Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет

информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра Информатики

Дисциплина «Метрология, стандартизация и сертификация (в инормационных технологиях)»

**ОТЧЕТ**

к лабораторной работе №1

на тему:

**«Метрики размера программ»**

БГУИР 6-05-0612-02

|  |
| --- |
| Выполнила студентка группы 353501  ГОРОДЕЦКАЯ Диана Анатольевна  МАЛИНОВСКАЯ Евгения Александровна |
| Проверила доцент каф. Информатики  БОЛТАК Светлана Владимировна |
|  |
| (дата, подпись преподавателя) |

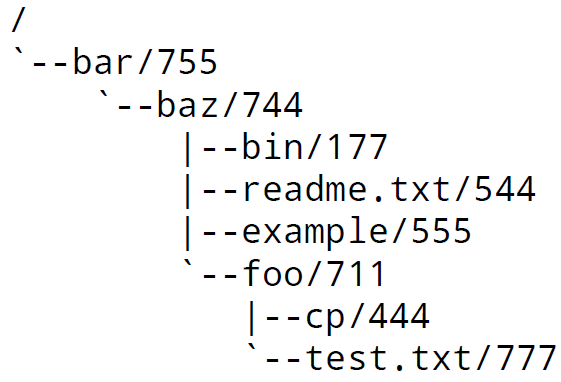
Минск 2025

**1 ЦЕЛЬ ВЫПОЛНЕНИЯ ЗАДАНИЯ**

Познакомиться с возможностями и получить практические навыки работы с системой контроля версий Git.

**2 ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ**

**Вариант 4.** Создать структуру папок согласно варианту:



**3 ВЫПОЛНЕНИЕ ЗАДАНИЯ**

Для подготовки рабочего окружения были установлены параметры *user.name* и *user.email*, что показано на рисунке 1.

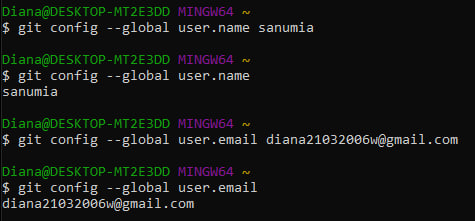


Рисунок 1 – Создание параметров *user.name* и *user.email*

Далее была создана структура папок с помощью командной строки с помощью команд *mkdir <name>* и *echo. > <name.type>* соответственно, запись в текстовые документы была произведена с помощью команды

*echo <text> > <name.type>*. Структуры папок в соответствии с индивидуальным заданием показана на рисунке 2.

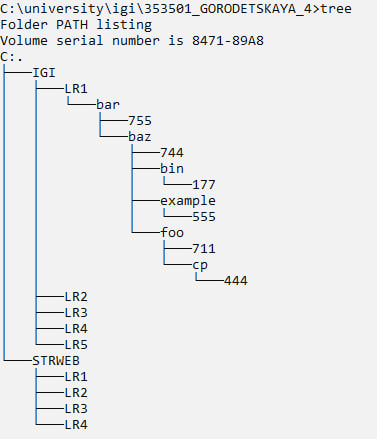


Рисунок 2 – Структура папок

Клонирование репозитория осуществляется с помощью команды git clone, а создание локального git репозитория с помощью git init, после чего в папке 353501\_GORODETSKAYA \_4 появилась папка .git, что показано на рисунке 3.



Рисунок 3 – Создание лoкального git репозитория

С помощью команды git add . были добавлены файлы в индексацию после чего был выполнен commit с именем “first commit”, что показано на рисунке 4.

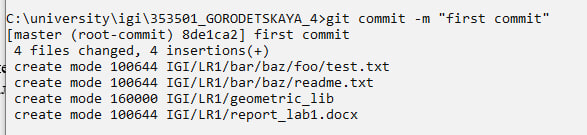


Рисунок 4 – Создание коммита к проиндексированным файлам

Добавление и удаление одного файла, группы файлов и переименование файла показано на рисунках 5, 6 и 7 соответственно.



Рисунок 5 – Создание и удаление 1 файла

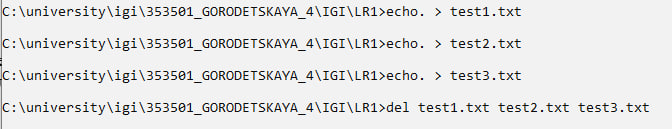


Рисунок 6 – Создание и удаление группы файлов

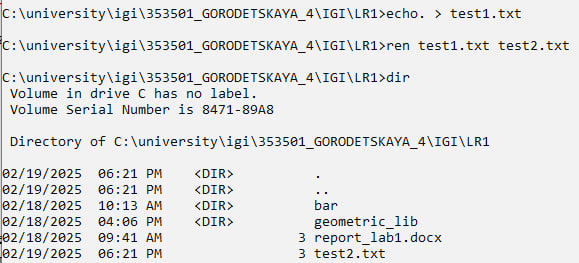


Рисунок 7 – Пример переименования файла

Для создания веток была использована команда git branch <name>, все созданные ветки показаны на рисунке 8.

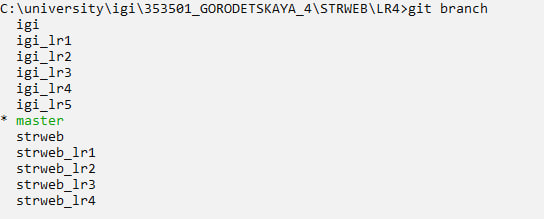


Рисунок 8 – Список всех созданных веток

Команда git log (рис. 9) используется для просмотра истории коммитов в репозитории, позволяя анализировать внесённые изменения, авторов и временные рамки разработки. Флаги в команде git log позволяют фильтровать и настраивать вывод коммитов: -n <number> (рис. 10) ограничивает количество выводимых коммитов, --since <date> и --after <date> показывают коммиты, сделанные после указанной даты, в то время как --until <date> и --before <date> отображают коммиты до определённой даты. Флаги --author <name> (рис. 11) и --committer <name> фильтруют коммиты по автору и коммиттеру соответственно, а --grep <pattern> позволяет искать коммиты по сообщениям, отображая только те, которые содержат указанный текст.



Рисунок 9 – Вывод всех коммитов

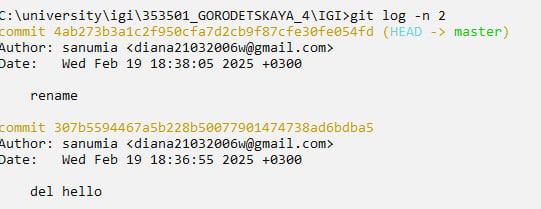


Рисунок 10 – Вывод двух последних коммитов из четырёх возможных



Рисунок 11 – Вывод всех коммитов выполненных пользователем sanumia

Следующие флаги команды git log позволяют настраивать вывод информации: -p отображает изменения (патчи) для каждого коммита (рис. 12), --stat показывает статистику изменений в файлах (рис. 13), --shortstat даёт краткую статистику изменений, --name-only выводит только имена изменённых файлов, а --name-status показывает имена файлов с их статусами. Флаг --relative-date отображает дату коммита в относительном формате

(рис. 14), --pretty позволяет настраивать формат вывода коммитов, а --graph визуализирует структуру ветвления и слияний в истории.

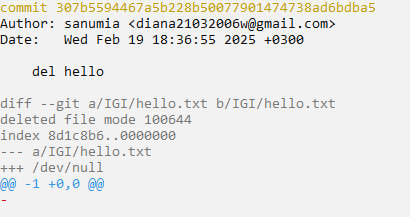


Рисунок 12 – Флаг p, показывающий изменения для коммита del hello



Рисунок 13 – Пример применения флага stat



Рисунок 14 – Пример применение флага relative-date

Для нахождения разницы между коммитами сначала были определены их хэши с помощью команды git log –oneline, а затем была найдена разница межд коммитом rename и del hello с помощью команды git diff <name> <name> (рис.15).

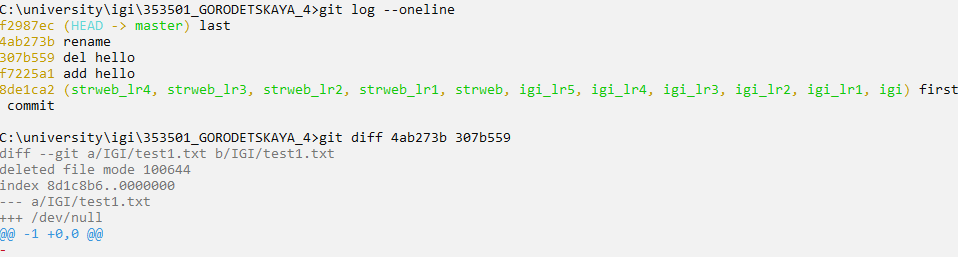


Рисунок 15 – Разница между коммитами rename и del hello

На рисунке 16 показано создание файла test.txt c строкой “номер варианта”, после чего был выполнен коммит, который затем был отменён, откатившись на 1 коммит назад с помощью команды git reset --hard HEAD~1.

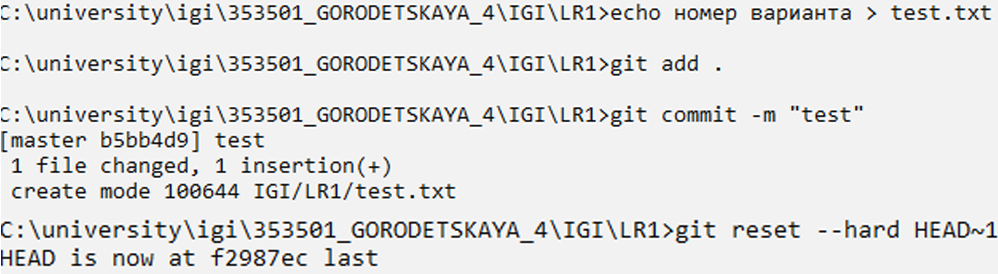


Рисунок 16 – Создание нового коммита и его отмена

Индексация файла test.txt и её отмена (git reset test.txt) показана на рисунке 17.

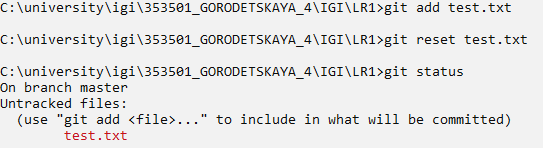


Рисунок 17 – Индексация файла и его отмена

На рисунке 18 показано изменение содержание файла test.txt с нового содержания “hey” на предыдущее “hello world” с помощью команды

git checkout -- <name>.

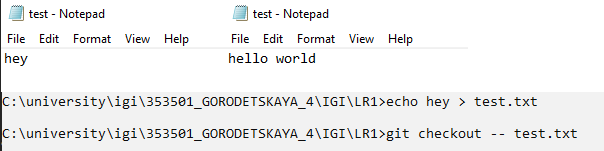


Рисунок 18 – Внесение изменения в файл и возвращение его в исходное состояние

С помощью команды git remote add origin <ssh> был добавлен удаленный репозиторий, в свою очередь команда git push origin master отправляет изменения из локальной ветки master в удалённый репозиторий с именем origin, что показано на рисунке 19.

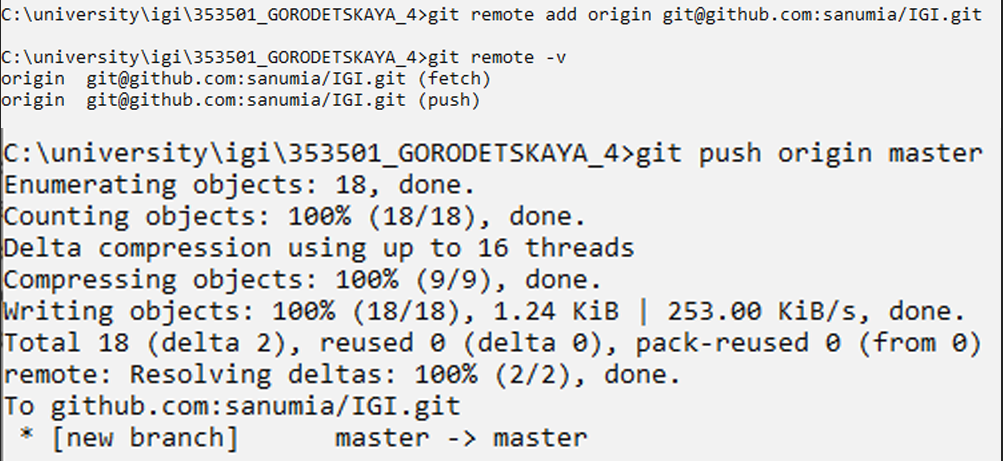


Рисунок 19 – взаимодействие с локальным репозиторием

Была создана новая ветка additional со всеми файлами основной ветки master, из которой мы в последствии удалили файл test.txt, при этом на ветке master данный файл всё ещё существует (рис. 20)

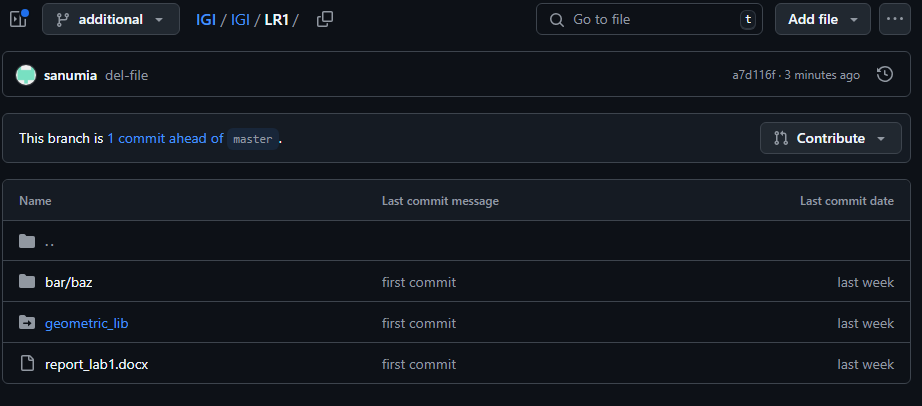


Рисунок 20 – Дополнительная ветка на удаленном репозитории

**ВЫВОД**

В ходе выполнения лабораторной работы по теме "Система контроля версий Git" были изучены ключевые возможности Git. После успешной установки и настройка Git на локальном компьютере была создана структура папок для репозитория в соответствии с заданием. В процессе работы были выполнены основные команды Git, такие как git init, git add, git commit, а также изучены способы управления ветками и индексацией файлов. Это позволило наглядно увидеть, как Git помогает отслеживать изменения и управлять проектом, обеспечивая возможность работы в команде без конфликтов.

Также был проведен ряд операций с удаленным репозиторием на GitHub, включая создание и переименование репозитория, а также перенос изменений между ветками. Все действия фиксировались с помощью коммитов, что позволило строить историю изменений проекта и использовать различные флаги команды git log для глубокого анализа.