**Командная строка:**

**dir** – информация о директории

1. **dir /a**  (Опция /a покажет все файлы, включая скрытые, системные и другие типы файлов.)
2. **dir /q** (Покажет владельца каждого файла в дополнение к стандартной информации (размер, дата последнего изменения).)
3. **dir /t:W**  (Сортирует файлы по времени их последнего изменения, где W — это сортировка по времени изменения файла.)
4. **dir /s**  (Эта команда выведет все файлы и папки, включая файлы в подкаталогах.)
5. **dir /ad**  (Показать список только с папками)
6. **dir /a-d**  (Показать список только с файлами)

**/p** — выводит результат постранично (если в директории много файлов).

**/w** — выводит список в виде широкой таблицы, где файлы и папки разделены пробелами, а не выводятся в столбик.

**mkdir** - создания новых директорий (папок):

1. **mkdir folder**  (создаст папку с именем new\_folder в текущей рабочей папке)
2. **mkdir folder1** folder2 folder3 (Создаст три папки: folder1, folder2 и folder3)
3. **mkdir -p parent/child/grandchild** (создаёт вложенные папки)
4. **mkdir** **-m 755 new\_folder**

**-m** — задает права доступа к создаваемой директории.

**-v** — выводит сообщение о создании каждой директории (verbose).

**-p** — позволяет создавать промежуточные директории, если их нет.  
В некоторых случаях, если директория с таким именем уже существует, команда mkdir вернет ошибку. Чтобы избежать этой ошибки, используйте опцию **-p**.

Вывести фразу в консоль: **echo Привет, мир!  
echo Привет, мир! > example.txt** (Эта команда создаст файл example.txt в текущей директории и запишет в него строку "Привет, мир!"). Если хотим добавить текст в файл (а не перезаписать его), используем двойной знак **>>  
  
del example.txt** (Эта команда удалит файл)

**rd folder** (Эта команда удалит папку)  
  
**cd ../** (cd – change direction. Перемещение между папками)

**type example.txt** (type – команда для просмотра содержимого файла)

**type nul > example.txt** (создаст пустой файл)

**GIT:**

**git --version** (смотрим версию гита)

Настройка гита перед работой. Авторизация:

**git config –global user.name “murad ibn Sabir”**

**git config –global user.email “murad@mail.ru”**

**git config --list** (смотрим список всех настроек)

**git config --list –show-origin** (смотрим список всех настроек + местонахождение конфигурационного файла .gitconfig)  
  
**Есть 2 варианта создания гита в папке:**

1. Инициализировать гит в папке с нуля: **git init**
2. Клонировать гит **git clone https://github.com/username/repository.git**

**Добавление файлов в гит:**1. **git add .** (добавление всех файлов)

**git add example.txt** (добавление определённого файла)

2. **git commit -m ‘first commit’**

Если написать просто **git commit,** то консоль выдаст «please enter the commit…»  
**В таком случае:**

1. Нажимаем **ESC** + **i**
2. Вводим комментарий
3. Нажимаем **ESC** + **:**
4. Пишем **:wq** и нажимаем Enter

**git status** (статус проекта. Если изменения и т.д.)

**git status --cached** (статус проекта с ещё не коммитироваными файлами)

**git commit –am “changes name”** (эта команда эквивалента **«гит адд + гит коммит»**)

**Удаление файлов:**1. **unlink fileName.txt** (Команда консоли. Ей можно удалить ещё не индексировнные файлы)

2. **git checkout -- example.txt** (**отмена изменений файла** после коммита)

3. **git reset example.txt** (перевод в неиндексированное состояние. Отмена «**гит адд**»)

4. **git rm example.txt** (**удалит файл как из** индекса, так и из рабочего каталога. Т.е. полностью)

**git log** (смотрим историю коммитов в хронологическом порядке)  
**git log -2** (вывдим только два коммита)

**git log -p** (покажет также и детальные изменения в каждом коммите)

**git log --stat** (покажет количества удалений, ихменений и добавлений)

**git log --pretty=oneline** (покажет коммиты в одну строку)

**git log --until=1.week** (искать коммиты заканчивая до 1 недели назад)

**git log -- since =2.week** (искать коммиты начиная с 2 недели назад)

**git log --since =2024-12-29 -until=2025-01-29**

**клавиша Q** – выход из режима простотра

**git blame example.txt** (смотрим историю изменения в определенном документе)

**.gitignore:**

**.log** (игнорировать файлы с расширением **.log**)

**tmp/\*** (игнорировать файлы в папке **tmp**)

**Управление ветками:**

**git branch** (выведет список всех веток в репозитории, помечая активную ветку звездочкой **\***)

**git branch <имя-ветки>** (создаём новую ветку)

**git branch -d <имя-ветки>** (удаляем ветку) Это работает случае, если она была слита с другой

**git branch -D <имя-ветки>** (удаляем ветку принудительно. Без проверки на слияние)

**git branch -m <старое-имя> <новое-имя>** (Переименовать ветку)

Переключиься на существующую ветку:

**git checkout <имя-ветки>** (Переключаемся на указанную ветку)

**git switch <имя-ветки>** (Переключаемся на указанную ветку. **На новых версиях Гита**)

Создать ветку и сразу же переключиться на неё:

**git checkout -b <имя-новой-ветки>**

**git switch -c <имя-ветки>** (На новых версиях Гита)

**git --version** (смотрим версию гита)

**git --version** (смотрим версию гита)

**git --version** (смотрим версию гита)

**Слияние веток:**

Для слияния ветки «myBranch» в ветку «master», мы должны находиться на ветке «master» и написать команду:  
**git merge <имя-ветки>** (в нашем случае, <имя ветки> = myBranch)