Министерство образования Российской Федерации  
Пензенский Государственный Университет  
Кафедра САПР

**ОТЧЁТ**о выполнении учебной (эксплуатационной) практики  
на тему: «Основы работы с системой контроля версий Git на примере разработки алгоритма нахождения кратчайшего пути в графе»

Выполнил: Алешин К.А.  
Группа: 23ВВВ4  
Принял: Гудков П.А.

Пенза 2025 г.

ОГЛАВЛЕНИЕ

[Оглавление 2](#_Toc76062557)

[Аннотация 3](#_Toc76062558)

[Работа с ситемой контроля версий 4](#_Toc76062559)

[Создание репозитория 4](#_Toc76062560)

[Добавление файлов 4](#_Toc76062561)

[Удаление файлов 6](#_Toc76062562)

[Перемещение файлов 7](#_Toc76062563)

[Игнорирование файлов 7](#_Toc76062564)

[Удалённый репозиторий 8](#_Toc76062565)

[Разработка алгоритма 11](#_Toc76062566)

[Вывод 19](#_Toc76062567)

АННОТАЦИЯ

В данной пояснительной записке приведено описание работы с системой контроля версий git на примере разработки алгоритма поиска кратчайшего пути в графе. Для работы с системой контроля версий использовался Git Bash. В данной записке продемонстрированы и описаны основные команды Git. Каждый этап проиллюстрирован скриншотами.

РАБОТА С СИСТЕМОЙ КОНТРОЛЯ ВЕРСИЙ

Создание репозитория

Git репозитория инициализируется в папке с проектом:

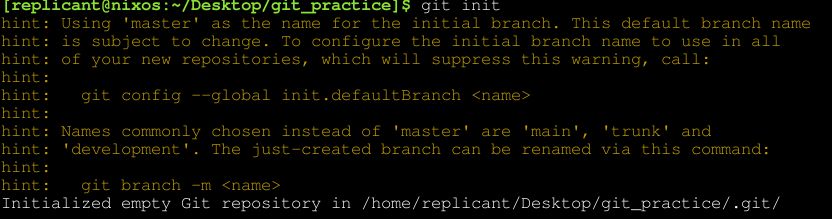


Рисунок 1 — Инициализация репозитория

Настраивается пользователь Git на локальном компьютере: вводится имя пользователя и адрес электронной почты:

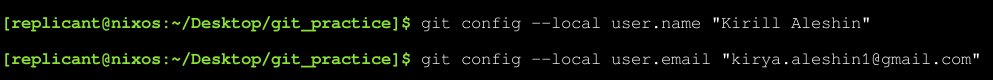


Рисунок 2 — Настройка пользователя

Добавление файлов

В папке с проектом создаётся файл «alg\_dejkstra.c». Изначально он пустой.

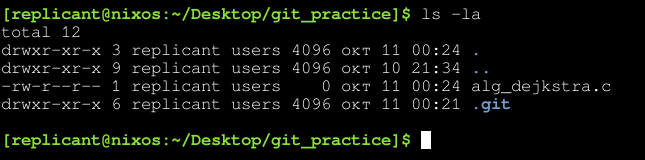


Рисунок 3 — Созданные файлы

Команда git status позволяет определить в каком состояние какие файлы находятся. В данном случае найдены новые файлы, но они ещё не отслеживаются.

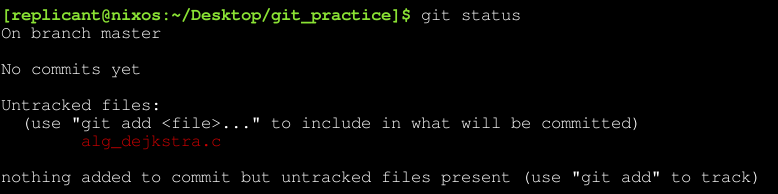


Рисунок 4 — Команда git status

Для отслеживания файлов и последующего сохранении в репозитории (commit) используется команда git add.

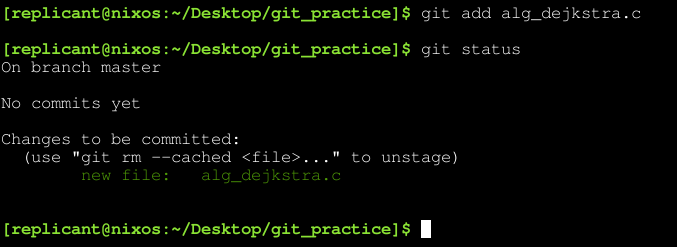


Рисунок 5 — Команда git add

Теперь можно сохранить текущую версию проекта в репозитории (commit). Для этого используется команда git commit. Метка -m позволяет описать сохраняемую версию.

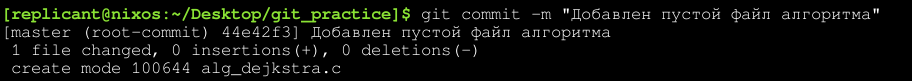


Рисунок 6 — Команда git commit

Для разработки алгоритма необходимы вспомогательные файлы.

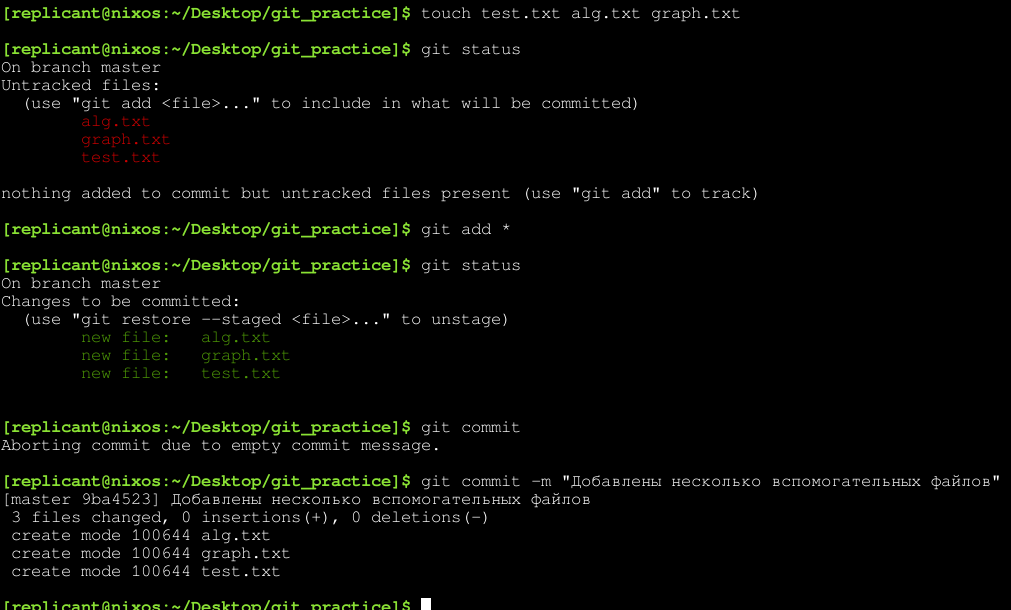


Рисунок 7 — Добавление папки в репозиторий

Удаление файлов

Удаление файлов производится с помощью команды git rm. Удалить можно из индекса (отслеживания), когда командой git add был добавлен лишний файл, или из директории.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 8 — Удаление файла

Перемещение файлов

Для перемещения файлов используется команда git mv. Также эта команда удобна для переименования файлов.

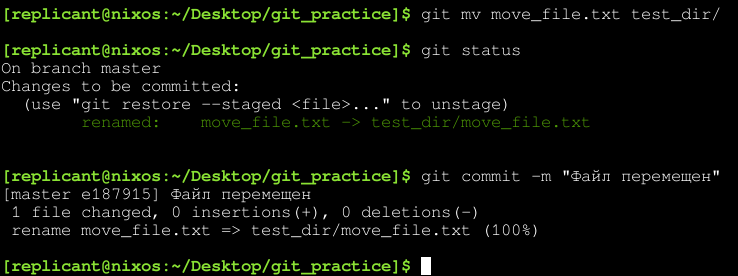


Рисунок 9 — Перемещение и переименование файла

В данном примере файл «move\_file.txt» одновременно переносится в папку test\_dir и меняет название на «test\_dir/move\_file.txt».

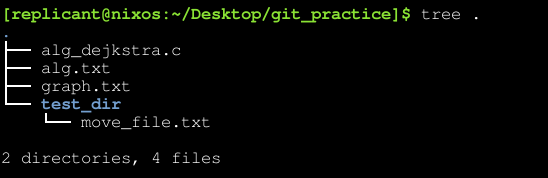


Рисунок 10 — Перемещённый и переименованный файл

Игнорирование файлов

Иногда бывает необходимо не сохранять некоторые файлы в систему контроля версий. Например, временные файлы среды разработки. Чтобы игнорировать какие-то файлы, необходимо создать файл «.gitignore», в котором указывается, что должно быть проигнорировано. В процессе разработки алгоритма не было необходимости в данной функции, поэтому для демонстрации работы был создан отдельный файл.

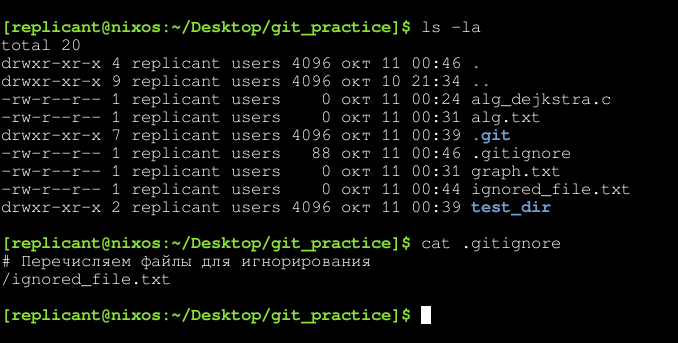


Рисунок 11 — Файл .gitignore и игнорируемый файл

Как только был добавлен файл «.gitignore», команда git status перестала отображать игнорируемый файл.

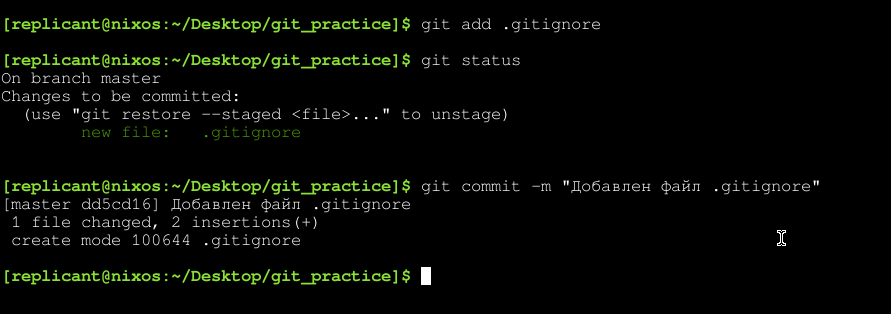


Рисунок 12 — Демонстрация механики игнорирования

Удалённый репозиторий

Для загрузки файлов с локального репозитория на удалённый сначала необходимо выбрать удалённый репозиторий с помощью команды git remote add, указав ссылку на необходимый репозиторий.

С помощью команды git remote -v можно просмотреть список подключённых репозиторий, убедившись, что предыдущая операция выполнена успешно.

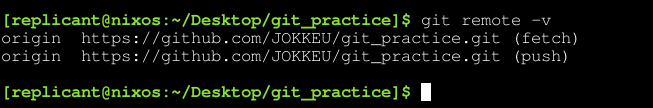


Рисунок 13 — Список удалённых репозиториев

Файлы из локального репозитория загружаются на удалённый с помощью команды git push.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 14 — Загрузка локального рекпозитория на удалённый

Загрузка на удалённый репозиторий github.com прошла успешно.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 15 — Удалённый репозиторий

Также можно скопировать файлы обратно в папку с проектом или в другое место. Для этого используется команда git clone.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 16 — команда git clone

Папка с проектом git\_practice из репозитория на github.com была скопирована на компьютер.



Рисунок 17 — Клонированный репозиторий

Если на удалённом репозитории находится более новая версия проекта, чем на локальном, можно «обновить» проект на локальном. Эта операция представляет собой загрузку файлов и объединение с текущим проектом. Вызывается командой git pull.

Для демонстрации этой функции был создан новый commit, файлы были загружены на удалённый репозиторий, а после на локальном этот commit был отменён.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 18 — Новый commit

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 19 — Загрузка нового commit`а на удалённый репозиторий

С помощью команды git pull удалось восстановить удалённый commit на локальном репозитории.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 20 — Команда git pull

РАЗРАБОТКА АЛГОРИТМА

Для примера был выбран алгоритм Декстры. В процессе разработки использовался язык программирования Си.

Сайт разрабатывался в несколько этапов, каждый из которых был занесён в систему контроля версий git.

Первым шагом были инициализированны переменные и выделена память под массивы.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 21 — Этап 1. иннициализация

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 22 — Этап 1. Commit

Далее были добавлены функции чтения графа из файла и его генерация случайного взвешанного графа.

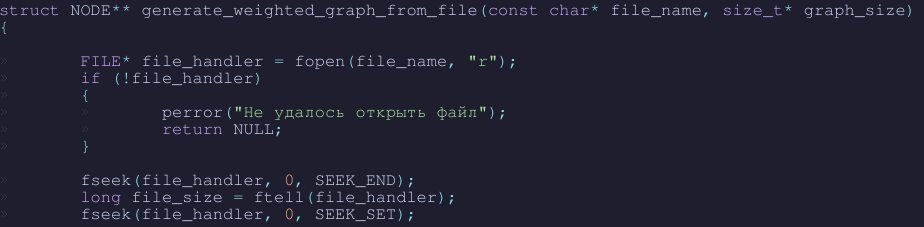
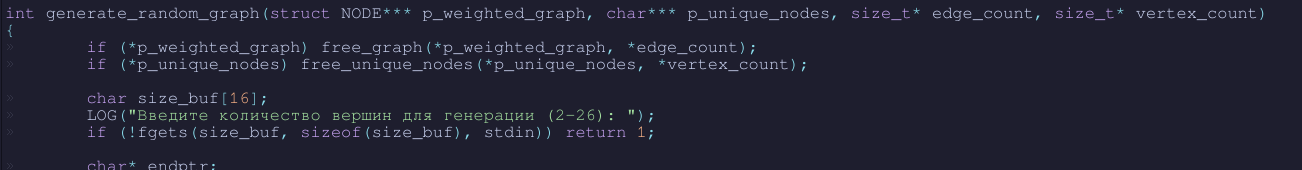


Рисунок 23 — Этап 2. Основные функции генерации графа

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 24 — Этап 2. Commit

Закончена работа над алгоримом Декстры. Слегка оптимизирован код, написанный на прошлых этапах.

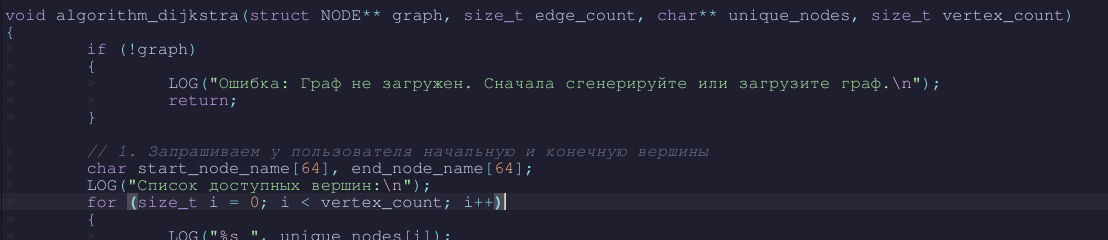


Рисунок 25 — Этап 3. Алгоритм Декстры

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 26 — Этап 3. Commit

Результат работы алгоритма Декстры

Изображение выглядит как млекопитающее, земля, большая кошка, внешний

Автоматически созданное описание

Рисунок 27 — Этап 4. работа алгоритма

В процессе разработки данного сайта были созданы 10 commit`ов разного объёма.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 28 — история commit`ов

**Вывод**

В процессе выполнения данной практической работы был разработан алгоритм Дейкстры на языке программирования Си, этапы создания которого были зафиксированы в системе контроля версий git.

Мной был освоен общий принцип работы git и основные команды для взаимодействия с ним.