1. Для решения каких задач предназначаются СКВ?

CVS (Concurrent Versions System) - это система контроля версий файлов. Основное предназначение - сохранение истории изменения файлов.

Необходимость в системе контроля версий появляется например тогда, когда в достаточно большом проекте некотороая ошибка проявляестя только через некоторое время после внесения изменений. С помощью CVS можно очень быстро и просто получить нужную версию файла и увидеть, что конкретно вызвало появление ошибки.

Превосходство CVS против обычного резервного копирования файлов после каждого изменения в том, что CVS сохраняет все версии в одном файле, добавляя только данные о внесенных изменениях.

Также CVS помогает при работе нескольких людей над одним проектом - она не позволяет изменять одновременно одну и ту же чать файла нескольким людям. При этом каждый отдельный разработчик работает только со своей копией файла, что не вызывает ошибок разделения доступа при одновременной работе нескольких разработчиков. CVS сама следит за объединением изменений.

1. Объясните следующие понятия СКВ: хранилище, история, рабочая копия. Примеры.

Хранилище документов (repository, depot) — место, где система управления версиями хранит все документы вместе с историей их изменения и другой служебной информацией.

История (history) — файл, в котором отслеживается каждое употребление команд checkout, commit, rtag, update и release.

Рабочая копия (working copy) — рабочая(локальная) копия документов.

Примером может выступать мой аккаунт на GitHub, в котором есть некоторые репозитории(хранилища) с проектами. Каждый репозиторий хранит историю и имеет рабочую копию документов.

1. Что такое и зачем могут быть нужны ветви (branches)?

Ветвь (branch) — направление разработки, независимое от других. Ветвь представляет собой копию части хранилища (например, одного каталога), в которую можно вносить изменения, не влияющие на другие ветви. Документы в разных ветвях имеют одинаковую историю до точки ветвления и разные — после неё.

Ветви нужны для безопасного внесения правок и изменений в документ, без угрозы нарушить структуру «главной» версии документа.

1. Что такое и зачем может быть нужна разность (diff)? Примеры.

Разность (diff) — команда сравнения файлов, выводящая разницу между двумя файлами.

Разность используется для срванения рабочих файлов с модификациями, на которых они основаны, и сообщения о любых найденных различиях.

При попытке отправить новую версию кода в свой репозиторий на GitHub, сначала показывается окно, в котором указана разница между версиями отправляемого файла и существующего в репозитории.

1. Что такое и зачем может быть нужно слияние (merge)?

Слияние (merge, integration) — объединение независимых изменений в единую версию документа. Осуществляется, когда два человека изменили один и тот же файл или при переносе изменений из одной ветки в другую.

Merge можно использовать не только для слияния. Фактически команда производит внесение в рабочую копию изменений, равных разнице между двумя директориями или файлами в хранилище, поэтому merge является универсальным средством для переноса изменений. Можно привести такие примеры использования команды merge:

* откат уже зафиксированных изменений, в том числе целого диапазона ревизий;
* удобный просмотр (в виде изменений в рабочей копии) разницы между двумя состояниями репозитория.

1. Объясните смысл действия commit.

Команда commit используется тогда, когда вы хотите внести в репозиторий сделанные в рабочей копии изменения и сделать их доступными для остальных разработчиков. Если вы не указываете конкретный файл для фиксации, то просматриваются все файлы в текущем каталоге и фиксируются, если они были изменены. (По умолчанию процесс присходит рекурсивно, т.е. также просматриваются и все подкаталоги. Если надо ограничить область просмотра, используется опция -l).

Если какой-либо файл имеет более старую версию, чем в репозитории, но был изменен, то фиксация не выполняется, а выводится сообщение Up-to-date check failed. Это означает, что сначала необходимо обновить рабочую копию при помощи команды update.

1. Объясните смысл действия rebase в СКВ Git.

Эта операция есть получение набора всех изменений в ветви **А**, с последующим их «накатом» на ветвь **B**. В результате ветвь **B** продвигается до состояния **AB**. В отличие от слияния, в истории ветви **AB** не останется никаких промежуточных коммитов ветви **A** (только история ветви **B** и запись о самом rebase, это упрощает интеграцию крупных и очень крупных проектов).