Лабораторная работа №7. Знакомство с Git и его использование на практике.

Задание: Привести 15 примеров команд системы контроля версий Git.

**1) git status**

Команда показывает состояния файлов в рабочей директории и индексе: какие файлы изменены, но не добавлены в индекс; какие ожидают коммита в индексе. Вдобавок к этому выводятся подсказки о том, как изменить состояние файлов.

**2) git add**

Команда добавляет содержимое рабочей директории в индекс (staging area) для последующего коммита. По умолчанию git commit использует лишь этот индекс, так что вы можете использовать git add для сборки слепка вашего следующего коммита.

**3) git diff**

Команда используется для вычисления разницы между любыми двумя Git деревьями. Это может быть разница между вашей рабочей директорией и индексом (собственно git diff), разница между индексом и последним коммитом (git diff --staged), или между любыми двумя коммитами (git diff master branchB).

**4) git difftool**

Команда просто запускает внешнюю утилиту сравнения для показа различий в двух деревьях, на случай если вы хотите использовать что-либо отличное от встроенного просмотрщика git diff.

**5) git reset**

Команда как можно догадаться из названия, используется в основном для отмены изменений. Она изменяет указатель HEAD и, опционально, состояние индекса.

**6) git rm**

Команда используется в Git для удаления файлов из индекса и рабочей директории. Она похожа на git add с тем лишь исключением, что она удаляет, а не добавляет файлы для следующего коммита.

**7) git commit**

Команда берёт все данные, добавленные в индекс с помощью git add, и сохраняет их слепок во внутренней базе данных, а затем сдвигает указатель текущей ветки на этот слепок.

**8) git mv**

Данная команда это всего лишь удобный способ переместить файл, а затем выполнить git add для нового файла и git rm для старого.

**9) git clean**

Команда используется для удаления мусора из рабочей директории. Это могут быть результаты сборки проекта или файлы конфликтов слияний.

**10) git branch**

Данная команда своего рода “менеджер веток”. Она умеет перечислять ваши ветки, создавать новые, удалять и переименовывать их.

**11) git checkout**

Команда используется для переключения веток и выгрузки их содержимого в рабочую директорию.

**12) git pull**

Команда работает как комбинация команд git fetch и git merge, т.е. Git вначале забирает изменения из указанного удалённого репозитория, а затем пытается слить их с текущей веткой.

**13) git archive**

Команда используется для упаковки в архив указанных коммитов или всего репозитория.

**14) git show**

Команда отображает объект в простом и человекопонятном виде. Обычно она используется для просмотра информации о метке или коммите.

**15) git bisect**

Ну а данная команда это чрезвычайно полезная утилита для поиска коммита в котором впервые проявился баг или проблема с помощью автоматического бинарного поиска.