Лабораторная работа 1

Елизавета Валерьевна Асеинова, НФИбд-01-19

Содержание

# **Цель работы**

Цель данной работы - научиться работать с системой управления версиями Git, познакомиться с его возможностями и закрепить полученные знания на практике.

# **Теоретическое введение**

Git - это бесплатная распределенная система контроля версий с открытым исходным кодом, предназначенная для быстрой и эффективной работы с любыми проектами - от небольших до очень крупных.

Git прост в освоении и обладает молниеносной производительностью. Его ключевые возможности - дешевое локальное ветвление, удобные области хранения и множество рабочих процессов.

Проект был создан Линусом Торвальдсом для управления разработкой ядра Linux, первая версия выпущена 7 апреля 2005 года.

# **Выполнение лабораторной работы**

***1.1 Подготовка***

1.1.1 Установка имени и электронной почты (риc.1)

Выполняем:   
git config --global user.name "Your Name"  
git config --global user.email "your\_email@whatever.com"

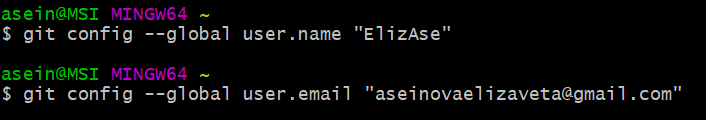


Figure 1: Установка имени и электронной почты

1.1.2 Параметры установки окончаний строк (риc.2)

Выполняем:  
git config --global core.autocrlf true  
git config --global core.safecrlf true

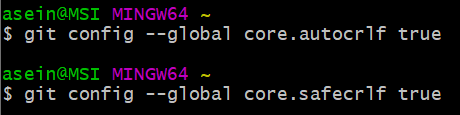


Figure 2: Параметры установки окончаний строк

1.1.3 Установка отображения unicode (риc.3)

Выполняем:  
git config --global core.quotepath off

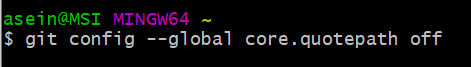


Figure 3: Установка отображения unicode

***1.2 Создание проекта***

1.2.1 Создайте страницу «Hello, World» (риc.4)

Начинаем работу в пустом рабочем каталоге с создания пустого каталога с именем  
hello, затем входим в него и создаем там файл с именем hello.html.

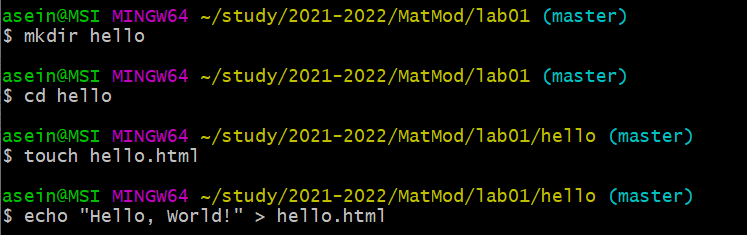


Figure 4: Создание страницы «Hello, World»

1.2.2 Создание репозитория (риc.5)

Чтобы создать git репозиторий из этого каталога, выполняем команду git init

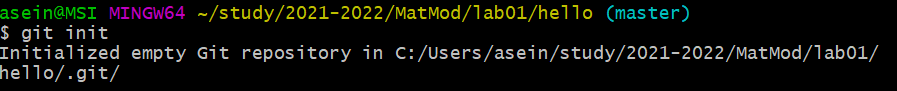


Figure 5: Создание репозитория

1.2.3 Добавление файла в репозиторий (риc.6)

Добавляем файл в репозиторий

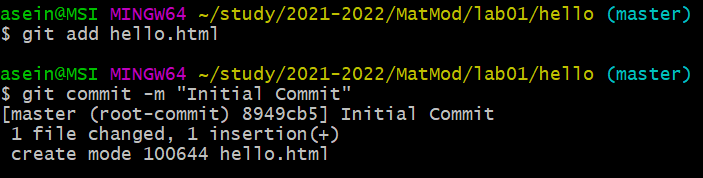


Figure 6: Добавление файла в репозиторий

1.2.4 Проверка состояния репозитория (риc.7)

Используем команду git status, чтобы проверить текущее состояние репозитория.

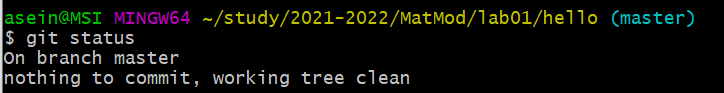


Figure 7: Проверка состояния репозитория

***1.3 Внесение изменений***

1.3.1 Измените страницу «Hello, World» (риc.8)

Добавим HTML-теги к нашему приветствию и проверяем состояние рабочего каталога.

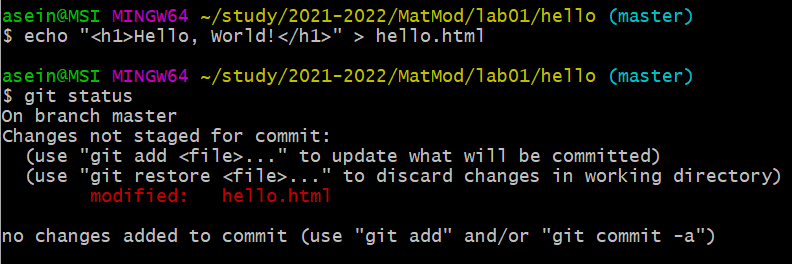


Figure 8: Изменение страницы «Hello, World»

***1.4 Индексация изменений*** (риc.9)

Теперь выполняем команду git, чтобы проиндексировать изменения. Проверяем состояние.

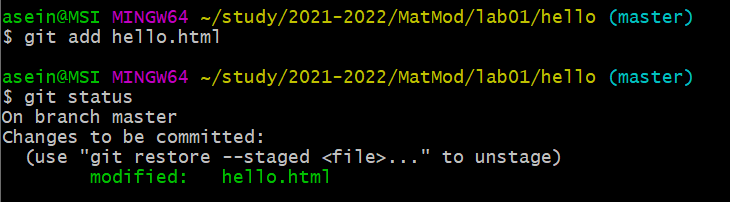


Figure 9: Индексация изменений

1.4.1 Коммит изменений (риc.10)

Делаем коммит и проверяем состояние. Открывается редактор. В первой строке вводим комментарий: «Added h1 tag». Сохраняем файл и выходим из редактора.  
Теперь еще раз проверим состояние

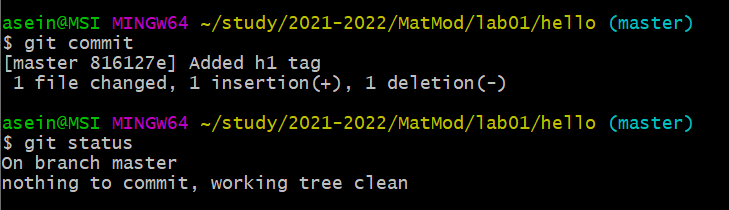
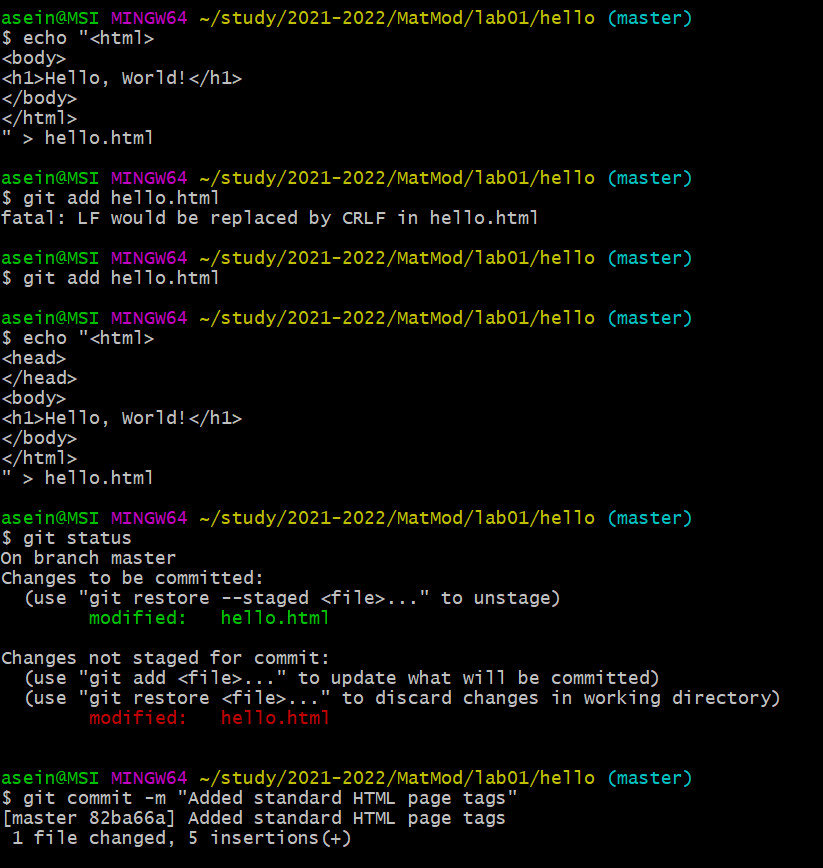
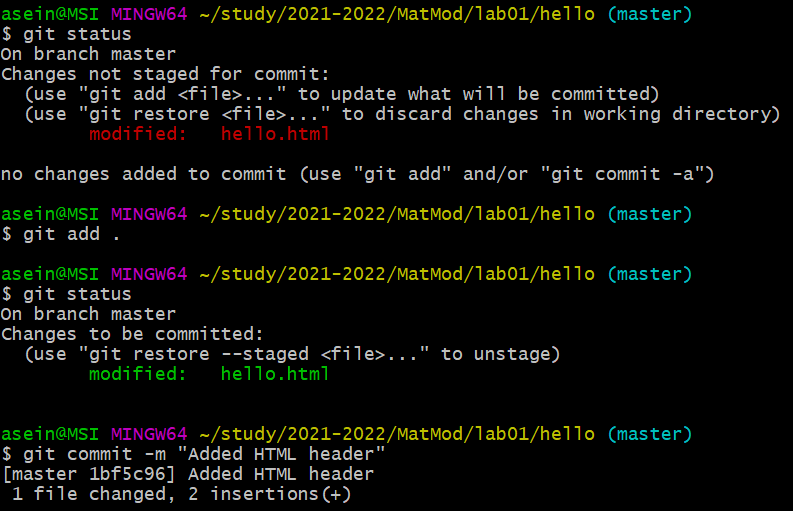


Figure 10: Коммит изменений

1.4.2 Добавьте стандартные теги страницы (риc.**¿fig:011?**)

Изменим страницу «Hello, World», чтобы она содержала стандартные теги <html> и <body>. Затем добавим это изменение в индекс git. Теперь добавим заголовки HTML (секцию <head>) к странице «Hello, World» и проверим текущий статус. Произведем коммит, а затем ещё раз проверим статус. Затем добавим второе изменение в индекс и снова проверим статус. В конце сделаем коммит второго изменения.

 { #fig:011 width=70% }



Добавление стандартных тегов 2

1.4.3 История (риc.[-@fig:012])  
Получим список произведенных изменений.

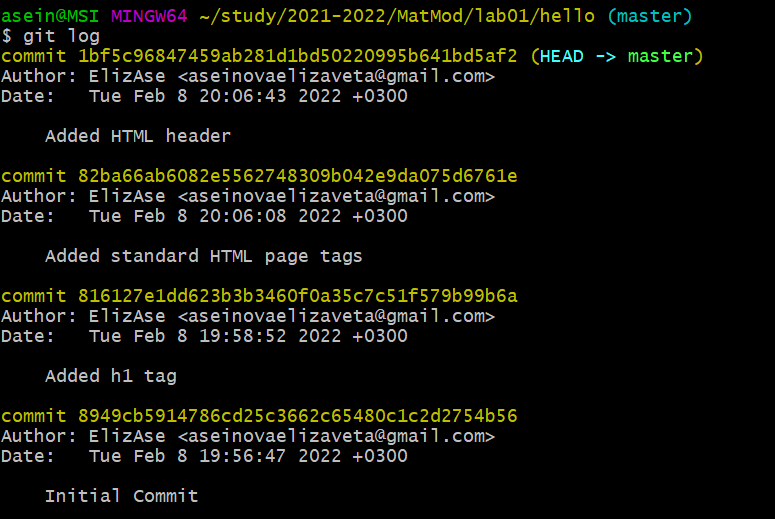


Figure 11: История

1.4.4 Получение старых версий (риc.12)

Изучим данные лога и найдем хэш для первого коммита.Затем проверим содержимое файла hello.html. Вернемся к последней версии в ветке master.

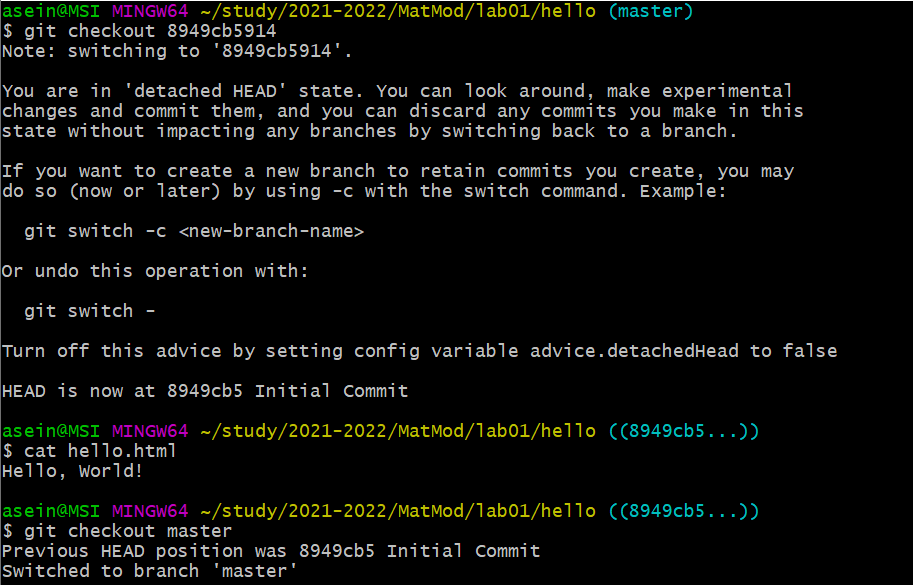


Figure 12: Получение старых версий

1.4.5 Создание тегов версий (риc.13)

Назовем текущую версию страницы hello первой (v1).  
Создадим тег первой версии.Создадим тег для версии, которая идет перед текущей версией и назовем его v1-beta.

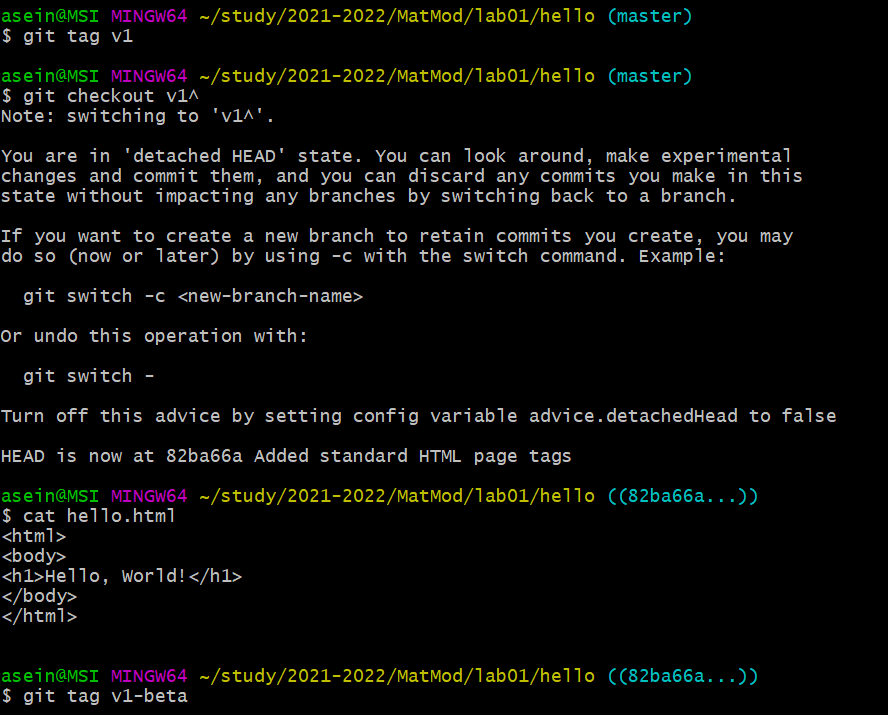


Figure 13: Создание тегов версий

1.4.6 Переключение по имени тега (риc.14)

Теперь пробуем попереключаться между двумя отмеченными версиями.

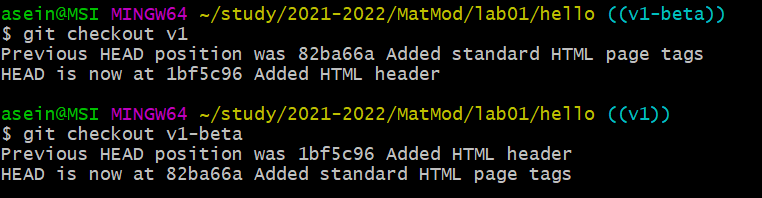


Figure 14: Переключение по имени тега

1.4.7 Просмотр тегов с помощью команды tag (риc.15)

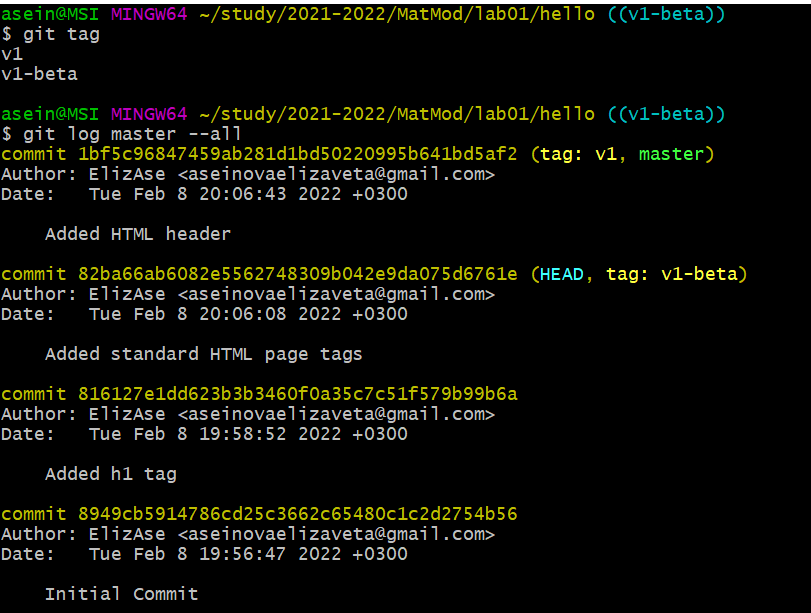


Figure 15: Просмотр тегов с помощью команды tag

***1.5 Отмена локальных изменений (до индексации)***

1.5.1 Переключитесь на ветку master (риc.16)

Убедимся, что находимся на последнем коммите ветки master, прежде чем продолжить работу.

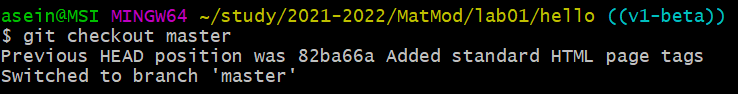


Figure 16: Переключение на ветку master

1.5.2 Измените hello.html (риc.17)

Вносим изменение в файл hello.html в виде нежелательного комментария.

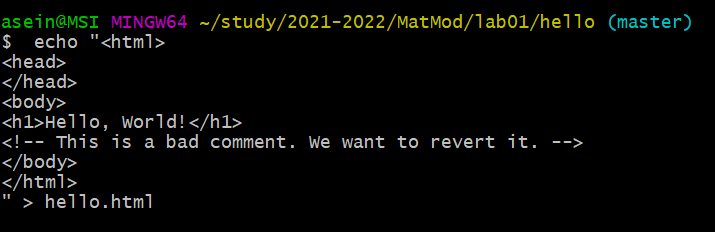


Figure 17: Изменение hello.html

1.5.3 Проверьте состояние (риc.18)

Проверим состояние рабочего каталога.

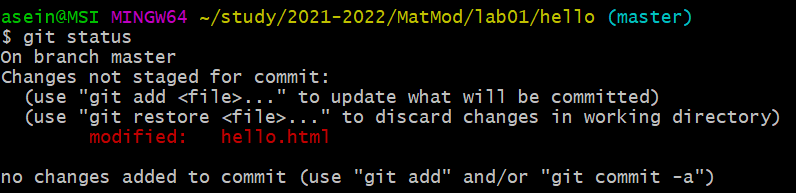


Figure 18: Проверка состояния

1.5.4 Отмена изменений в рабочем каталоге (риc.19)

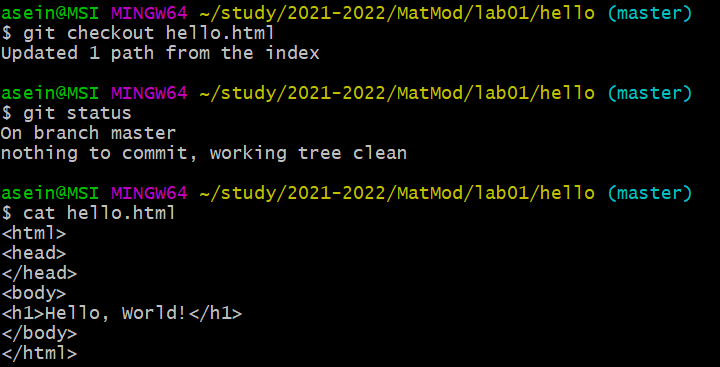


Figure 19: Отмена изменений в рабочем каталоге

***1.6 Отмена проиндексированных изменений (перед коммитом)***

1.6.1 Измените файл и проиндексируйте изменения(риc.20)

Внесем изменение в файл hello.html в виде нежелательного комментария

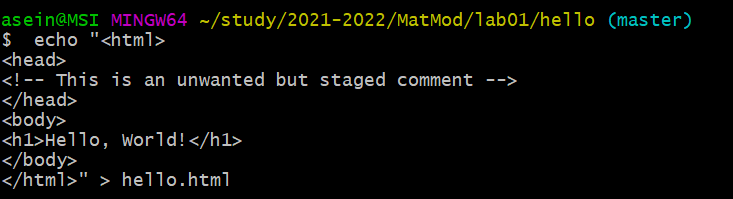


Figure 20: Изменение файла и индексация изменения

1.6.2 Проверьте состояние (риc.21)

Проверим состояние нежелательного изменения.

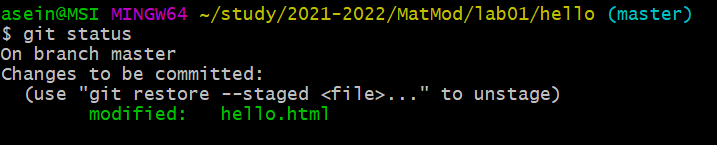


Figure 21: Проверка состояния

1.6.3 Выполните сброс буферной зоны (риc.22)

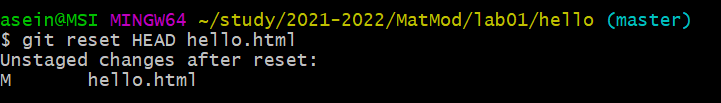


Figure 22: Сброс буферной зоны

1.6.4 Переключитесь на версию коммита (риc.23)

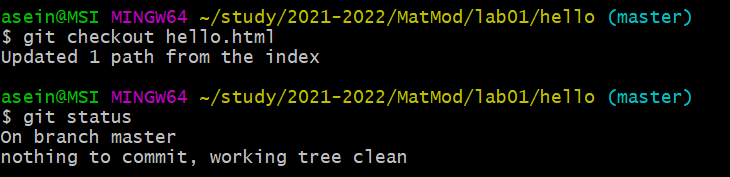


Figure 23: Переключение на версию коммита

***1.7 Отмена коммитов***

1.7.2 Измените файл и сделайте коммит (риc.24)

Меняем файл hello.html

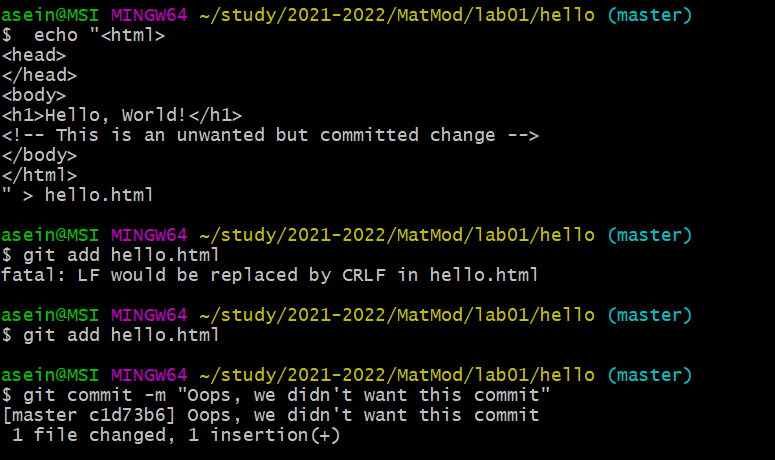


Figure 24: Смена файла и коммит

1.7.3 Сделайте коммит с новыми изменениями, отменяющими предыдущие (риc.25)

Чтобы отменить коммит, нам необходимо сделать коммит, который удаляет изменения, сохраненные нежелательным коммитом.

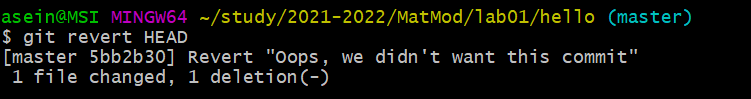


Figure 25: Коммит с новыми изменениями

1.7.4 Проверьте лог (риc.26)

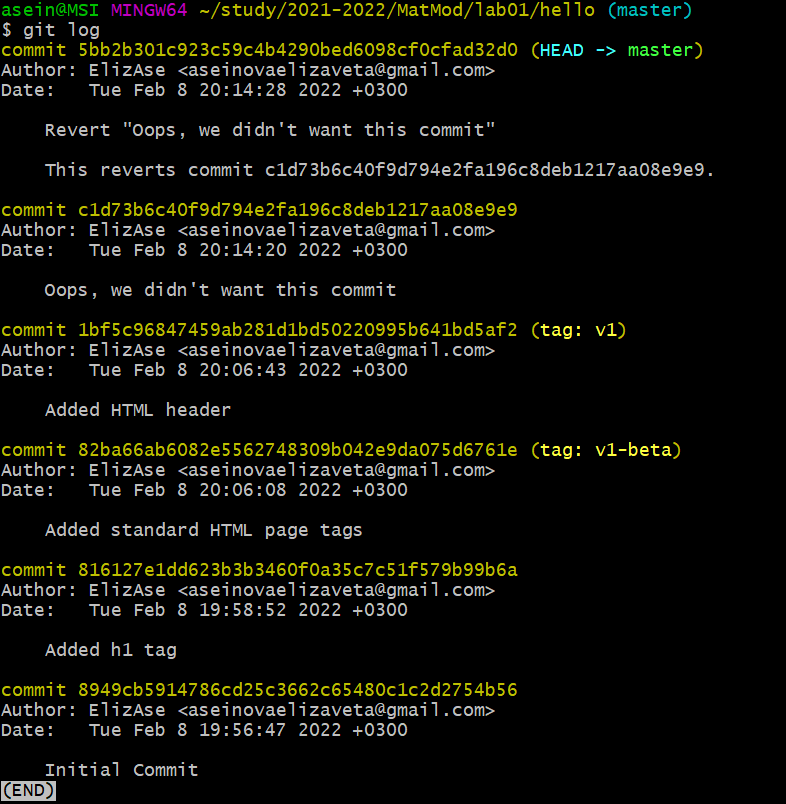


Figure 26: Проверка лога

***1.8 Удаление коммиттов из ветки***

1.8.1 Команда git reset

1.8.2 Проверьте нашу историю

Сделаем быструю проверку нашей истории коммитов.

1.8.3 Для начала отметьте эту ветку (риc.27)

Прежде чем удалить коммиты, отметим последний коммит тегом, чтобы  
потом можно было его найти.

Figure 27: Отмечаем ветку

Figure 27: Отмечаем ветку

1.8.4 Сброс коммитов к предшествующим коммиту Oops (риc.28)

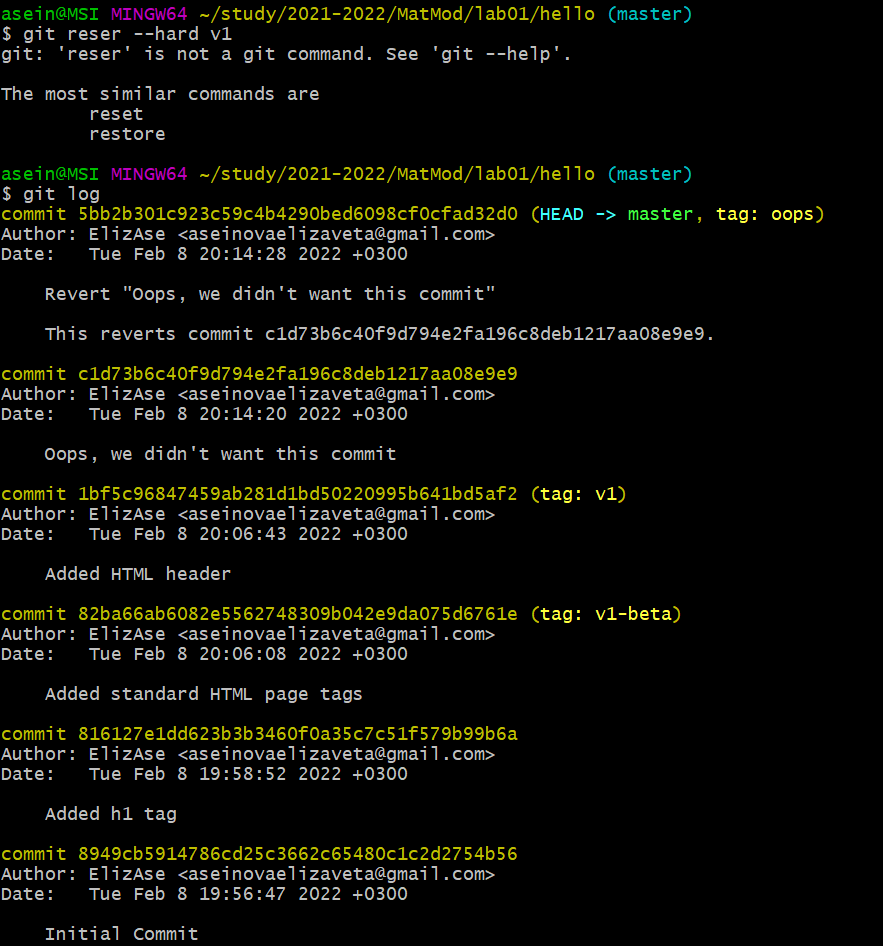


Figure 28: Сброс коммитов

1.8.5 Ничего никогда не теряется (риc.29)

Посмотрим на все коммиты.  
Мы видим, что ошибочные коммиты не исчезли. Они все еще находятся в репозитории. Просто они отсутствуют в ветке master

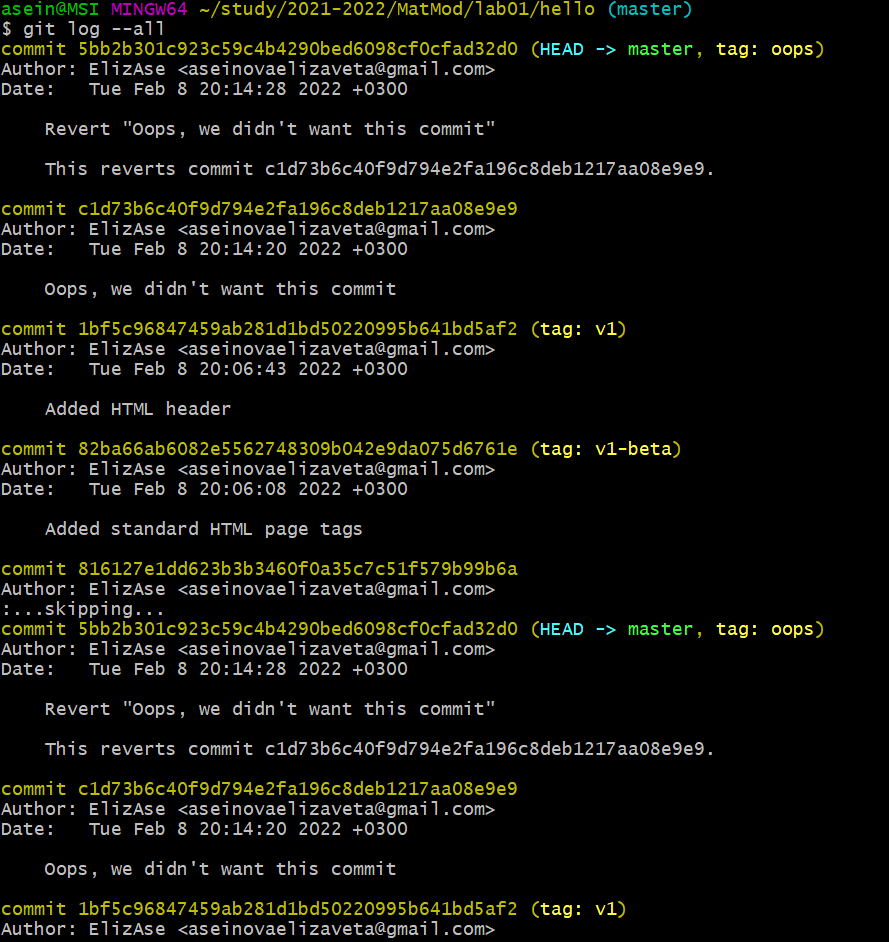


Figure 29: Просмотр коммитов

***1.9 Удаление тега oops***

1.9.1 Удаление тега oops (риc.30)

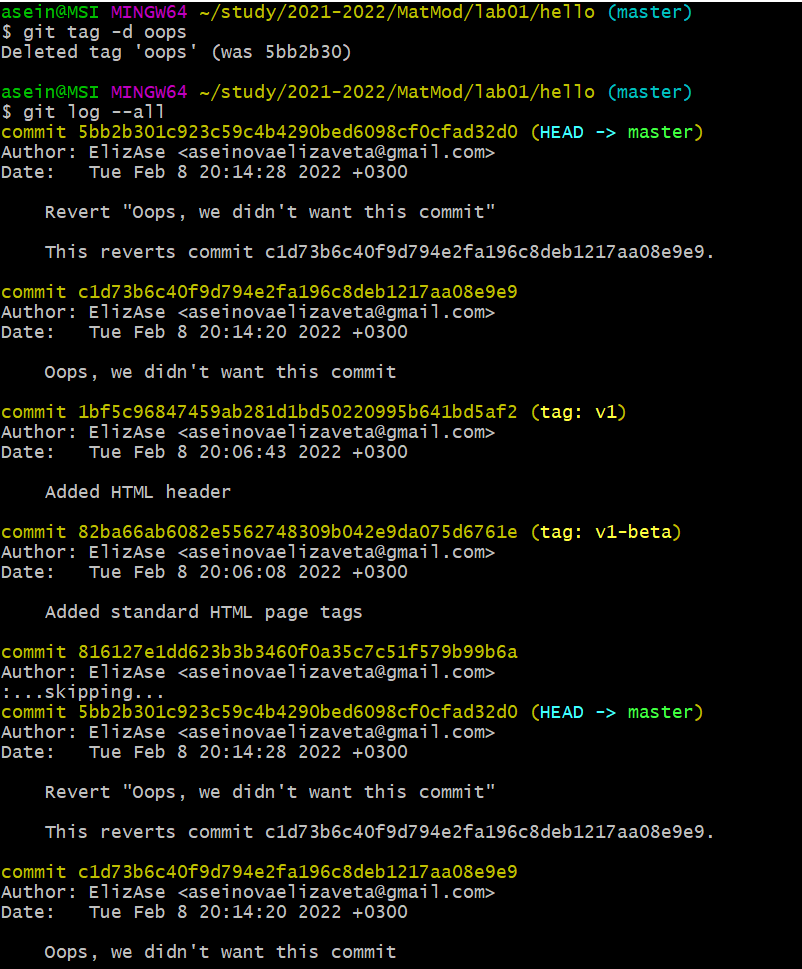


Figure 30: Удаление тега oops

***1.10 Внесение изменений в коммиты***

1.10.1 Измените страницу, а затем сделайте коммит (риc.31)

Добавим в страницу комментарий автора

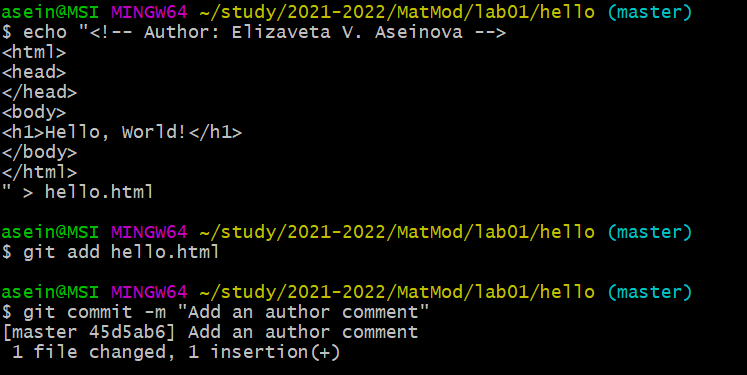


Figure 31: Изменение и коммит

1.10.2 Необходим email (риc.32)

Обновим страницу hello, включив в  
нее email.

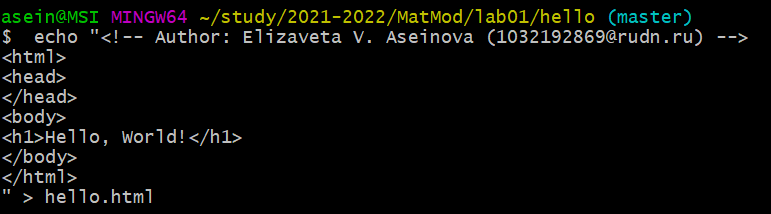


Figure 32: Добавление почты

1.10.3 Измените предыдущий коммит (риc.33)

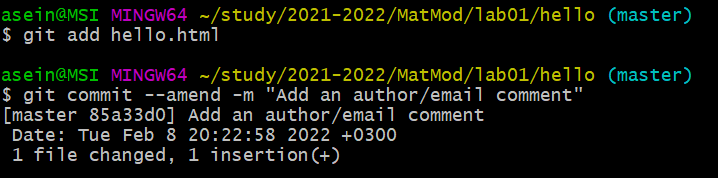


Figure 33: Изменение предыдущего коммита

1.10.4 Просмотр истории (риc.34)

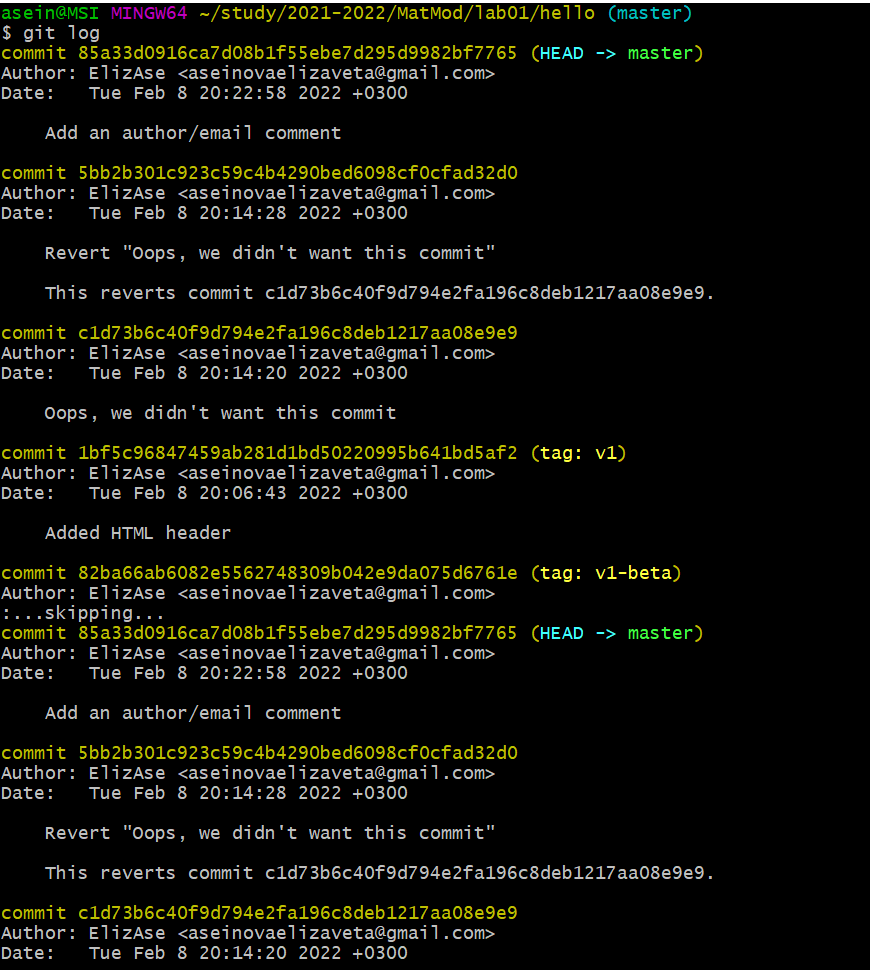


Figure 34: Просмотр истории

***1.11 Перемещение файлов***

1.11.1 Переместите файл hello.html в каталог lib (риc.35)

Перенесем страницу в каталог lib.

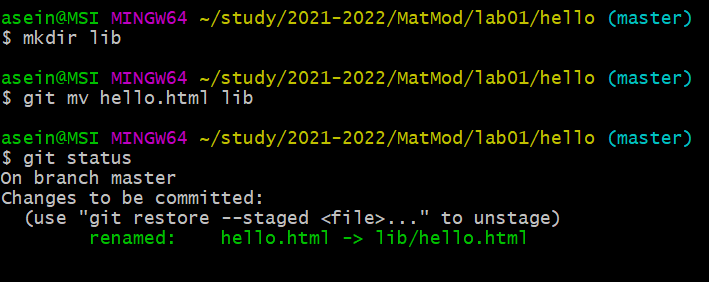


Figure 35: Просмотр истории

***1.12 Второй способ перемещения файлов***

1.12.1 Коммит в новый каталог (риc.36)

Сделаем коммит перемещения

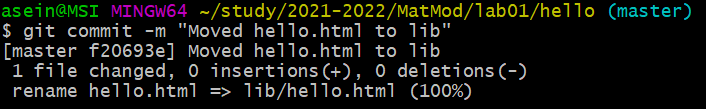


Figure 36: Коммит в новый каталог

***1.13 Подробнее о структуре***

1.13.1 Добавление index.html (риc.37)

Добавим файл index.html в наш репозиторий

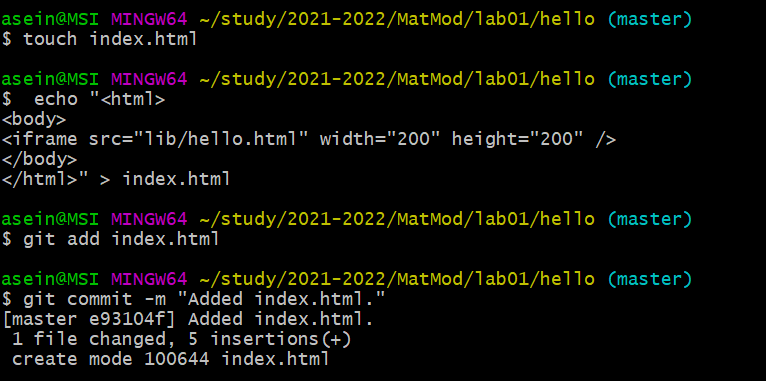


Figure 37: Добавление index.html

***1.14 Git внутри: Каталог .git***

1.14.1 Каталог .git (риc.38)

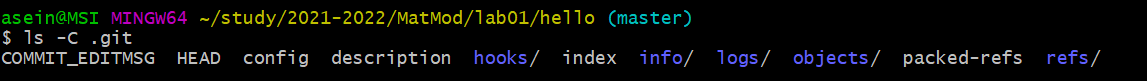


Figure 38: Каталог .git

1.14.2 База данных объектов (риc.39)

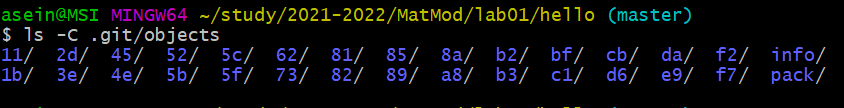


Figure 39: База данных объектов

1.14.3 Углубляемся в базу данных объектов (риc.40)

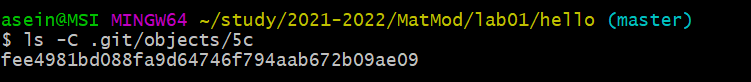


Figure 40: Подробнее о базе данных

1.14.4 Config File (риc.41)

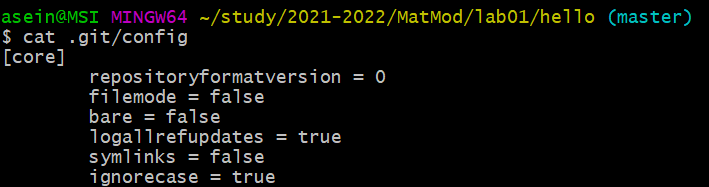


Figure 41: Config File

1.14.5 Ветки и теги (риc.42)

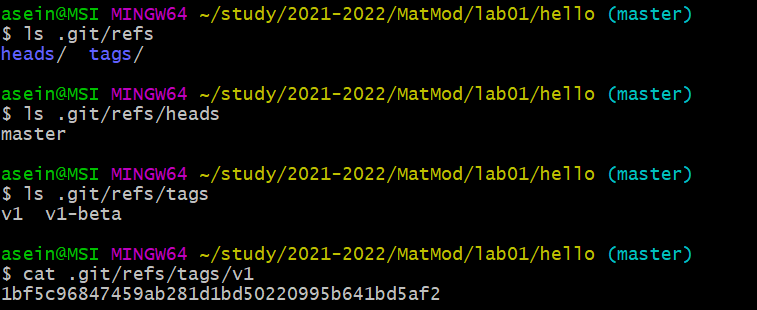


Figure 42: Ветки и теги

1.14.6 Файл HEAD (риc.43)

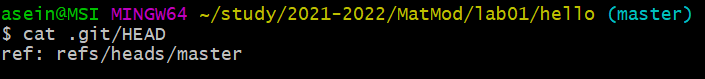


Figure 43: Файл HEAD

***1.15 Работа непосредственно с объектами git***

1.15.1 Поиск последнего коммита (риc.44)

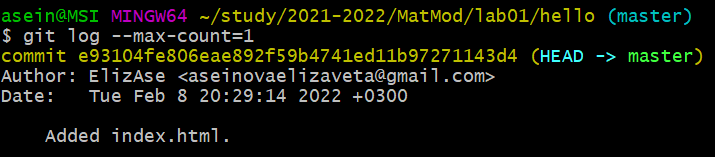


Figure 44: Поиск последнего коммита

1.15.2 Вывод последнего коммита с помощью SHA1 хэша (риc.45)

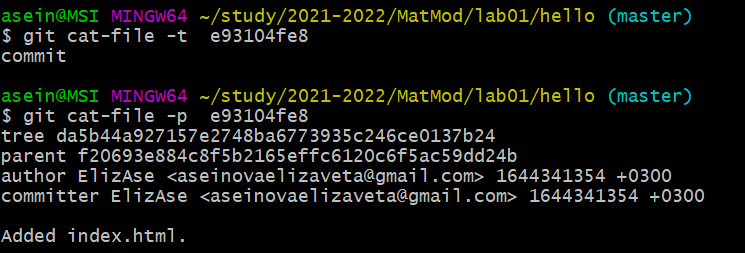


Figure 45: Вывод последнего коммита

1.15.3 Поиск дерева (риc.46) Мы выведем дерево каталогов, ссылка на который идет в коммите.

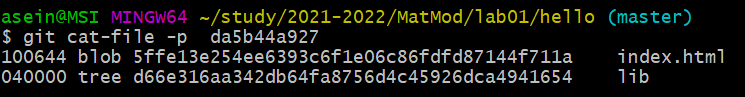


Figure 46: Поиск дерева

1.15.4 Вывод каталога lib (риc.47)

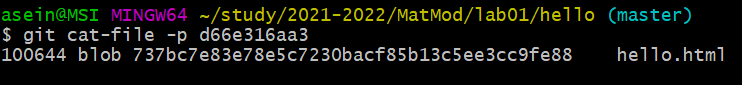


Figure 47: Вывод каталога lib

1.15.5 Вывод файла hello.html (риc.48)

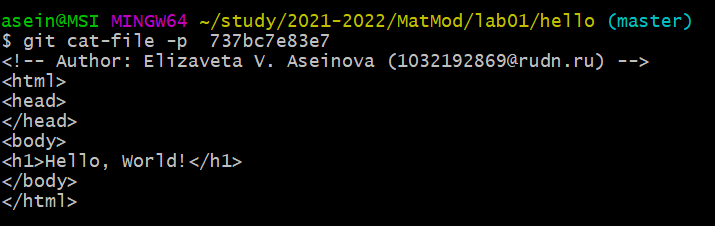


Figure 48: Вывод файла hello.html

***1.16 Создание ветки***

1.16.1 Создайте ветку (риc.49)

Назовем нашу новую ветку «style».

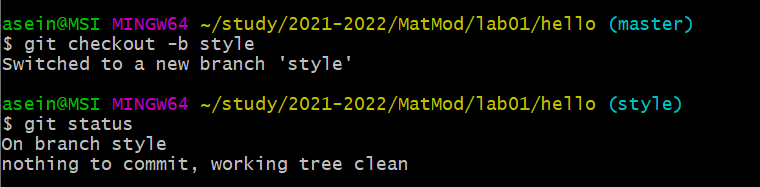


Figure 49: Вывод файла hello.html

1.16.2 Добавьте файл стилей style.css (риc.50)



Figure 50: Добавление файла стилей style.css

1.16.3 Измените основную страницу (риc.51)

Обновим файл hello.html, чтобы использовать стили style.css.

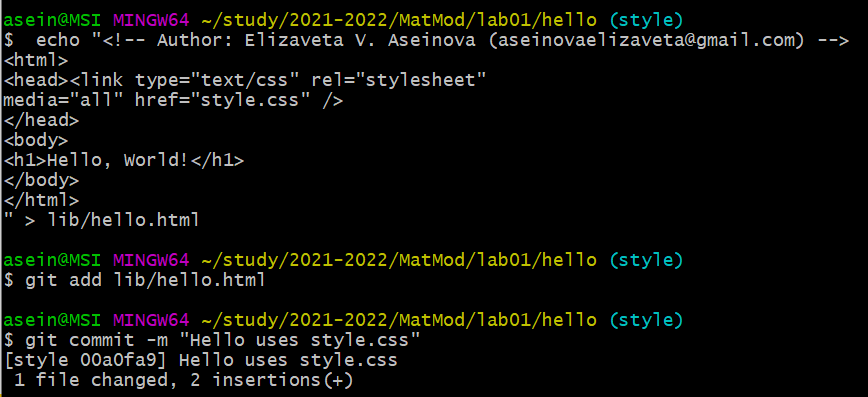


Figure 51: Изменение основной страницы

1.16.4 Измените index.html (риc.52)

Обновим файл index.html, чтобы он тоже использовал style.css

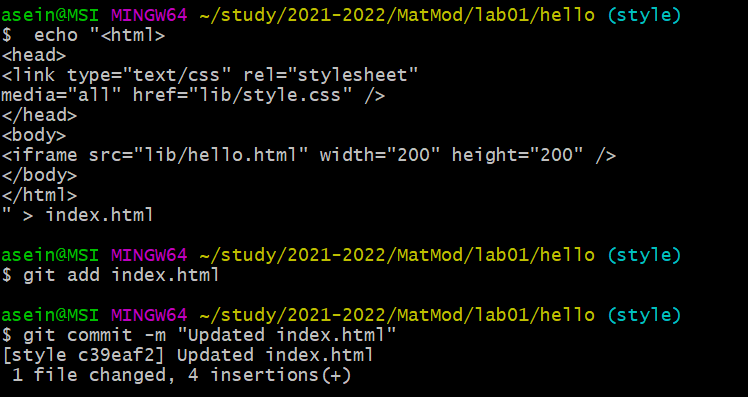


Figure 52: Изменение основной страницы

***1.17 Навигация по веткам*** (риc.53)

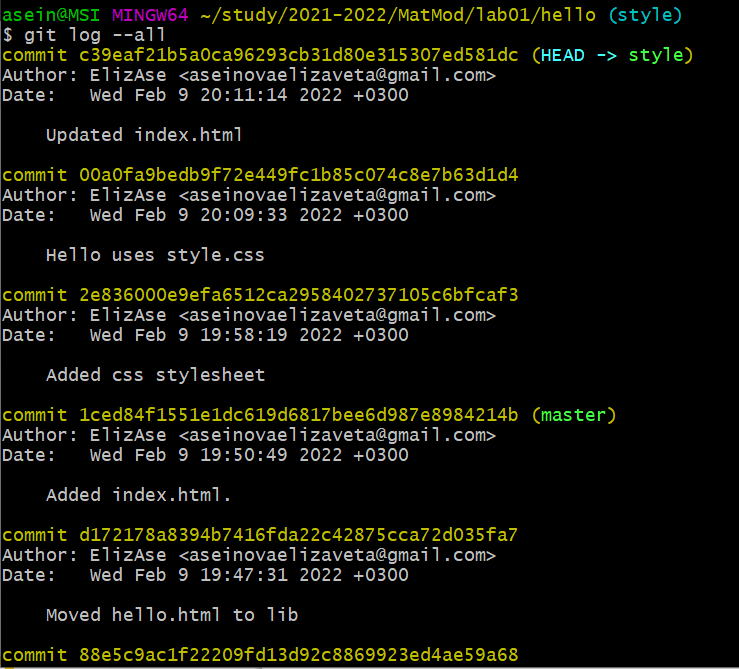


Figure 53: Ветки

1.17.1 Переключение на ветку master (риc.54)

Используем команду git checkout для переключения между ветками

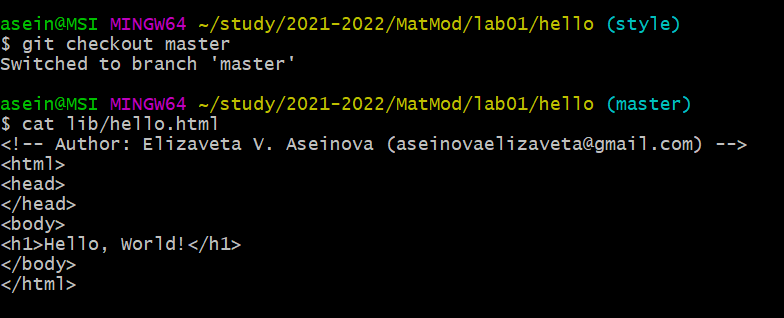


Figure 54: Переключение на ветку master

1.17.2 Вернемся к ветке style (риc.55)

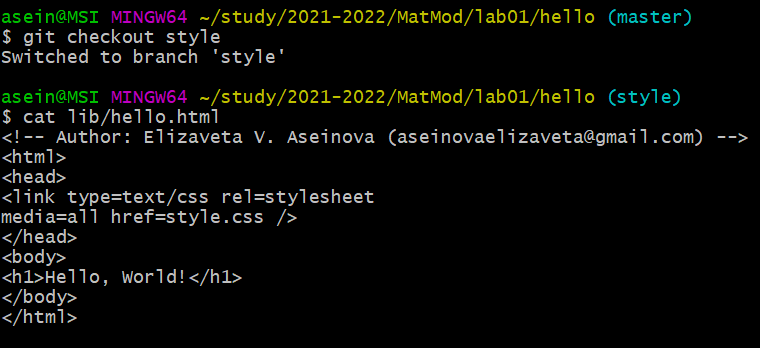


Figure 55: Ветка style

***1.18 Изменения в ветке master***

1.18.1 Создайте файл README в ветке master (риc.56)

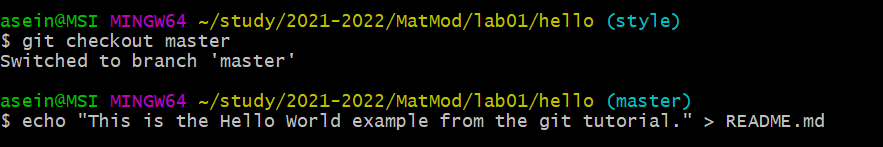


Figure 56: Файл README

***1.19 Сделайте коммит изменений README.md в ветку master*** (риc.57)

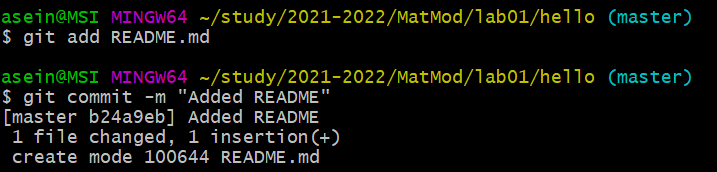


Figure 57: Коммит изменений README.md

1.19.2 Просмотрите текущие ветки (риc.58)

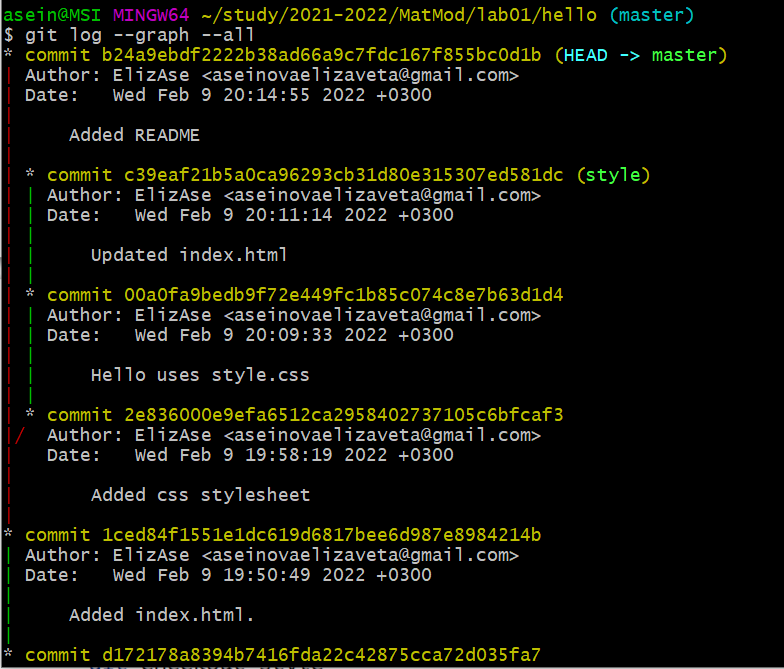


Figure 58: Коммит изменений README.md

***1.20 Слияние***

1.20.1 Слияние веток (риc.59)

Слияние переносит изменения из двух веток в одну. Вернемся к ветке style и сольем master с style.

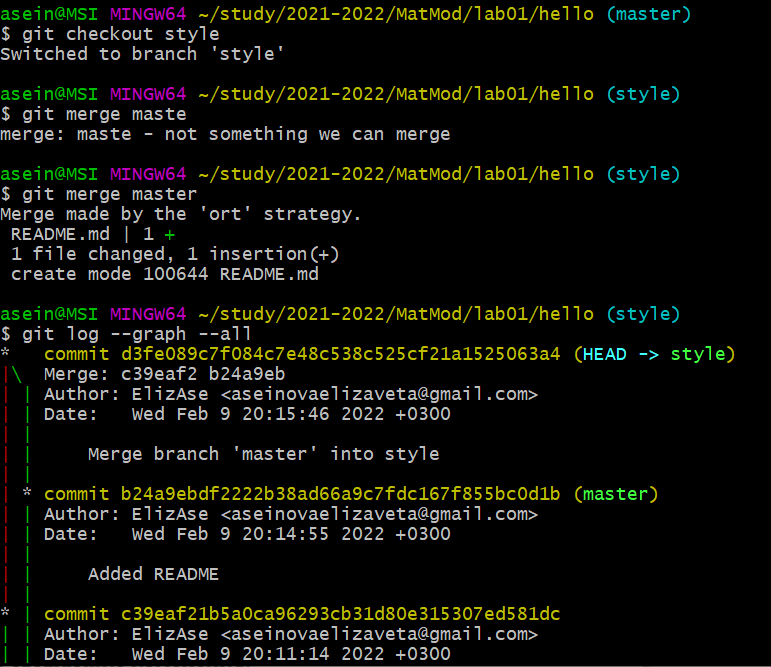


Figure 59: Слияние веток

***1.21 Создание конфликта***

1.21.1 Вернитесь в master и создайте конфликт (риc.60)

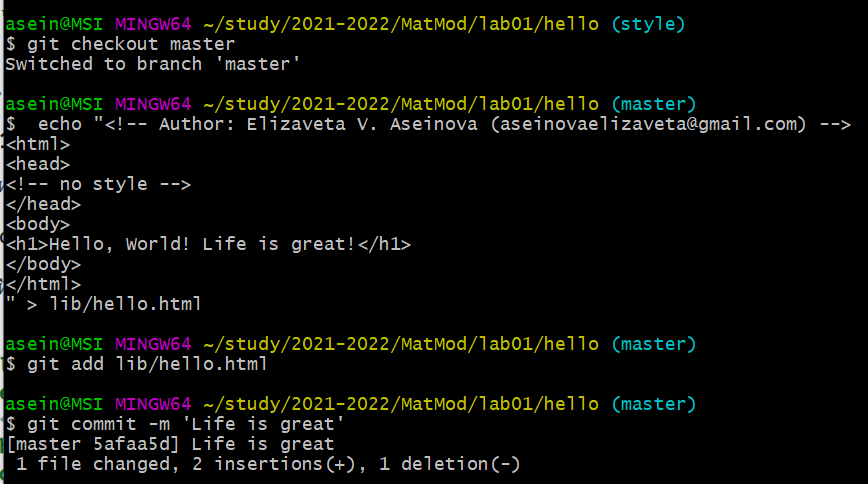


Figure 60: Конфликт

1.21.2 Просмотр веток (риc.61)

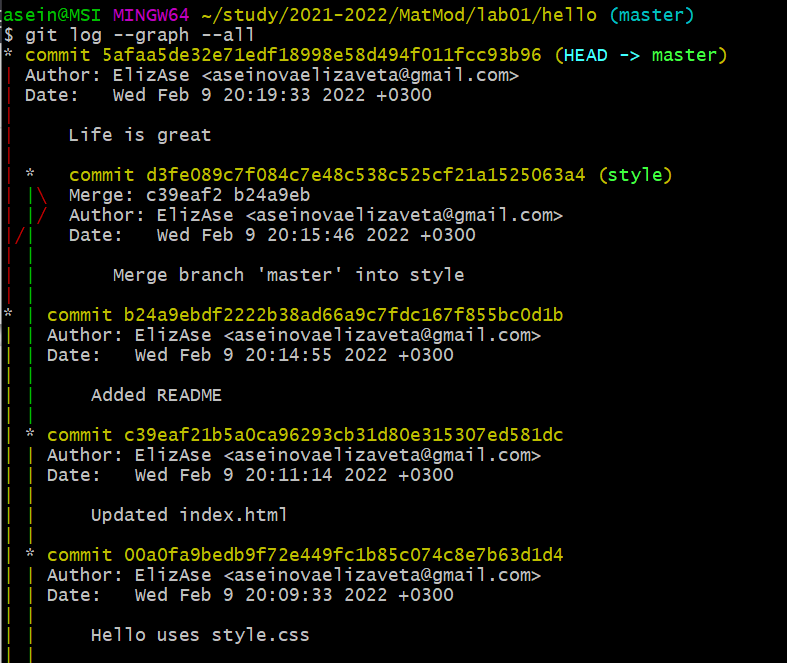


Figure 61: Просмотр веток

***1.22 Разрешение конфликтов***

1.22.1 Слияние master с веткой style (риc.62)

Теперь вернемся к ветке style и попытаемся объединить ее с новой веткой master.

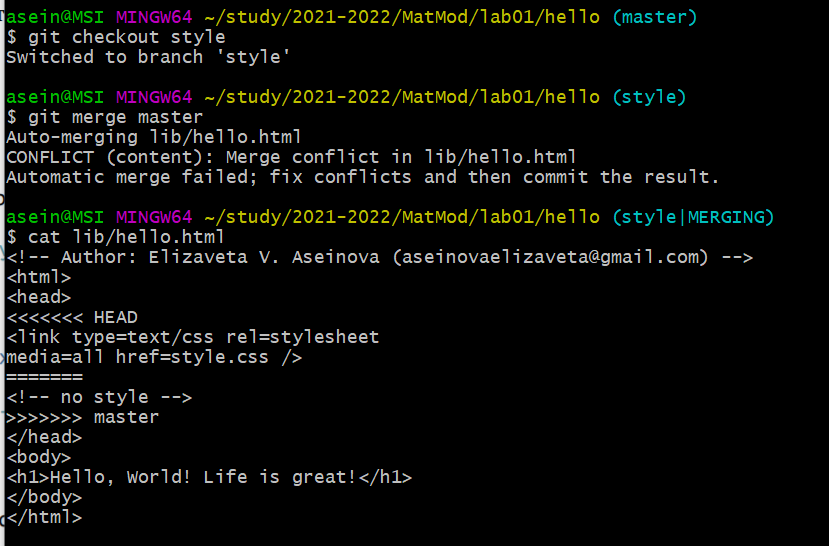


Figure 62: Слияние

1.22.2 Решение конфликта (риc.63)

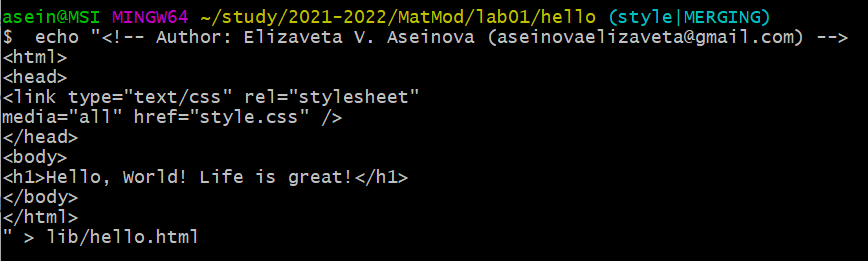


Figure 63: Решение конфликта

1.22.3 Сделайте коммит решения конфликта (риc.64)

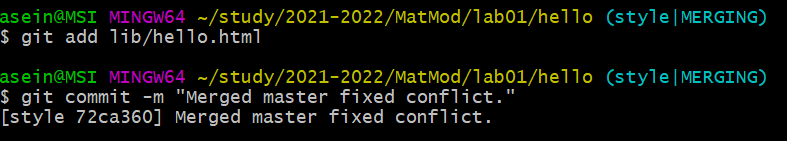


Figure 64: Коммит решения конфликта

***1.23 Сброс ветки style***

1.23.1 Сброс ветки style (риc.65)

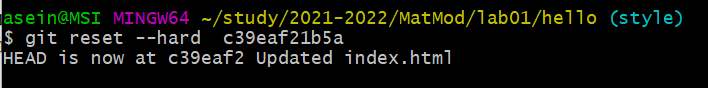


Figure 65: Сброс ветки

1.23.2 Проверьте ветку. (риc.66)

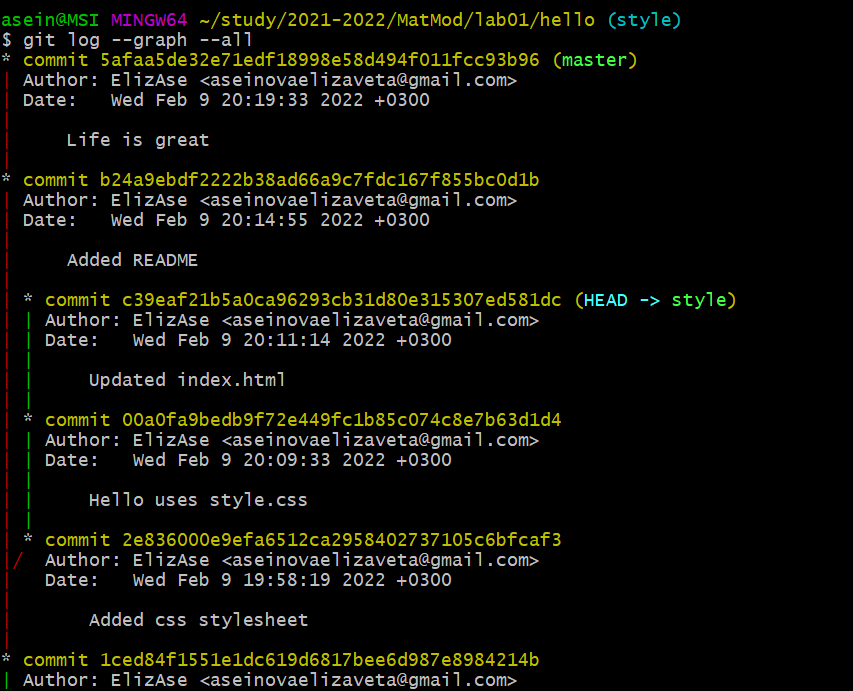


Figure 66: Проверка ветки

***1.24 Сброс ветки master***

1.24.1 Сброс ветки master (риc.67)

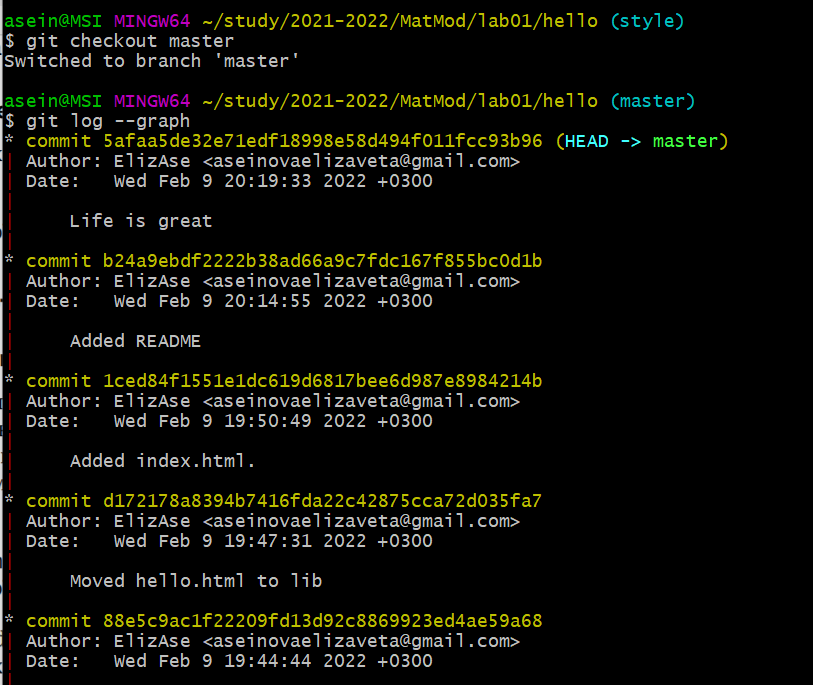


Figure 67: Сброс ветки master

***1.25 Перебазирование*** (риc.68)

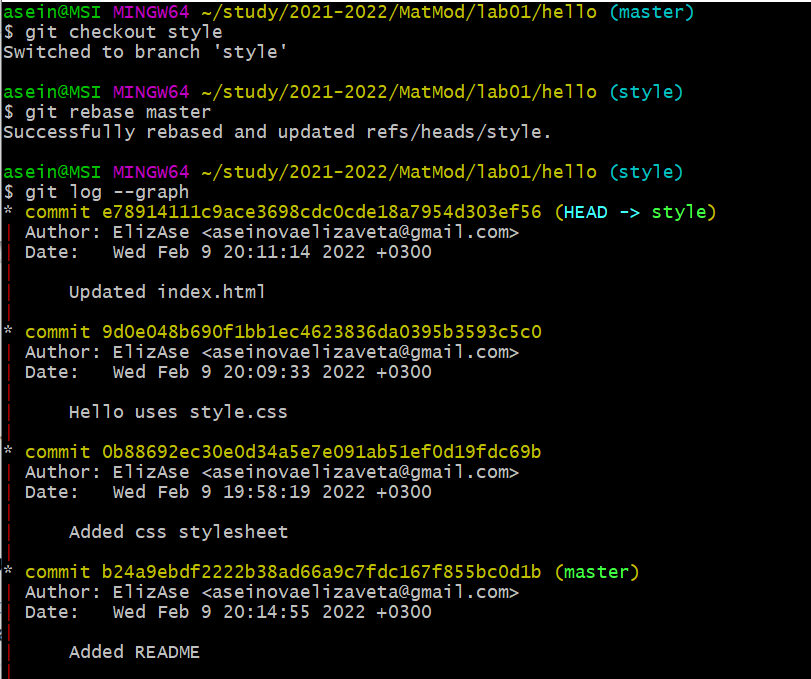


Figure 68: Перебазирование

***1.26 Слияние в ветку master***

1.26.1 Слияние style в master (риc.69)

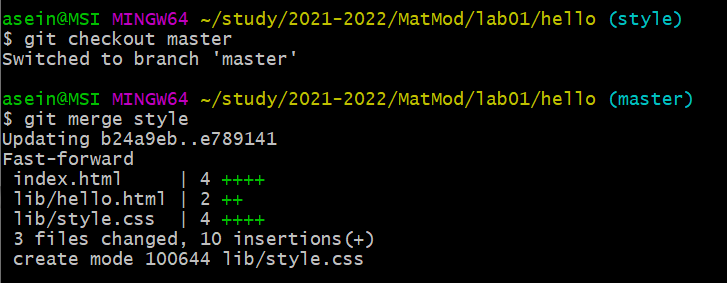


Figure 69: Слияние style в master

1.26.2 Просмотрите логи (риc.70)

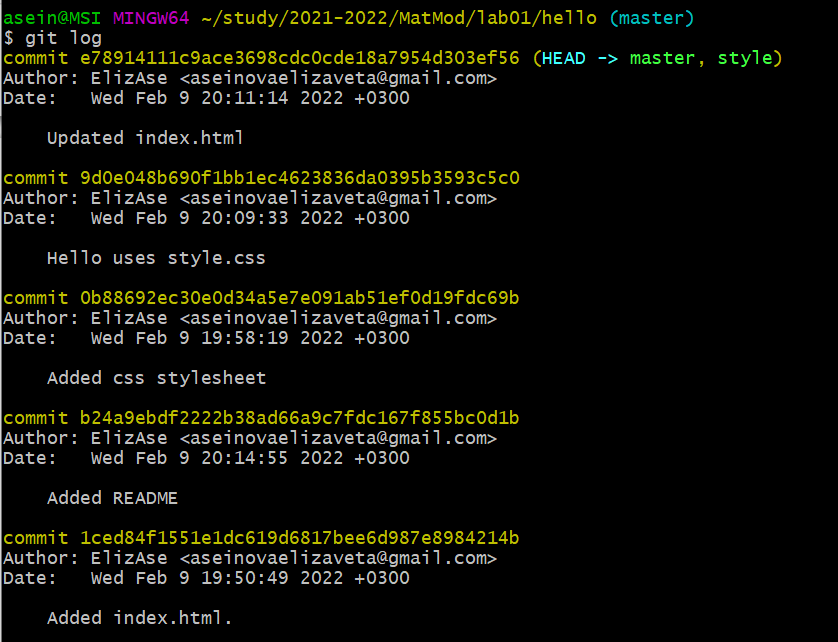


Figure 70: Просмотр логов

***1.27 Клонирование репозиториев***

1.27.1 Перейдите в рабочий каталог (риc.71)

Перейдем в рабочий каталог и сделаем клон репозитория hello.

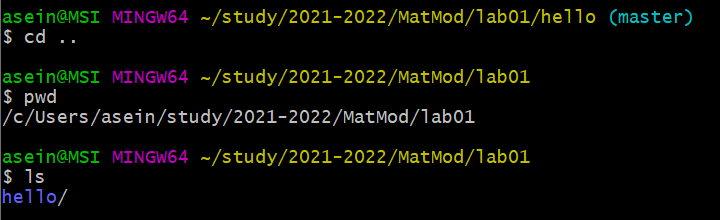


Figure 71: Переход в рабочий каталог

1.27.2 Создайте клон репозитория hello (риc.72)

Создадим клон репозитория.

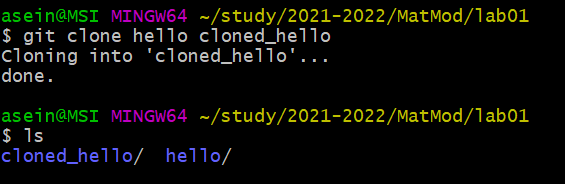


Figure 72: Клон репозитория

***1.28 Просмотр клонированного репозитория***

1.28.1 Давайте взглянем на клонированный репозиторий. (риc.73)

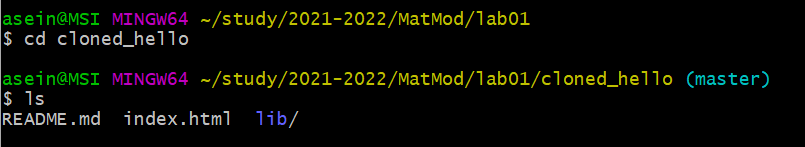


Figure 73: Клонированный репозиторий

1.28.2 Просмотрите историю репозитория (риc.74)

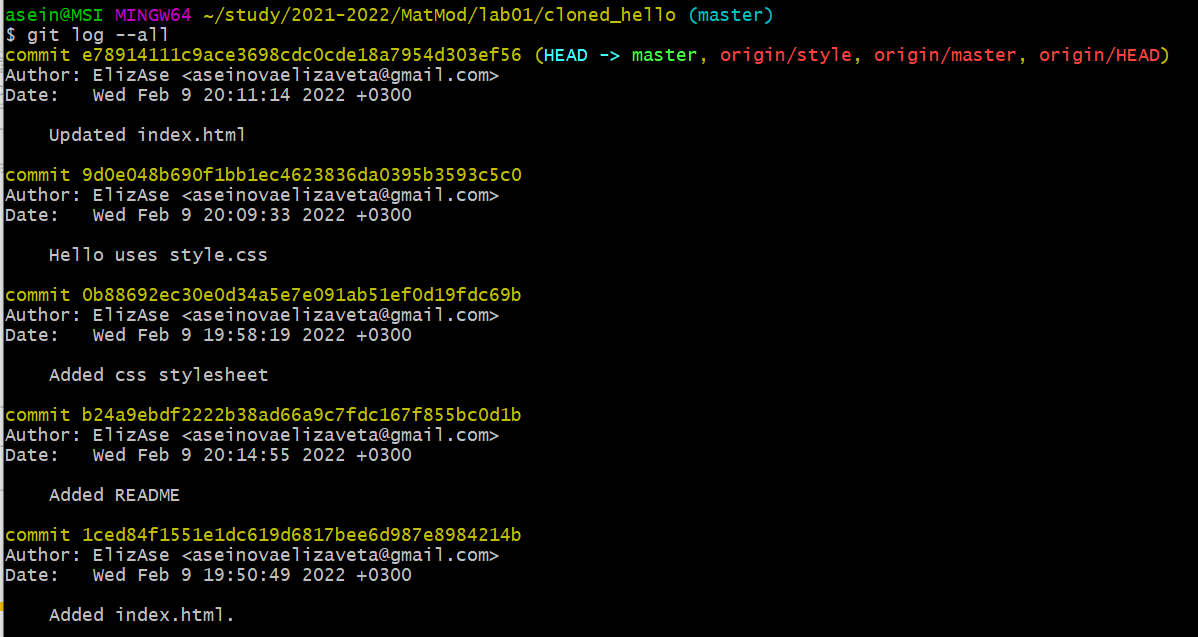


Figure 74: История репозитория

***1.29 Что такое origin?*** (риc.75)

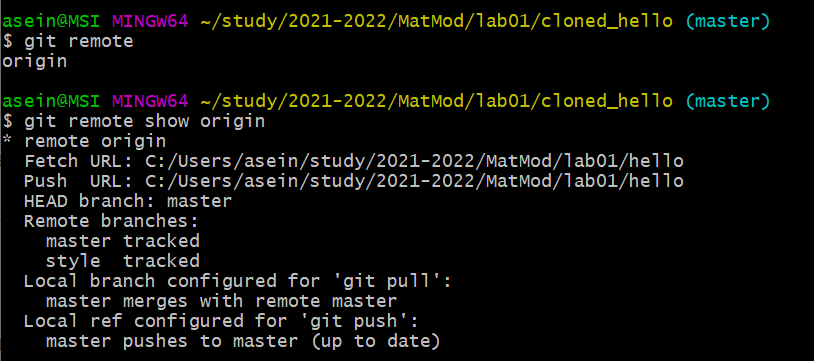


Figure 75: Origin

***1.30 Удаленные ветки*** (риc.76)

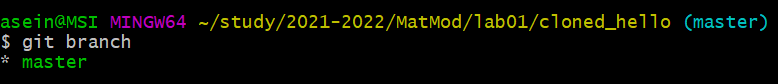


Figure 76: Удаленные ветки

1.30.1 Список удаленных веток (риc.77)

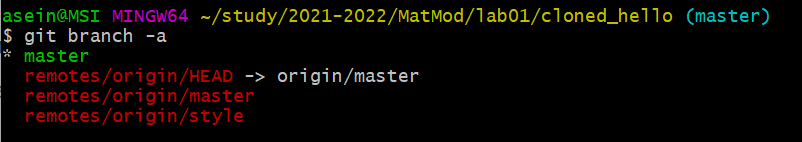


Figure 77: Список удаленных веток

***1.31 Изменение оригинального репозитория***

1.31.1 Внесите изменения в оригинальный репозиторий hello (риc.78)

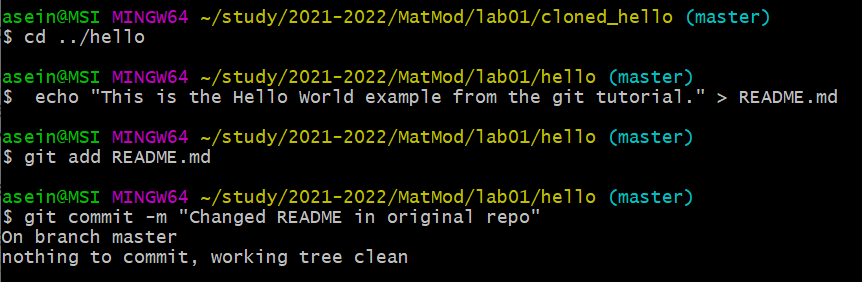


Figure 78: Изменения

1.31.2 Извлечение изменений (риc.79)

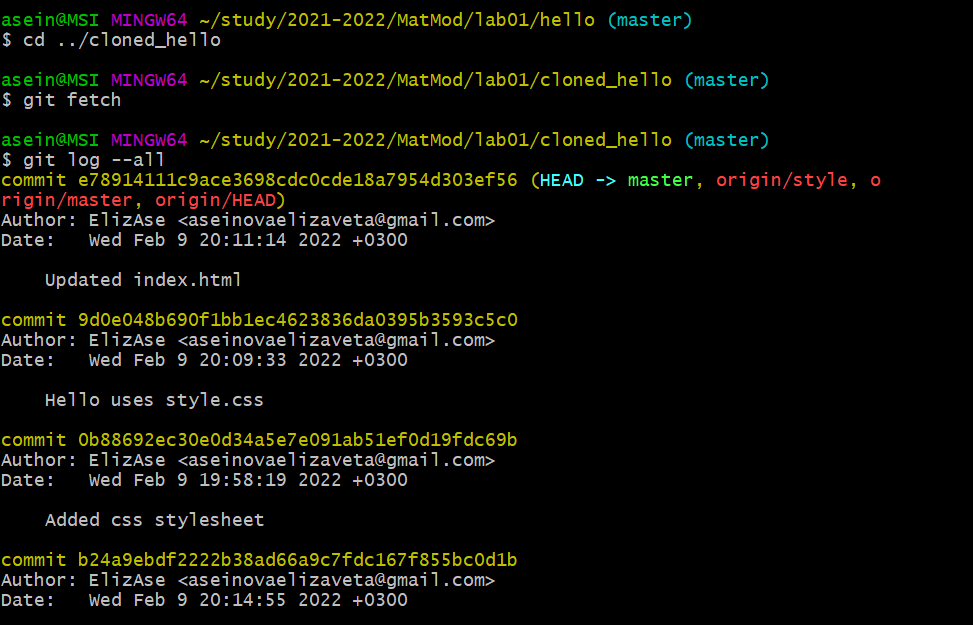


Figure 79: Извлечение изменений

1.31.3 Проверьте README.md (риc.80)

Мы можем продемонстрировать, что клонированный файл README.md не изменился.

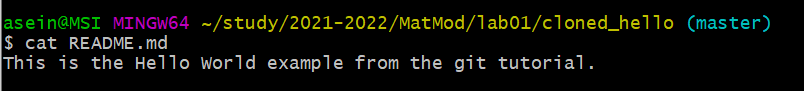


Figure 80: Проверка README

***1.32 Слияние извлеченных изменений***

1.32.1 Слейте извлеченные изменения в локальную ветку master (риc.81)

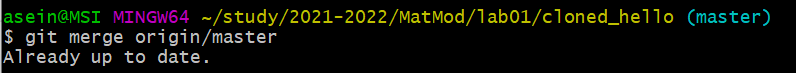


Figure 81: Извлеченные изменения в master

1.32.2 Еще раз проверьте файл README.md (риc.82)

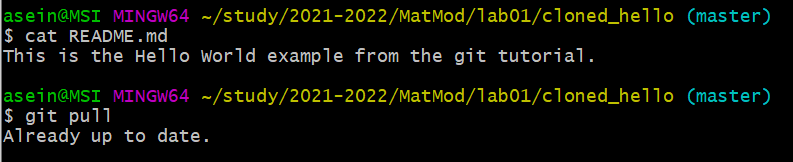


Figure 82: Проверка README 2

***1.33 Добавление ветки наблюдения***

1.33.1 Добавьте локальную ветку, которая отслеживает удаленную ветку (риc.83)



Figure 83: Добавление локальной ветки

***1.35 Создайте чистый репозиторий*** (риc.84)

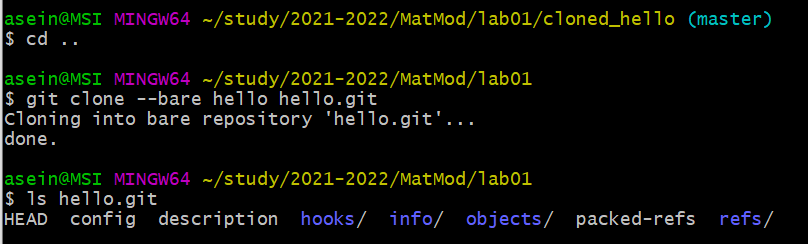


Figure 84: Чистый репозиторий

***1.36 Добавление удаленного репозитория*** (риc.85)

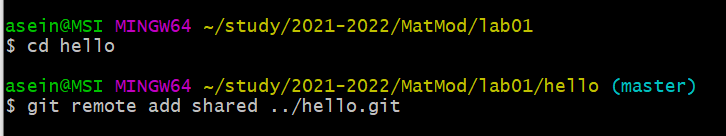


Figure 85: Чистый репозиторий

***1.37 Отправка изменений*** (риc.86)

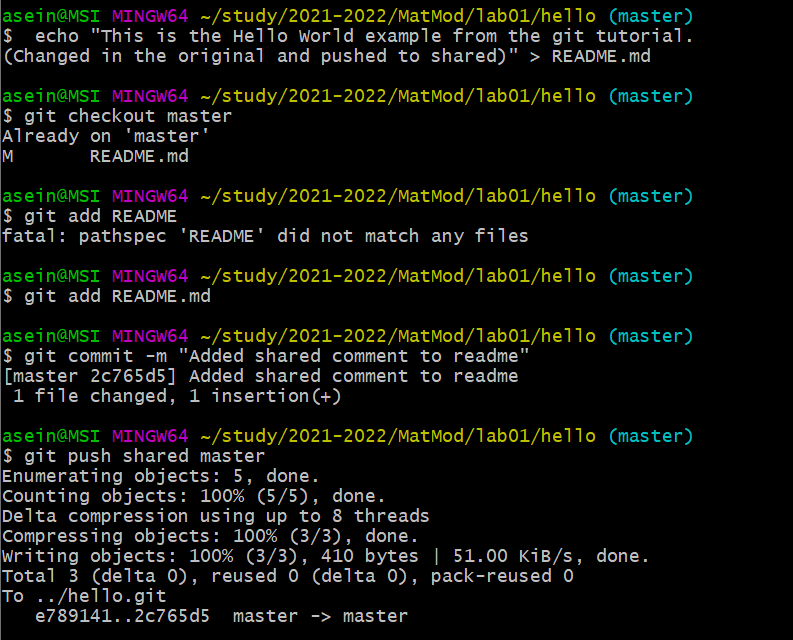


Figure 86: Отправка изменений

***1.38 Извлечение общих изменений*** (риc.87)

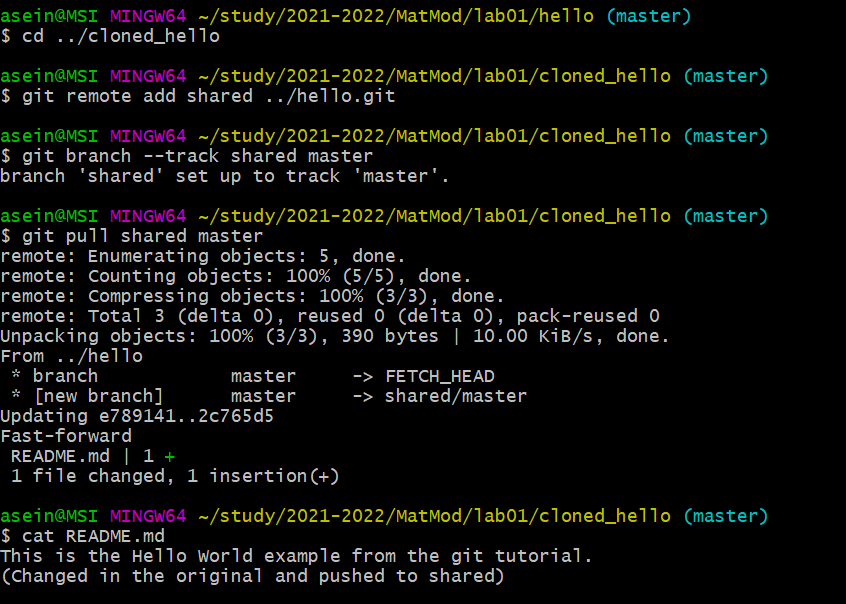


Figure 87: Извлечение общих изменений

# Выводы

Благодаря данной лабораторной работе я подробнее познакомилась c системой управления версиями Git и обрела базовые навыки для дальнейшего использования, а также закрепила их на практике.

# Список литературы

1. Кулябов, Д.С. Работа с Git [Текст] / Д.С.Кулябов. - Москва: - 31 с.
2. Rick Umali. Learn Git in a Month of Lunches [Текст] / Rick Umali. - Boston: 2015. -376 c.