий.

* 1. **репозиторий**

Это место хранения скв, которая хранит всю информацию о всех версиях состояний рабочего каталога, но при этом не входит в неё.

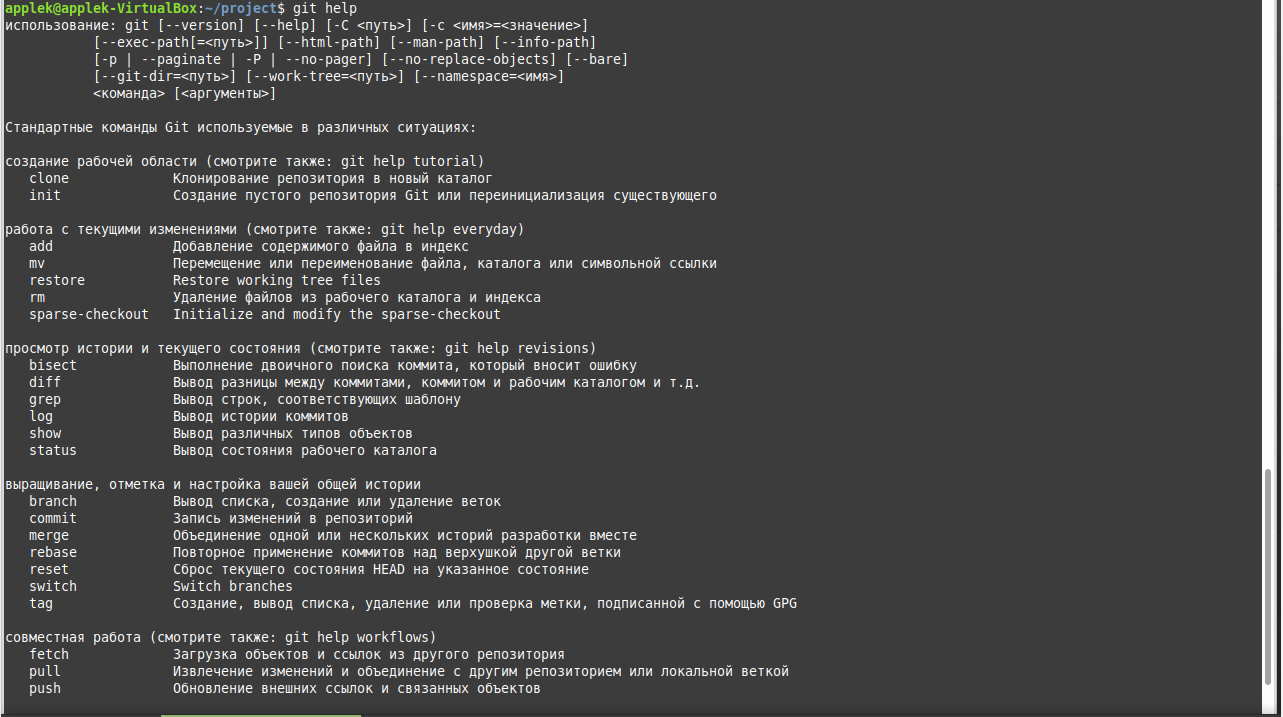
* 1. **коммит**

Оно представляет собой фиксирование изменений в новое состояние работы.

* 1. **ветка**

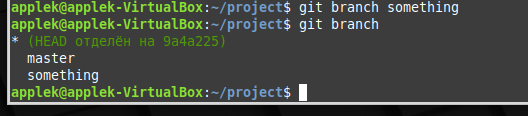
Это указатели на коммиты, которые могут перемещаться в зависимости появлением нового коммита.

1. Ознакомьтесь с гайдом по выбранной вами программе-клиенту Git.

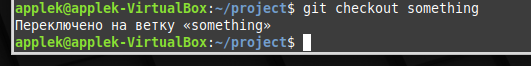


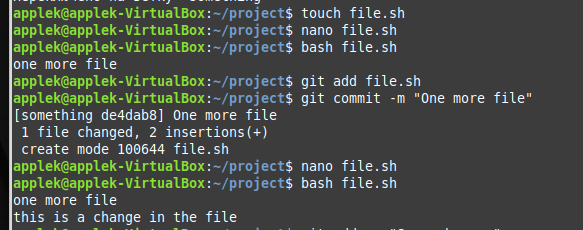
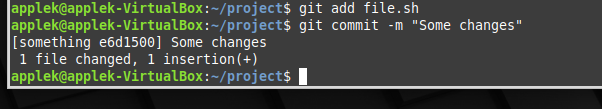
**Дополнительные задания**

1. Представьте, что вы начинаете большой раздел работы. Для изоляции изменений создайте новую ветку. Назовите ее, чтобы было понятно, что вы в ней будете делать.

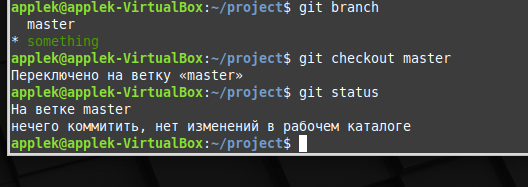


1. Перейдите в новую ветку и сделайте несколько коммитов.

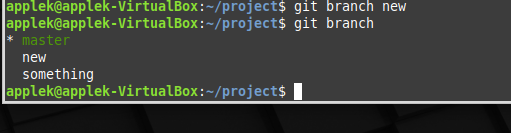


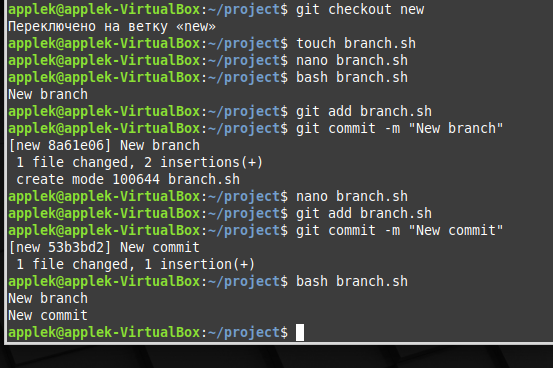
1. Перейдите в основную ветку и обратите внимание на состояние рабочей директории.



1. Создайте еще одну ветку для работы над другим направлением в вашей программе. Обычно так работают в команде, каждый участник в собственной ветке. Либо в ветках может идти параллельная работа над разными возможностями программы. В таком случае эти ветки называются тематическими.

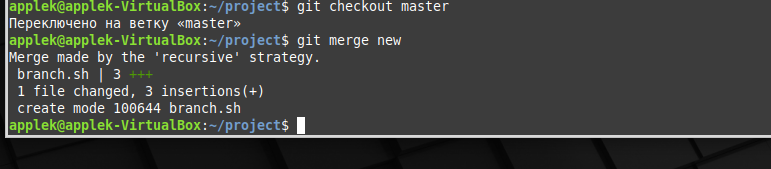


1. Сделайте несколько коммитов во вновь созданную ветку.

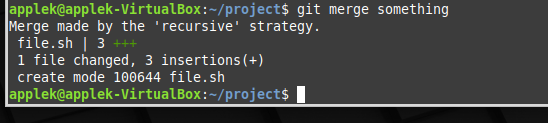


1. Перейдите в основную ветку и слейте в нее первую тематическую ветку.

(случайно добавила вторую ветку, а не первую, но в следующем задании добавила первую)



1. Слейте в основную ветку вторую тематическую. Если возникли конфликты слияния, разрешите их и завершите слияние.



1. Удалите более не нужные тематические ветки. Обратите внимание в истории, что даже при удалении веток никакие коммиты не теряются.

