git add

Переносит изменения из рабочего каталога в раздел проиндексированных файлов. В этом разделе можно подготовить снимок состояния, прежде чем сделать коммит изменений в официальную историю.

git branch

Эта команда выступает универсальным инструментом администрирования веток. С ее помощью можно создавать изолированные среды разработки в одном репозитории.

### Git checkout

С командой git checkout можно не только получать старые коммиты и прежние версии файлов, но и осуществлять навигацию по существующим веткам. В сочетании с базовыми командами Git она позволяет сосредоточиться на определенном направлении разработки.

### git clean

Удаляет неотслеживаемые файлы из рабочего каталога. Это логический аналог команды git reset, которая (обычно) работает только с отслеживаемыми файлами.

### git clone

Создает копию существующего репозитория Git. Клонирование — самый распространенный способ, с помощью которого разработчики могут получить рабочую копию центрального репозитория.

### git commit

Получает проиндексированный снимок состояния и выполняет его коммит в историю проекта. Эта команда в сочетании с командой git add определяет классический рабочий процесс для всех пользователей Git.

### git commit --amend

Команда git commit с флагом --amend позволяет внести изменения в последний коммит. Она может оказаться полезной, если вы забыли проиндексировать файл или не указали важную информацию в комментарии к коммиту.

### git config

Удобный способ для настройки параметров конфигурации в инсталляции Git. Обычно эту команду используют сразу после установки Git на новую машину разработчика.

### git fetch

С помощью команды извлечения можно загрузить ветку из другого репозитория вместе со всеми связанными с ней коммитами и файлами, однако при этом изменения не будут интегрированы в локальный репозиторий. Благодаря этому вы сможете проверить изменения перед слиянием с проектом.

### git init

Инициализирует новый репозиторий Git. Если вы хотите использовать в проекте контроль версий, эту команду следует изучить раньше остальных.

### git log

Позволяет изучить предыдущие версии проекта. Эта команда имеет несколько параметров форматирования, с помощью которых можно отобразить снимки состояния после выполнения коммитов.

### Git merge

Эффективный способ интеграции изменений из разошедшихся веток. После разветвления истории проекта командой git branch можно использовать команду git merge, чтобы объединить отдельные ветки.

### git pull

Команда git pull — это автоматизированная версия команды git fetch. Она загружает ветку из удаленного репозитория и сразу же объединяет ее с текущей веткой. Эта команда представляет собой git-эквивалент команды svn update.

### git push

Команда git push противоположна команде извлечения (с некоторыми оговорками). С ее помощью можно перенести локальную ветку в другой репозиторий и без труда опубликовать поступивший код. Эта команда похожа на svn commit с тем исключением, что она отправляет не один набор изменений, а серию коммитов.

### git rebase

С помощью команды перебазирования можно переместить ветки и избежать ненужных коммитов слияния. Полученную линейную историю зачастую намного легче понять и изучить.

### git rebase -i

С помощью флага -i можно запустить перебазирование в интерактивном режиме. При этом сохраняются все преимущества обычного перебазирования и появляется возможность добавлять, редактировать или удалять коммиты по ходу операции.

### git reflog

Git отслеживает изменения в конце веток с помощью механизма журналов ссылок (reflog). Он позволяет вернуться к наборам изменений, даже если на них не ссылается никакая ветка или тег.

### git remote

Удобный инструмент для администрирования удаленных подключений. С его помощью вместо полного URL-адреса в командах fetch, pull и push можно использовать более удобное сокращение.

### git reset

Отменяет изменения в файлах в рабочем каталоге. Эта команда сброса позволяет очистить или полностью удалить изменения, которые не были отправлены в публичный репозиторий.

### git revert

Отменяет коммит снимка состояния. Если вы обнаружили ошибочный коммит, его можно легко и безопасно удалить из базы кода с помощью команды git revert.

### git status

Показывает состояние рабочего каталога и проиндексированного снимка состояния. Эту команду можно выполнять в сочетании с git add и git commit, чтобы узнать, что именно будет включено в следующий снимок.

### git branch -M main

Переименовывает ветку в main и делает ее главной и ли что-то типа того, хз короч