**МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ ОРДЕНА ТРУДОВОГО КРАСНОГО ЗНАМЕНИ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «МОСКОВСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ СВЯЗИ И ИНФОРМАТИКИ»**

Кафедра математической кибернетики и информационных технологий

Лабораторная работа №2

по дисциплине «Кроссплатформенные технологии программирования»

на тему «Методы ветвления в системе контроля версий»

Выполнил студент

группы БСТ2003

Чечеткин С.А.

Проверила:

Полянцева К.А.

Москва 2022

**Содержание**

[Цель и задание 2](#_Toc95334809)

[Ход выполнения работы 3](#_Toc95334810)

[1. Настроить игнорирование git’ом файлов определенного расширения с использованием glob-шаблонов 3](#_Toc95334811)

[2. Вывести лог коммитов в измененном пользовательском формате 4](#_Toc95334812)

[3. Переименовать ветку 5](#_Toc95334813)

[4. Провести в ветке работу и слить с веткой master, удалить слитую ветку 5](#_Toc95334814)

[5. Настроить удалённый репозиторий: получить изменения из репозитория и отправить изменения. 6](#_Toc95334815)

[6. Организовать совместную работу над проектом в соответствии с выбранной идеологией 8](#_Toc95334816)

[Вывод 9](#_Toc95334817)

[Список использованных источников: 10](#_Toc95334818)

# Цель и задание

**Цель:** изучение подходов к командной разработке в системе контроля версий git

**Задание:** В созданном в предыдущей работе либо вновь созданном репозитории:

1. Настроить игнорирование git’ом файлов определенного расширения с использованием glob-шаблонов
2. Вывести лог коммитов в измененном пользовательском формате
3. Переименовать ветку
4. Провести в ветке работу и слить с веткой master, удалить слитую ветку
5. Настроить удалённый репозиторий: получить изменения из репозитория и отправить изменения.
6. Организовать совместную работу над проектом в соответствии с выбранной идеологией

# Ход выполнения работы

## 1. Настроить игнорирование git’ом файлов определенного расширения с использованием glob-шаблонов

В папке создаём файл с расширением .gitignore. После открываем его с помощью текстового редактора Блокнот и вносим в него расширения, которые необходимо игнорировать.

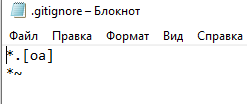


Рисунок 1 – Внесение расширений для игнорирования

Далее делаем коммит с добавленным файлом .gitignore

git add .gitignore

git commit -m «added .gitignore»

После чего буду игнорироваться любые файлы заканчивающиеся на «.o» или «.a» — объектные и архивные файлы, которые могут появиться во время сборки кода. Вторая строка предписывает игнорировать все файлы заканчивающиеся на тильду (~), которая используется во многих текстовых редакторах, например Emacs, для обозначения временных файлов.

## 2. Вывести лог коммитов в измененном пользовательском формате

git log – вывод истории коммитов

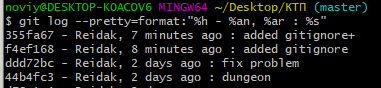


Рисунок 2 – Вывод лог коммитов

Наиболее интересной опцией является format, которая позволяет указать формат для вывода информации. Особенно это может быть полезным когда мы хотим сгенерировать вывод для автоматического анализа — так как мы указываем формат явно, он не будет изменен даже после обновления Git:

Полезные опции для git log --pretty=format отображает наиболее полезные опции для изменения формата.

| **Опция** | **Описания вывода** |
| --- | --- |
| %H | Хеш коммита |
| %h | Сокращенный хеш коммита |
| %T | Хеш дерева |
| %t | Сокращенный хеш дерева |
| %P | Хеш родителей |
| %p | Сокращенный хеш родителей |
| %an | Имя автора |
| %ae | Электронная почта автора |
| %ad | Дата автора (формат даты можно задать опцией --date=option) |
| %ar | Относительная дата автора |
| %cn | Имя коммитера |
| %ce | Электронная почта коммитера |
| %cd | Дата коммитера |
| %cr | Относительная дата коммитера |
| %s | Содержание |

## 3. Переименовать ветку

Необходимо перейти на ветку, которую будем переименовывать и с помощью команды, представленной на рисунке ниже, переименовать ветку.

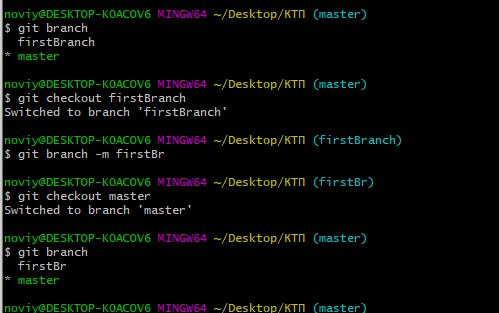


Рисунок 3 – Изменение названия ветки

## 4. Провести в ветке работу и слить с веткой master, удалить слитую ветку

На рисунке представлены команды для слияния и удаления ветки

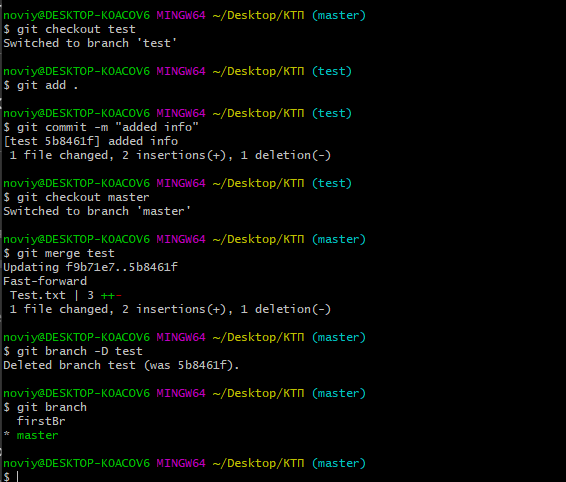


Рисунок 4 – Слияние и удаление ветки

## 5. Настроить удалённый репозиторий: получить изменения из репозитория и отправить изменения.

На GitHub создаём репозиторий, после чего необходимо прописать в Git следующие команды



Рисунок 5 – Репозиторий, в котором будут храниться файлы

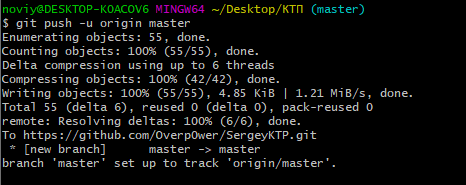


Рисунок 6 – Первоначальная загрузка на удалённый репозиторий

Для получения изменений используем команду git pull, после чего файлы автоматически изменятся

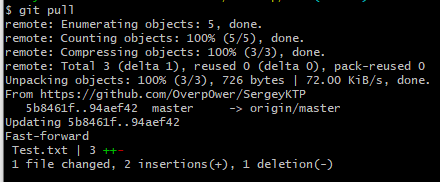


Рисунок 7 – Получение обновлений

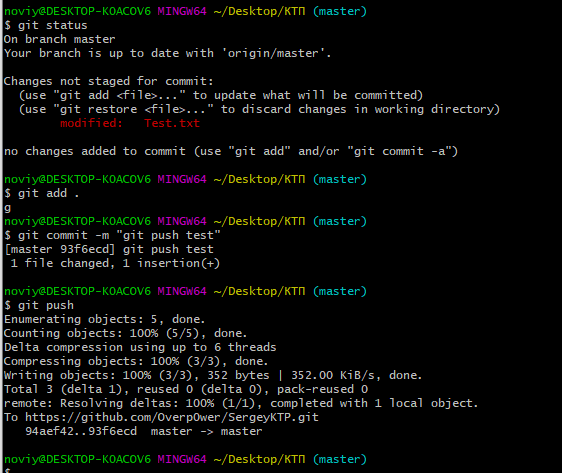


Рисунок 8 – Отправка изменений

## 6. Организовать совместную работу над проектом в соответствии с выбранной идеологией

Для организации совместной работы, необходимо добавить коллегу на Github. Делается это в разделе Settings, Collaborators. Далее нажимаем Add people и вписываем имя пользователя Git, после этого нажимаем add <имя пользователя> to this repository. Следующим шагом коллеге необходимо подтвердить приглашение, которое пришло на почту. Подготовительная часть закончена. Далее коллега должен скачать наш репозиторий. Делается это командной git clone <ссылка на репозиторий>. Далее необходимо прописать права доступа, делается это командой git remote add <краткое название репозитория> <ссылка на репозиторий>. Все, настройка закончена. Чтобы получить актуальную версию файлов, нужно написать git pull <название репозитория>. Чтобы загрузить последний коммит, необходимо прописать git push <название репозитория>.

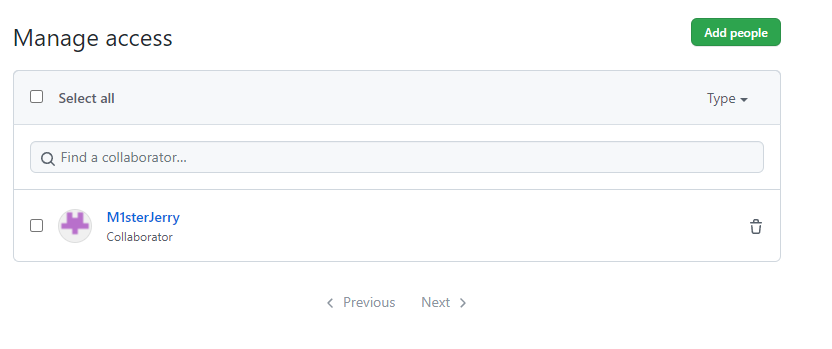


Рисунок 9 – Добавленный коллега в проект

# Вывод

Выполнив данные задачи были изучены подходы к командной разработке в системе контроля версий git

# Список использованных источников:

1. Камаев В.А., Костерин В.В. Технологии программирования. М.: Высшая школа,

2006.

1. Жоголев Е.А.Технология программирования. – М.: Научный мир, 2004.

Git:

1. Scott Chacon, Ben Straub «Pro Git»
2. [git-scm.com](https://git-scm.com/)