Лабораторная работа №4

Дисциплина: Архитектура компьютера

Кондратьев Арсений Вячеславович

02.10.2022

Содержание

# 1 Цель работы

Составить отчет по 3 лабораторной в формате Markdown

# 2 Выполнение лабораторной работы

1. Сделать предварительную конфигурацию git(рис.1)

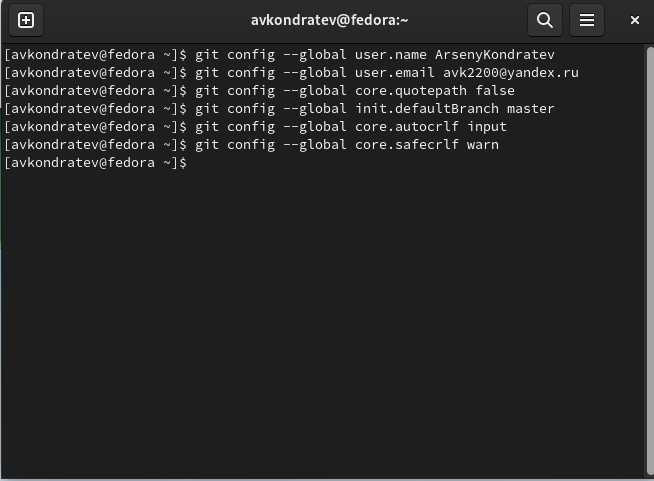


Figure 1: Рис. 1

1. Создание SSH ключа(рис.2)

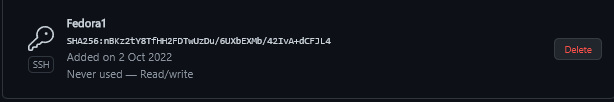


Figure 2: Рис. 2

1. Создать каталог для предмета(рис.3)

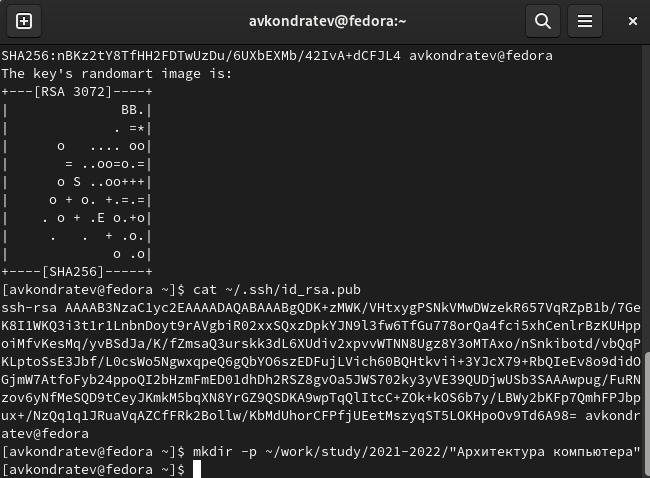


Figure 3: Рис. 3

1. Клонировать созданный репозиторий(рис.4)

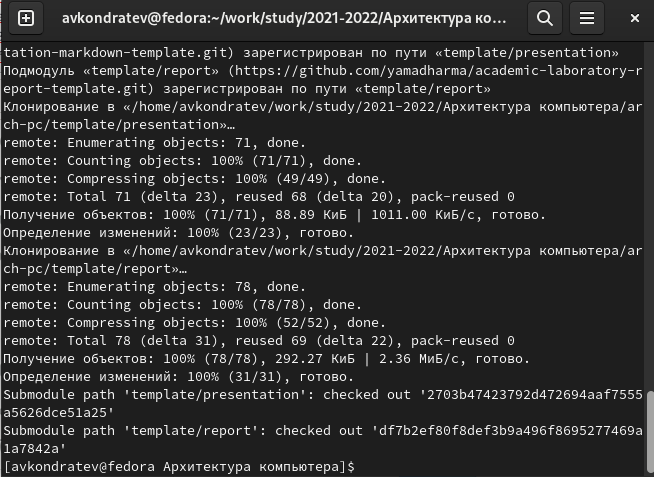


Figure 4: Рис. 4

1. Удалил лишние файлы и создал необходимые каталоги(рис.5)

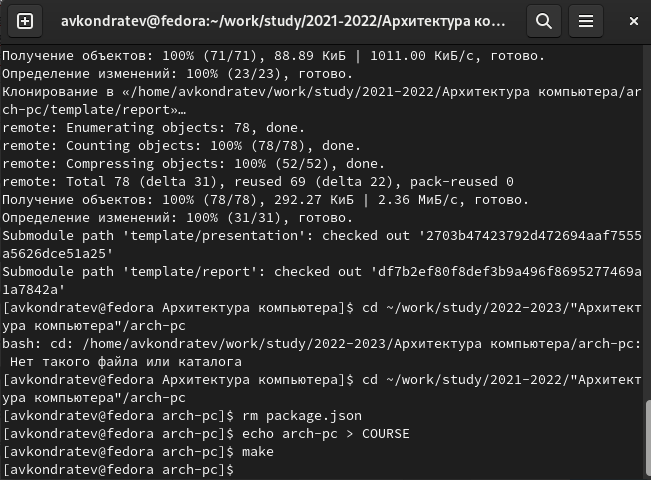


Figure 5: Рис. 5

1. Отправил файлы на сервер(рис.6)

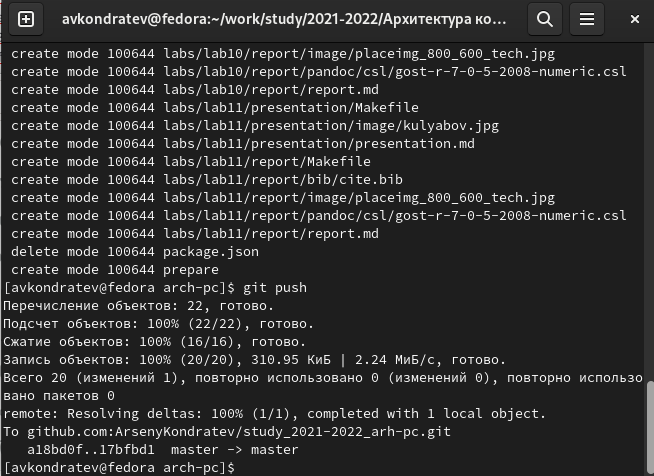


Figure 6: Рис. 6

# 3 Выводы

Я составил отчет по 3 лабораторной в формате Markdown

# 4 Контрольные вопросы

1. Системы контроля версий – это программные инструменты, помогающие командам разработчиков управлять изменениями в исходном коде с течением времени. Применяются при работе нескольких человек над одним проектом
2. Хранилище – место, где хранятся изменения кода. Commit - снимок состояния проекта на текущий момент времени. История – список снимков состояния проекта к которым можно при необходимости откатиться. Рабочая копия - Рабочая копия является снимком одной версии проекта.
3. Централизованные системы контроля версий представляют собой приложения типа клиент-сервер, когда репозиторий проекта существует в единственном экземпляре и хранится на сервере.(CVS, Subversion) Децентрализованные системы контроля версий (Distributed Version Control System, DVCS) позволяют хранить репозиторий (его копию) у каждого разработчика, работающего с данной системой.(Git, Mercurial)
4. Создаем свою ветку, базирующуюся на главной(git checkout -b имя\_ветки), вносим изменения, делаем снимок(git commit) и затем вносим эти изменения в свою ветку(git push)
5. Отдельные ветки разработчиков внедряются в общую master ветку
6. Git позволяет несокльким разработчикам с удобством работать над одним проектом. Возможность получать изменения, внесенные другим человеком и откатываться на прошлые версии в случае ошибок.
7. Создание основного дерева репозитория – git init b. получение обновлений (изменений) текущего дерева из центрального репозитория – git pull c. отправка всех произведённых изменений локального дерева в центральный репозиторий – git push d. просмотр списка изменённых файлов в текущей директории – git status e. просмотр текущих изменений – git diff f. добавить все изменённые и/или созданные файлы и/или каталоги – git add g. сохранить все добавленные изменения и все изменённые файлы – git commit h. создание новой ветки, базирующейся на текущей: - git checkout -b имя\_ветки i. переключение на некоторую ветку - git checkout имя\_ветки j. слияние ветки с текущим деревом - git merge –no-ff имя\_ветки k. удаление локальной уже слитой с основным деревом ветки - git branch -d имя\_ветки l. принудительное удаление локальной ветки - git branch -D имя\_ветки
8. С локальным: commit(снимок состояния проекта) С удаленным: push(отправляем изменения) pull(загружаем изменения)