МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

ФИЛИАЛ УЧРЕЖДЕНИЯ ОБРАЗОВАНИЯ

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

«БОБРУЙСКИЙ ГОСУДАРСТВЕННЫЙ ЛЕСОТЕХНИЧЕСКИЙ КОЛЛЕДЖ»

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 1**

**СОЗДАНИЕ ТАБЛИЦ И ЗАПРОСОВ В БД**

**Учебная дисциплина**  КиАПС

**Специальность** 2-40 01 01 Программное обеспечение информационных технологий

Подготовили уч-ся группы ПО-1

Гордиенко Алексей и Александров Матвей

Бобруйск 2020

**Лабораторная работа №1**

**Тема:** Разработка программ с использованием системы управления версии.

**Цели:** Закрепление навыков работы по созданию таблиц и запросов.

**Технологическое оснащение работы:**

– Windows 10;

– Visual Studio 2017;

– Microsoft Word 2016.

**Выполнение работы**

1.Что такое системы контроля версий (VCS) и для решения каких задач они предназначаются?

**Ответ на вопрос**: Системы контроля версий (Version Control System, VCS) применяются при работе нескольких человек над одним проектом. Обычно основное дерево проекта хранится в локальном или удалённом репозитории, к которому настроен доступ для участников проекта.

2. Объясните следующие понятия VCS и их отношения: хранилище, commit, история, рабочая копия.

**Ответ на вопрос**:

**Хранилище** (repository), или репозиторий, — место хранения всех версий и служебной информации.

**Версия** (revision), или ревизия, — состояние всех файлов на определенный момент времени, сохраненное в репозитории, с дополнительной информацией

**Commit** («вклад») — синоним версии; процесс создания новой версии.

**История** – Список всех версий программы на этапе ее создания

**Рабочая копия** (working copy) — текущее состояние файлов проекта, основанное на версии, загруженной из хранилища (обычно на последней).

3. Что представляют собой и чем отличаются централизованные и децентрализованные VCS? Приведите примеры VCS каждого вида.

**Ответ на вопрос**:

**Централизованные:**

Простота использования. Вся история — всегда в едином общем хранилище. Нужно подключение к сети. Резервное копирование нужно только одному хранилищу. Удобство разделения прав доступа к хранилищу. Почти все изменения навсегда попадают в общее хранилище.

**Распределенные:**

Двухфазный commit: 1) запись в локальную историю; 2) пересылка изменений другим. Подключение к сети не нужно. Локальные хранилища могут служить резервными копиями. Локальное хранилище контролирует его владелец, но общее — администратор. Возможна правка локальной истории перед отправкой на сервер.

4. Опишите действия с VCS при единоличной работе с хранилищем.

**Ответ на вопрос**: Ответ на вопрос представлен в виде схемы на рисунке 1

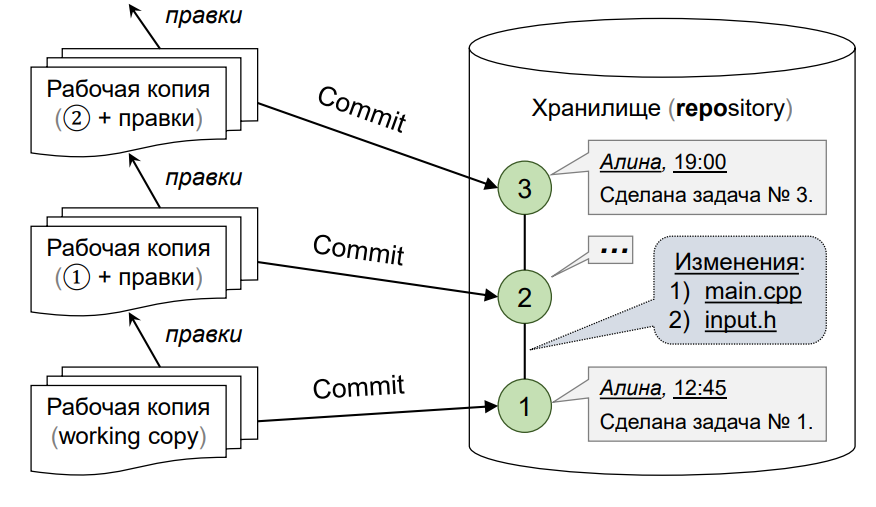


Рисунок 1 – VCS при единоличной работе

5. Опишите порядок работы с общим хранилищем VCS.

**Ответ на вопрос**: Ответ на вопрос представлен в виде схемы на рисунке 2

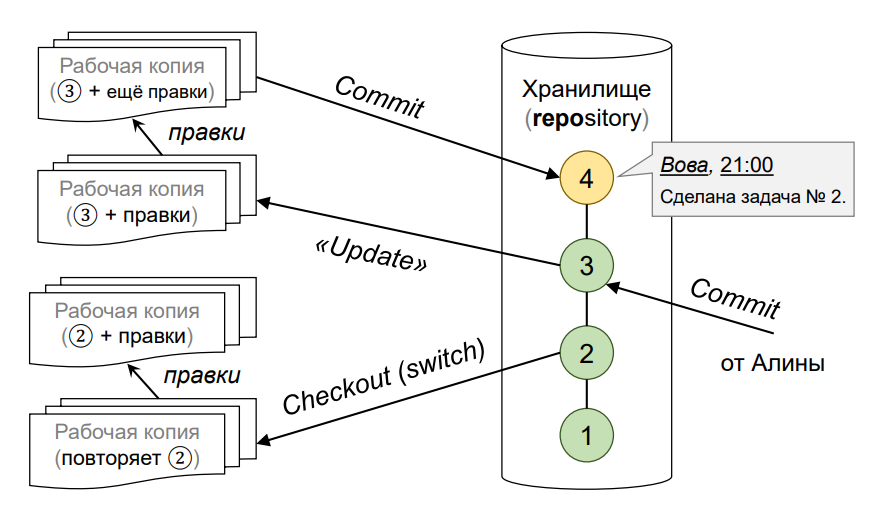


Рисунок 2 – Работа с общим хранилищем VCS

6. Краткая история Git.

**Ответ на вопрос:** В 2005 году отношения между сообществом разработчиков ядра Linux и коммерческой компанией, которая разрабатывала BitKeeper, прекратились, и бесплатное использование утилиты стало невозможным. Это сподвигло сообщество разработчиков ядра Linux (а в частности Линуса Торвальдса — создателя Linux) разработать свою собственную утилиту, учитывая уроки, полученные при работе с BitKeeper. Некоторыми целями, которые преследовала новая система, были:

1. Скорость
2. Простая архитектура
3. Хорошая поддержка нелинейной разработки (тысячи параллельных веток)
4. Полная децентрализация

Возможность эффективного управления большими проектами, такими как ядро Linux (скорость работы и разумное использование дискового пространства)

7. Каковы основные задачи, решаемые инструментальным средством git? **Ответ на вопрос**: Слежка за целостностью данных. Клонирование существующих репозиториев. Запись изменений в репозитории. Индексация изменённых файлов. Откат версий. Просмотр индексированных и не индексированных изменений. Удаление файлов. Перемещение файлов.

8. Назовите и дайте краткую характеристику командам git.

**Ответ на вопрос**:

$ git checkout -b "name"// Имя новой ветки, создать и сразу переключится

Посмотреть ветки

$ git branch (-v доп. инфа и последние комиты)

Просто создание ветки без переключения

$ git branch “namebranch”

Переключение на другую ветку

$ git checkout "namebranch"

9. Приведите примеры использования при работе с локальным и удалённым репозиториями.

**Ответ на вопрос**:

**Просмотр удалённых репозиториев.** Для того, чтобы просмотреть список настроенных удалённых репозиториев, вы можете запустить команду git remote. Она выведет названия доступных удалённых репозиториев. Если вы клонировали репозиторий, то увидите, как минимум origin — имя по умолчанию, которое Git даёт серверу, с которого производилось клонирование.

**Добавление удалённых репозиториев** В предыдущих разделах мы уже упоминали и приводили примеры добавления удалённых репозиториев, сейчас рассмотрим эту операцию подробнее. Для того, чтобы добавить удалённый репозиторий и присвоить ему имя (shortname), просто выполните команду git remote add <shortname> <url>

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

**Ответ на вопрос**:

11. Как и зачем можно игнорировать некоторые файлы при commit?

**Ответ на вопрос**: Игнорируемые файлы – это, как правило, специфичные для платформы файлы или автоматически созданные файлы из систем сборки. Некоторые общие примеры включают в себя:

* Файлы времени выполнения, такие как журнал, блокировка, кэш или временные файлы.
* Файлы с конфиденциальной информацией, такой как пароли или ключи API.
* Скомпилированный код, такой как .class или .o.
* Каталоги зависимостей, такие как /vendor или /node\_modules.
* Создавать папки, такие как /public, /out или /dist.
* Системные файлы, такие как .DS\_Store или Thumbs.db
* Конфигурационные файлы IDE или текстового редактора.

Файл .gitignore представляет собой обычный текстовый файл, в котором каждая строка содержит шаблон для файлов или каталогов игнорировать.

.gitignore использует шаблоны с подстановкой для сопоставления имен файлов с подстановочными знаками. Если у вас есть файлы или каталоги, содержащие шаблон подстановки, вы можете использовать одну обратную косую черту (\) для экранирования символа.