







Ответы на вопросы.

1. Что такое СКВ и каково ее назначение?  
Система контроля версий (СКВ) — это система, регистрирующая изменения в одном или нескольких файлах с тем, чтобы в дальнейшем была возможность вернуться к определённым старым версиям этих файлов.

2. В чем недостатки локальных и централизованных СКВ?  
В локальном СКВ можно легко забыть, в какой директории вы находитесь, и случайно изменить не тот файл или скопировать не те файлы, которые вы хотели. В централизованных СКВ если сервер вышел из строя, то можно потерять всё – всю историю проекта, не считая единичных снимков репозитория, которые сохранились на локальных машинах разработчиков.

3. К какой СКВ относится Git?  
Распределённые системы контроля версий (РСКВ)

4. В чем концептуальное отличие Git от других СКВ?  
Концептуально, большинство других систем хранят информацию в виде списка изменений в файлах.

5. Как обеспечивается целостность хранимых данных в Git?  
Git представляет свои данные как, поток снимков.

6. В каких состояниях могут находится файлы в Git? Как связаны эти состояния?  
Git имеет три основных состояния, в которых могут находиться файлы: изменённые, индексированные и зафиксированные.

* Изменённый означает, что вы изменили файл, но ещё не зафиксировали его в своем локальном репозитории.
* Индексированный -это изменённый файл, текущую версию которого вы отметили для включения в следующий коммит (для фиксации в своём локальном репозитории).
* Зафиксированный означает, что файл уже сохранён в вашем локальном репозитории.

7. Что такое профиль пользователя в GitHub?  
Профиль - это наша публичная страница на GitHub, как и в социальных сетях. Когда мы ищем работу в качестве программиста, работодатели могут посмотреть наш профиль GitHub и принять его во внимание, когда будут решать, брать на работу или нет.

8. Какие бывают репозитории в GitHub?  
Приватные и публичные

9. Укажите основные этапы модели работы с GitHub.

* Создание repository
* Клонирование repository
* Изменение содержимого
* Сохранение изменений
* Распространение изменений в исходный repository

10. Как осуществляется первоначальная настройка Git после установки?

Нужно настроить имя пользователя и email для идентификации.

11. Опишите этапы создания репозитория в GitHub.

* Нужно дать название repository
* Не обязательно написать описание
* Нужно выбрать каким будет repository публичный или приватный
* Добавить файл README
* Добавить .gitignore
* Добавить лицензию

12. Какие типы лицензий поддерживаются GitHub при создании репозитория?

* Apache
* GPL
* MIT
* ISC
* GNU
* BSD
* И т.д.

13. Как осуществляется клонирование репозитория GitHub? Зачем нужно клонировать

репозиторий?

При помощи команды git clone  
Плюсы удаленного репозитория

* выполняет роль резервной копии
* возможность работать в команде
* некоторые дополнительные возможности, которые предоставляет хостинг. Например, визуализация истории или возможность работать над проектом прямо в веб-интерфейсе

14. Как проверить состояние локального репозитория Git?

При помощи команды git status

15. Как изменяется состояние локального репозитория Git после выполнения следующих

операций: добавления/изменения файла в локальный репозиторий Git; добавления нового/

измененного файла под версионный контроль с помощью команды git add; фиксации

(коммита) изменений с помощью команды git commit и отправки изменений на сервер с

помощью команды git push?

Все изменения фиксируются в виде «снимок».

16. У Вас имеется репозиторий на GitHub и два рабочих компьютера, с помощью которых Вы

можете осуществлять работу над некоторым проектом с использованием этого репозитория.

Опишите последовательность команд, с помощью которых оба локальных репозитория,

связанных с репозиторием GitHub будут находиться в синхронизированном состоянии.

Примечание: описание необходимо начать с команды git clone.

Чтобы внести изменения, сделанные кем-то другим, нужно использовать git pull

Для того чтобы изменения выложить на удаленный репозиторий используется команда git push

17. GitHub является не единственным сервисом, работающим с Git. Какие сервисы еще Вам

известны? Приведите сравнительный анализ одного из таких сервисов с GitHub.

Sourceforge, Google Code, GitLab, Codebase и т.д. GitHub делает упор на высокую доступность и производительность своей инфраструктуры и делегирует другие сложные функции сторонним инструментам, а GitLab, наоборот, фокусируется на включении всех функций на одной проверенной и хорошо интегрированной платформе.

18. Интерфейс командной строки является не единственным и далеко не самым удобным

способом работы с Git. Какие Вам известны программные средства с графическим

интерфейсом пользователя для работы с Git? Приведите как реализуются описанные в

лабораторной работе операции Git с помощью одного из таких программных средств.

GitHub Desktop, Fork, Tower, Sourcetree, SmartGit