Міністерство освіти і науки

Національний університет «Львівська політехніка»

Інститут комп’ютерних наук та інформаційних технологій

Кафедра автоматизованих систем управління

**Лабораторна робота №1-53**

**з дисципліни**

командна робота та презентаційні навички

Тема:

**«*Git*»**

Виконав:  
Ст. гр. КН-103 Cалтан В.С.  
Перевірила:  
Доцент Шпак З.Я.

**Львів 2018**

**Лабораторна работа 1  
Налаштування**

Налаштування імені та пошти

git config --global user.name "WhatIsLoveBabyDon’tHurtMe"

git config --global user.email "BabyDon’tHurtMe@gmail.com"

Налаштування параметрів закінчення ліній

git config --global core.autocrlf true

git config --global core.safecrlf true



**Лабораторна работа 3**

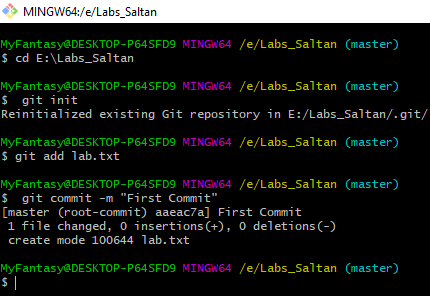
**Створення проекту**

cd E:\Labs\_Saltan

git init

git add lab.txt

git commit -m "First Commit"

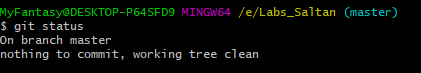


**Лабораторна робота 4**

**Перевірка статусу**

Перевіряю статус репозиторія

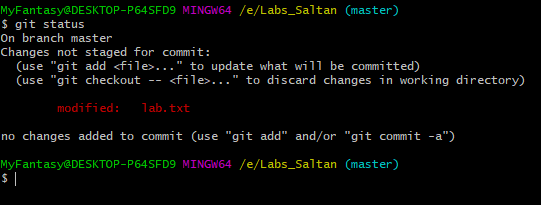
git status



**Лабораторна робота 5**

**Внесення змін**

Вношу зміни до файла та перевіряю статус.

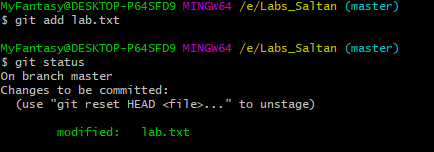


**Лабораторна робота 6  
 Індексація змін**

Індексуємо зміни в файлі lab.txt та перевіряємо стан.

git add lab.txt

git status



**Лабораторна робота 7**

**Індексація та комміт**

Роблю файли lab.txt і lab2.txt одним коммітом, а lab3.txt не повязанний з ними.

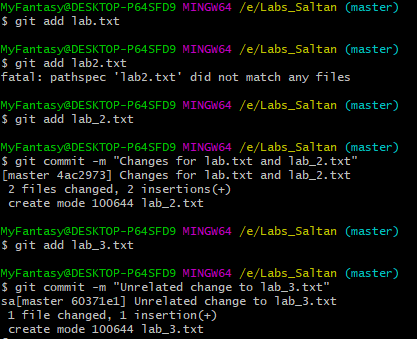
git add lab.txt

git add lab\_2.txt

git commit -m "Changes for lab.txt and lab\_2.txt"

git add lab\_3.txt

git commit -m "Unrelated change to lab\_3.txt"

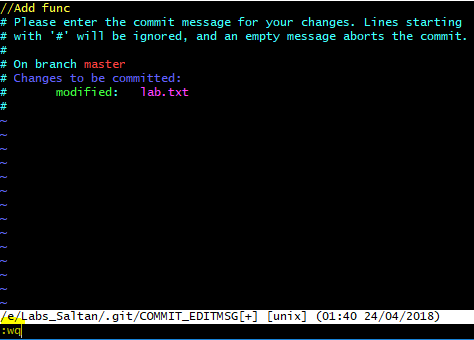


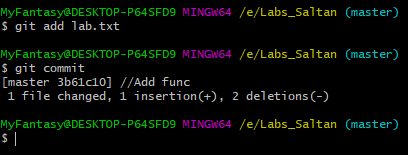
**Лабораторна робота 8**

**Комміт змін**

Коли я раніше використовував git commit для комміта першої версії файлу lab.txt в репозиторії, я включив мітку -m, яка робить коментар у командному рядку. Команда commit дозволить інтерактивно редагувати коментарі для комміту. Спробуємо зробити це без мітки –m

git commit





**Лабораторна робота 9**

Git фокусується на змінах в файлі, а не самому файлі. Коли ви здійснюєте команду git add file, ви не говорите git додати файл в репозиторій. Швидше ви говорите, що git треба відзначити поточний стан файлу, Комміт якого буде проведено пізніше. Зміни, а не файли

Роблю зміни

Додаю їх командою git add

Роблю зміни знову

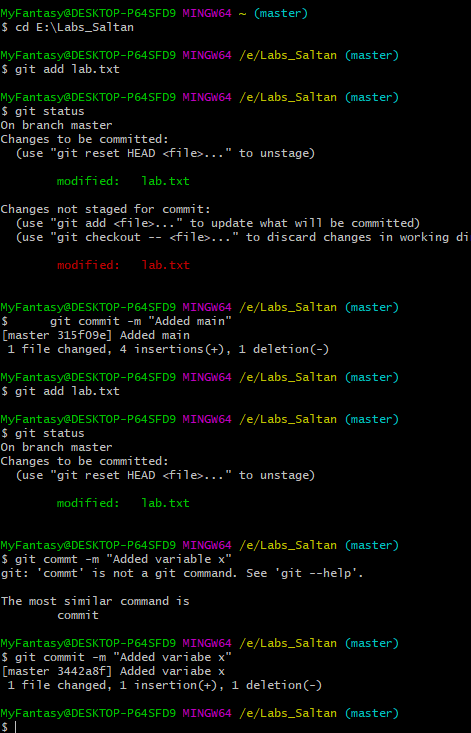
Перевіряю статус git status

Роблю комміт першої зміни

git commit -m "Added main"

Додаю другу зміни в індекс та перевіряю статус

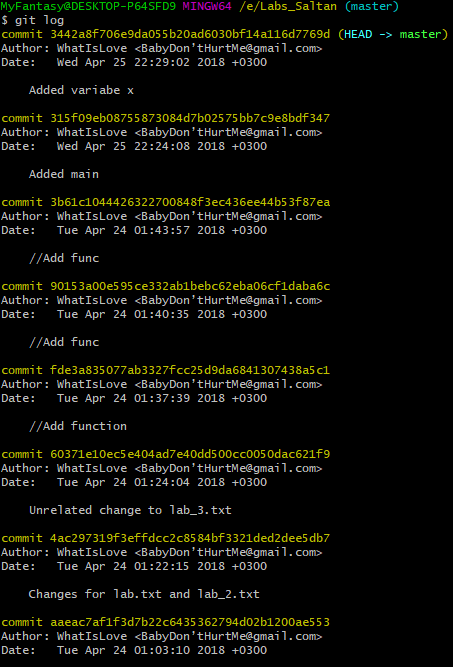
Роблю комміт другої зміни



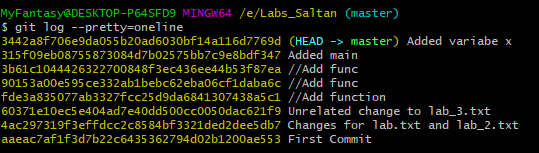
**Лабораторна робота 10**

**Історія**

Отримання списка проведених змін – git log



Одностроковий формат git log --pretty=oneline



Ми також можемо контролювати відображення записів за допомогою команд:

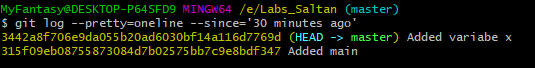
git log --pretty=oneline --max-count=2

git log --pretty=oneline --since='5 minutes ago'

git log --pretty=oneline --until='5 minutes ago'

git log --pretty=oneline --author=<your name>

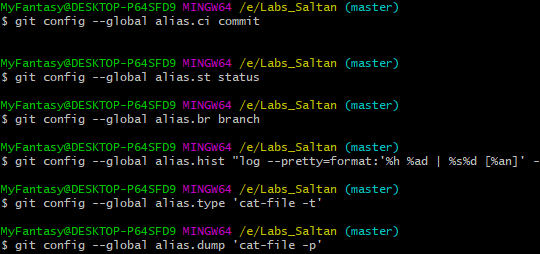
git log --pretty=oneline –all



**Лабораторна робота 11**

**Аліаси**

Налаштовуємо аліаси і шорткати для команд git

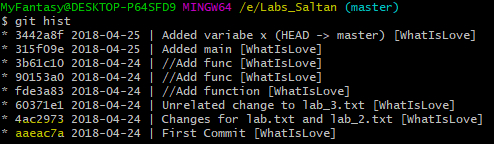


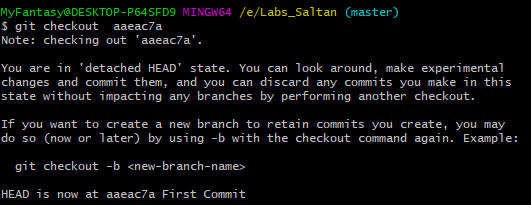
**Лабораторна робота 12**

**Отримання старих версій**

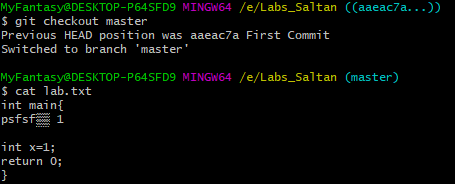
Візьмемо хеш первого комміта за допомою команди git log (git hist)

Перевірямо вміст файла





Повернемось до останньої версії в бренчі master



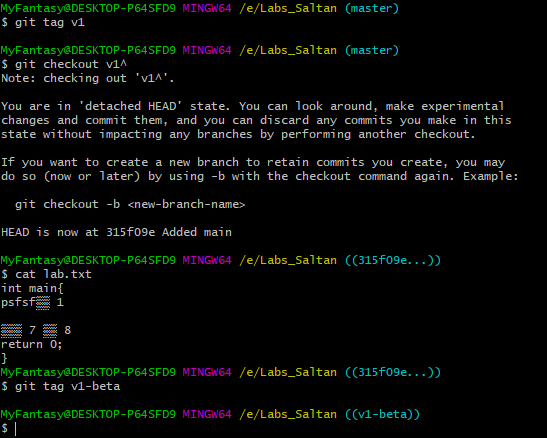
**Лабораторна робота 13**

**Cтворення тегів версій**

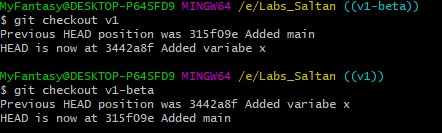
Створено тег першої версії

git tag v1

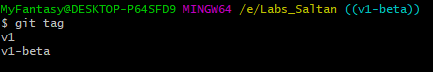
Створемо тег для попередньої версії за допомою ‘^’



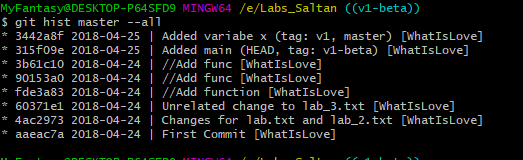
Переключення по імені тега



Перегляд тегів



Перегляд тегів в історії



**Лабораторна робота 14**

**Відміна локальних змін**

Переключимся на гілку мастер та отримаємо статус файла в якому вже зроблені зміни, які потрібно відмінити

Git checkout master

Git status

Використаємо checkout для переключення в версію файла lab.txt в репозиторії.

Git checkout lab.txt

Git status

Cat lab.txt

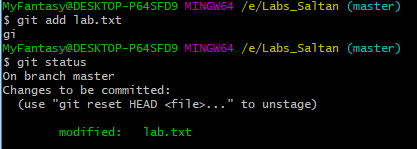
**Лабораторна робота 15**

**Відміна проіндексированих змін (перед коммітом)**

Вносимо зміни та індесуємо її. Перевіряємо статус.

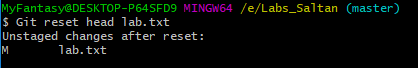
Git add lab.txt

Git status



Робимо зброс буферної зони

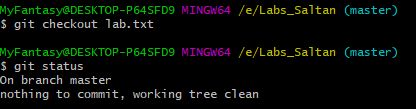
Git reset head lab.txt



Переключимось на версії коміта

Git checkout lab.txt

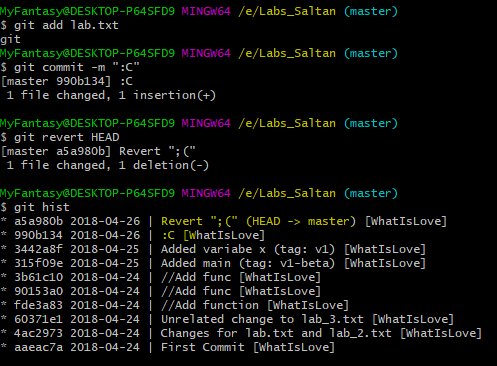
Git status



**Лабораторна робота 16**

**Відміна коммітів**

Змінюємо файл та робимо комміт, а потім робимо комміт з новими змінами, які відміняють попередні (git revert HEAD)



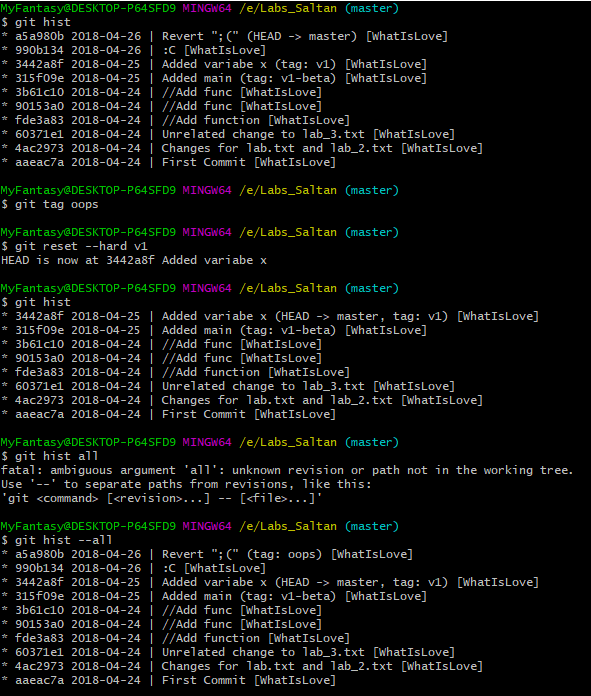
**Лабораторна робота 17**

**Видалення коммітів з гілки**

Видалемо комміти за допомогою команди reset.

Відмітимо останній комміт, а потім сбросимо комміти до попереднього комміту.

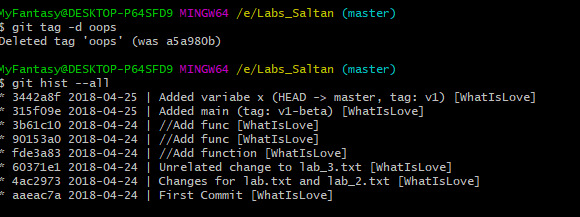
Але комміти все ще будуть знаходитись в репозиторії. Ми все ще можемо на переключатись, адже вони просто не містяться в гілці мастер. Якщо б ми не відмітили їх тегами, то вони б і далі знаходились в репозиторії.



**Лабораторна робота 18**

**Видалення тега**

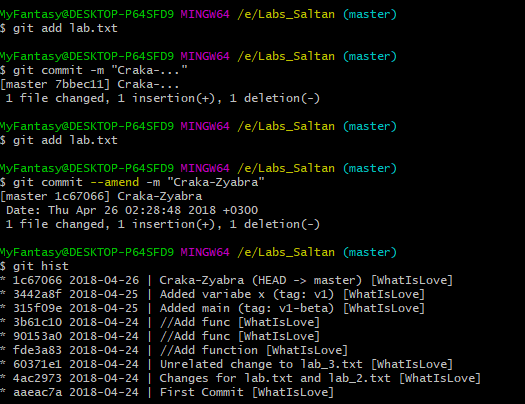
Видалимо тег oops. Видалемо його комміти на які він ссилався збиральником сміття. Після цього тег не буде відображатись в репозиторії.



**Лабораторна робота 19**

**Внесення змін в комміти**

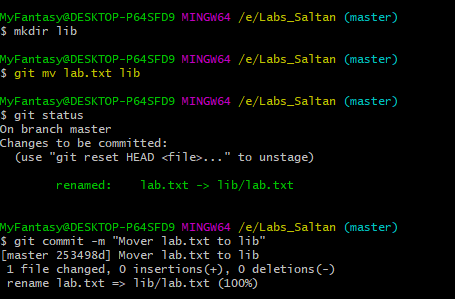
Змінюємо попереднії комміт.



**Лабораторна робота 20**

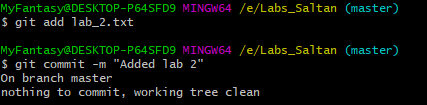
**Переміщення файлів**

Переміщуємо файл в каталог lib



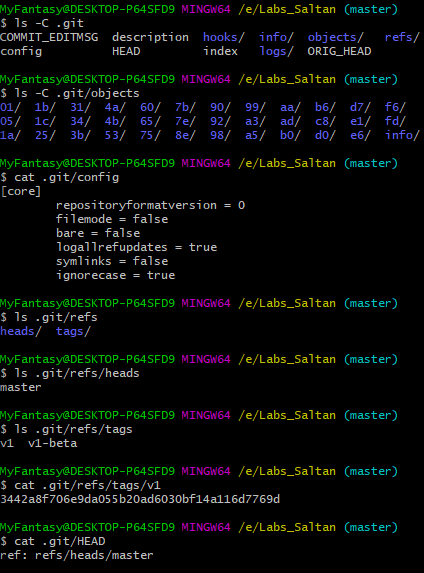
**Лабораторна робота 21**

**Завантаження файлу в репозиторій**



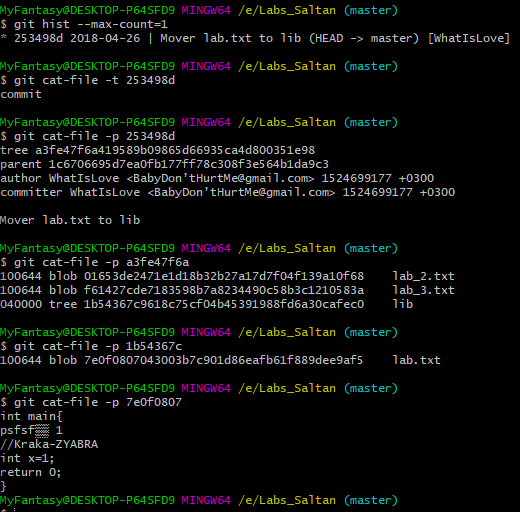
**Лабораторна робота 22**

**Git всередині. Каталог .git**



**Лабораторна робота 23**

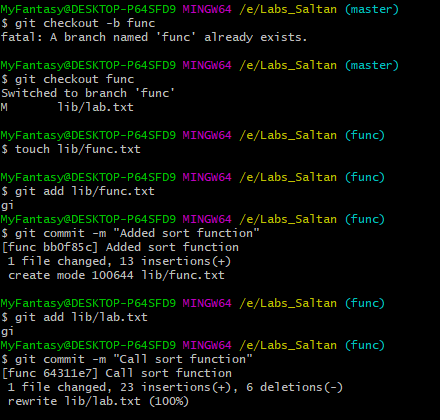
**Git робота всередині.**



**Лабораторна робота 24**

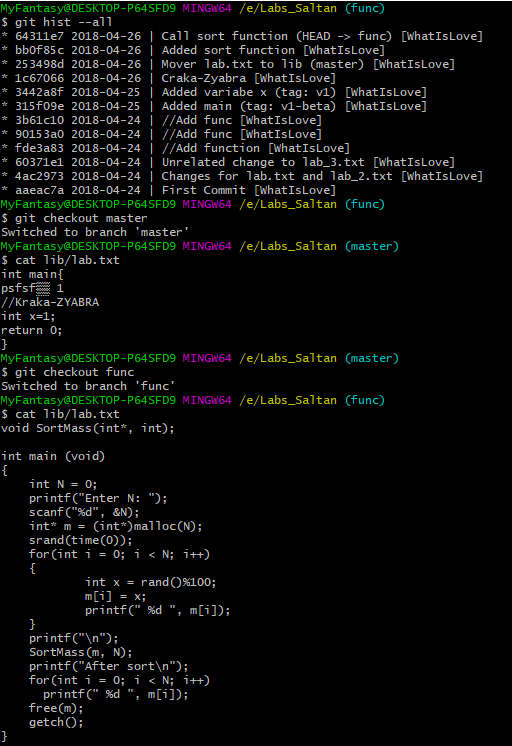
**Cтворення гілки**

Git checkout –b func   
\*-b шорткат для branch



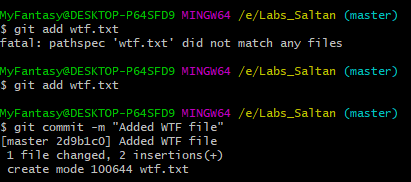
**Лабораторна робота 25**

**Навігація по гілкам**



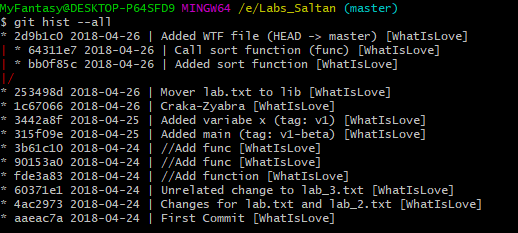
**Лабораторна робота 26**

**Зміни в бренчі Мастер**



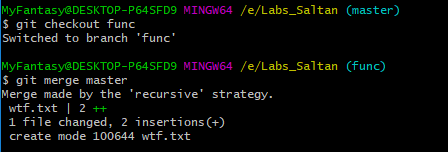
**Лабораторна робота 27**

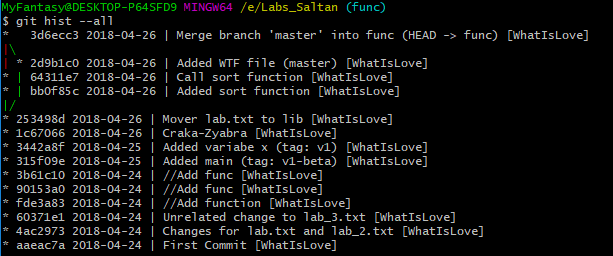
Тепер ми маємо дві різні гілки



**Лабораторна робота 28**

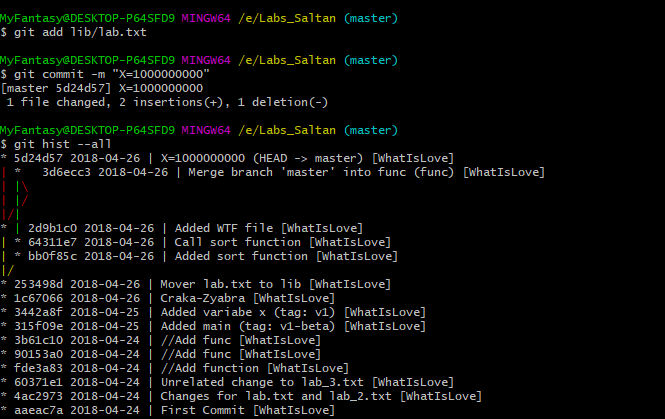
**Злиття гілок**





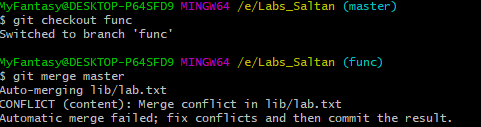
**Лабораторна робота 29**

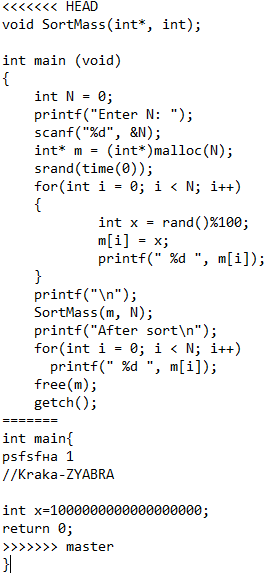
**Створення конфлікту**

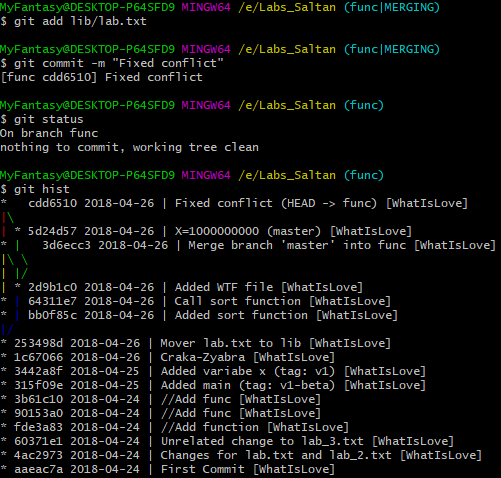


**Лабораторна робота 30**

**Вирішення конфлікту**







**Лабораторна робота 31**

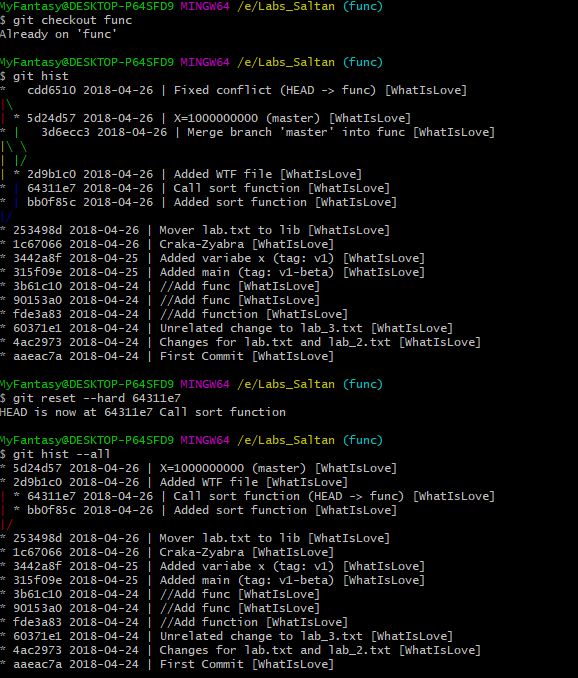
**Перебазування як альтернатива злиттю**

Розглянемо різницю між перебазуванням та злиттям в наступних лабораторних.

Будемо використовувати команду reset для повернення до попереднього стану.

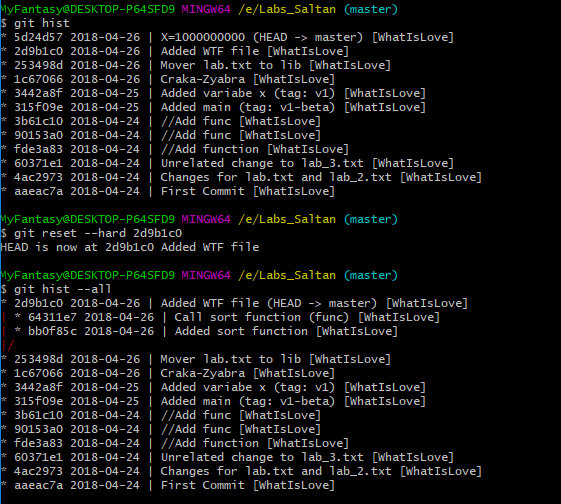
**Лабораторна робота 32**

**Зброс гілкі func**



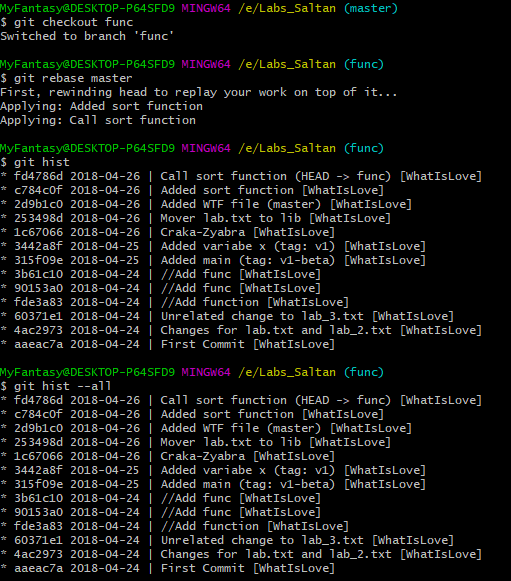
**Лабораторна робота 33**

**Зброс гілки Мастер**



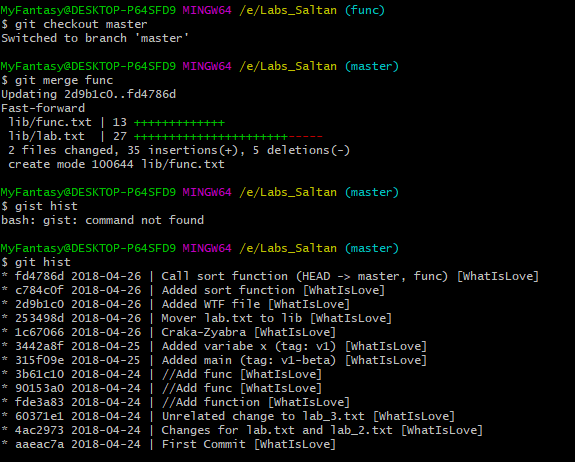
**Лабораторна робота 34**

**Перебазування**



**Лабораторна робота 35**

**Злиття в гілку Мастер**



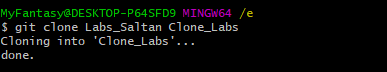
**Лабораторна робота 36**

**Декілька репозиторіїв**

У наступному розділі створимо новий репозиторій з ім'ям «Clone\_Labs». Ми покажемо, як переміщувати зміни з одного сховища в інший і як вирішувати конфлікти, що виникають в результаті роботи з двома репозиторіями.

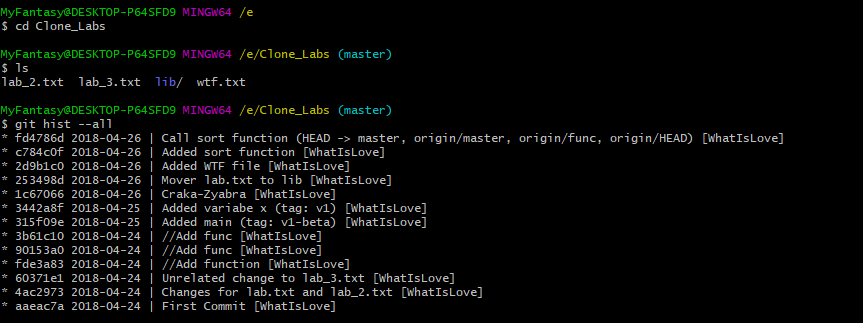
**Лабораторна робота 37**

**Клонування репозиторіїв**



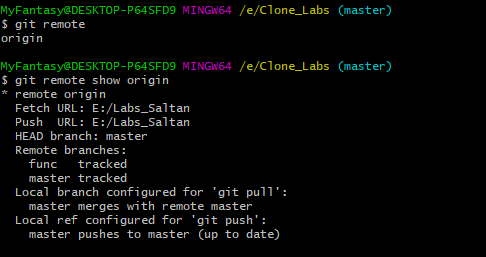
**Лабораторна робота 38**

**Перегляд клонованого репозиторія**



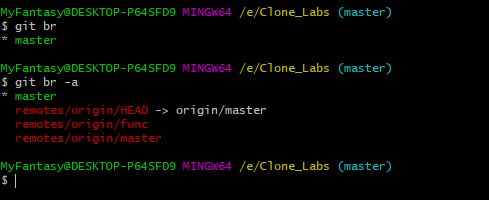
**Лабораторна робота 39**

**Origin**



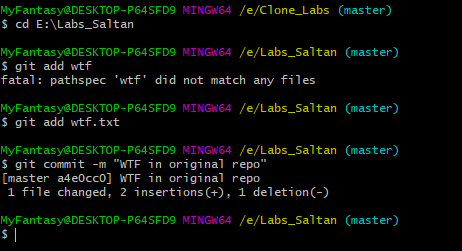
**Лабораторна робота 40**

**Видалені гілки**



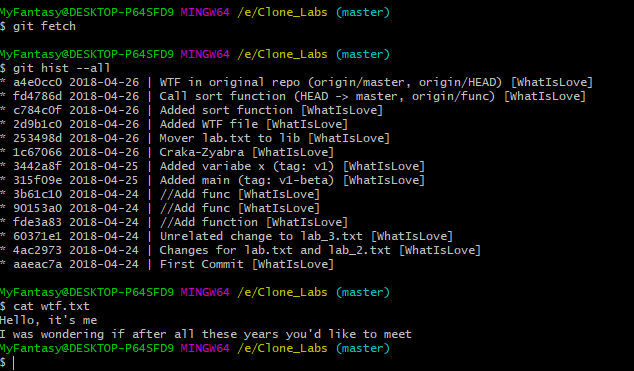
**Лабораторна робота 41**

**Зміна оригінально репозиторія**



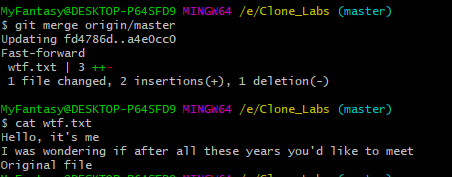
**Лабораторна робота 42**

**Витяг змін**



**Лабораторна робота 43**

**Злиття витягнутих змін**



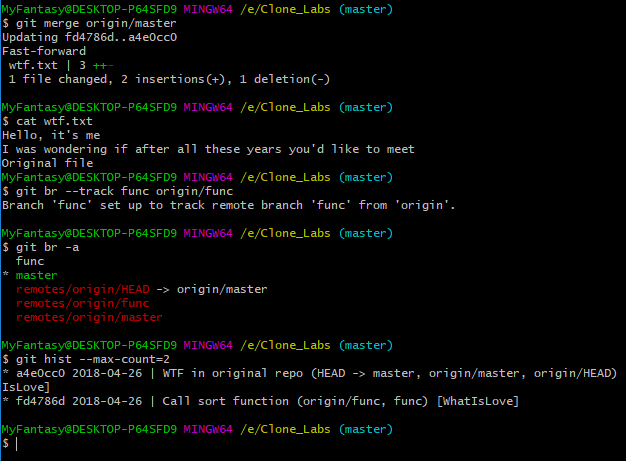
**Лабораторна робота 44**

**Витяг та злиття змін одною командою**

Git pull еквівалентне наступним двум командам: git fetch git merge origin/master

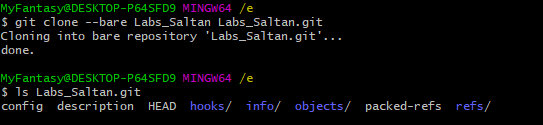
**Лабораторна робота 45**

**Створення гілкі нагляду**



**Лабораторна робота 46**

**Чисті репозиторії**



**Лабораторна робота 47**

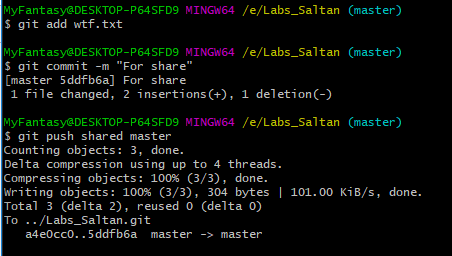
**Добавлення видаленого репозиторія**

Додамо чистий репозиторій в якості видаленого репозиторія до нашого оригінального репозиторія.



**Лабораторна робота 48**

**Відправка змін**



**Лабораторна робота 49**

**Витяг загальних змін**

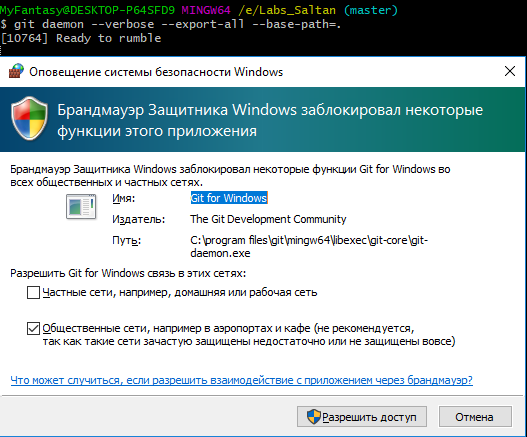
Cd ../Clone\_Labs

Git br –track shared master

Git pull share master

**Лабораторна робота 50**

**Розміщення git репозиторіїв**



**Лабораторна робота 51**

**Лабораторна робота 52**

**Лабораторна робота 53**

Лабораторні 50, 51, 52, 53 були виконані через інший репозиторій. Можу їх також загрузити на github