ОТЧЕТ по лабораторной работе № 3

дисциплина: Архитектура компьютера

Студент: Акопян Сатеник\* Группа:НКАбд-01-22

Содержание

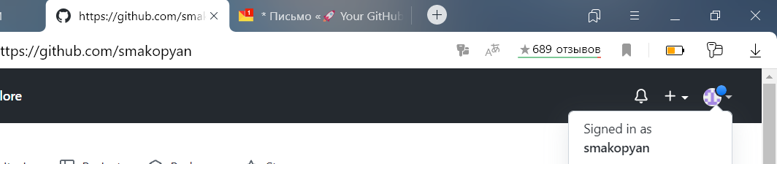
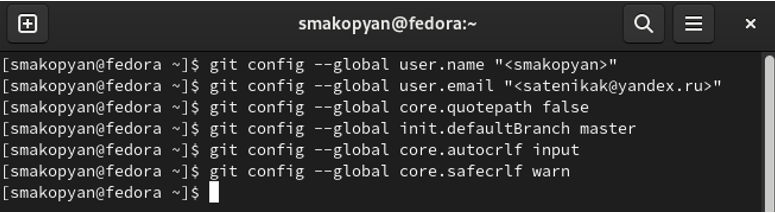
# 1 Цель работы

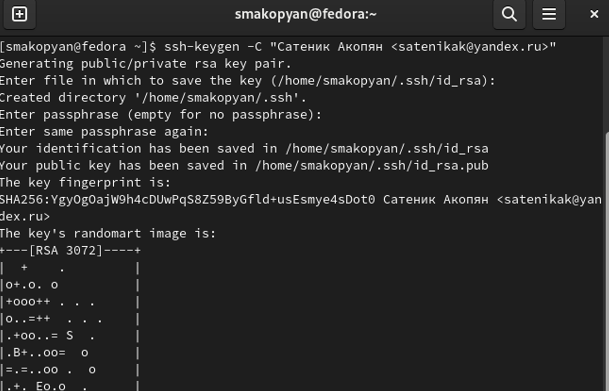
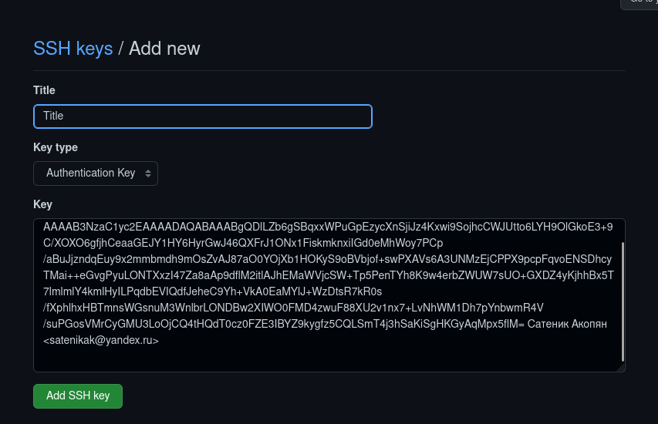
Целью работы является изучить идеологию и применение средств контроля версий. Приобрести практические навыки по работе с системой git.

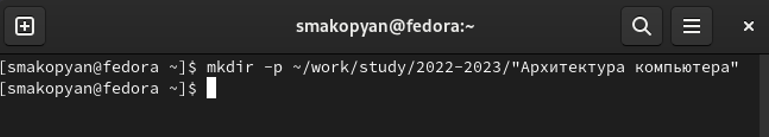
# 2 Теоретическое введение

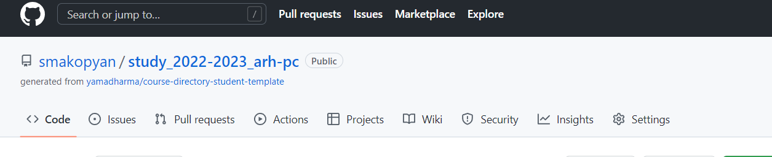
Системы контроля версий (Version Control System, VCS) применяются при работе нескольких человек над одним проектом. Обычно основное дерево проекта хранится в локальном или удалённом репозитории, к которому настроен доступ для участников проекта. При внесении изменений в содержание проекта система контроля версий позволяет их фиксировать, совмещать изменения, произведённые разными участниками проекта, производить откат к любой более ранней версии проекта, если это требуется.В классических системах контроля версий используется централизованная модель, предполагающая наличие единого репозитория для хранения файлов. Выполнение большинства функций по управлению версиями осуществляется специальным сервером.Участник проекта (пользователь) перед началом работы посредством определённых команд получает нужную ему версию файлов. После внесения изменений, пользователь размещает новую версию в хранилище. При этом предыдущие версии не удаляются из центрального хранилища и к ним можно вернуться в любой момент. Сервер может сохранять не полную версию изменённых файлов, а производить так называемую дельта-компрессию — сохранять только изменения между последовательными версиями, что позволяет уменьшить объём хранимых данных. Системы контроля версий поддерживают возможность отслеживания и разрешения конфликтов, которые могут возникнуть при работе нескольких человек над одним файлом. Можно объединить (слить) изменения, сделанные разными участниками (автоматически или вручную), вручную выбрать нужную версию, отменить изменения вовсе или заблокировать файлы для изменения. В зависимости от настроек блокировка не позволяет другим пользователям получить рабочую копию или препятствует изменению рабочей копии файла средствами файловой системы ОС, обеспечивая таким образом, привилегированный доступ только одному пользователю, работающему с файлом. Системы контроля версий также могут обеспечивать дополнительные, более гибкие функциональные возможности. Например, они могут поддерживать работу с несколькими версиями одного файла, сохраняя общую историю изменений до точки ветвления версий и собственные истории изменений каждой ветви. Кроме того, обычно доступна информация о том, кто из участников, когда и какие изменения вносил. Обычно такого рода информация хранится в журнале изменений, доступ к которому можно ограничить. В отличие от классических, в распределённых системах контроля версий центральный репозиторий не является обязательным. Среди классических VCS наиболее известны CVS, Subversion, а среди распределённых — Git, Bazaar, Mercurial. Принципы их работы схожи, отличаются они в основном синтаксисом используемых в работе команд.

# 3 Выполнение лабораторной работы

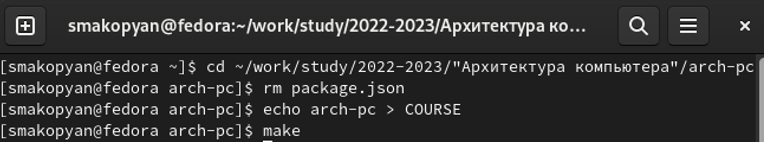
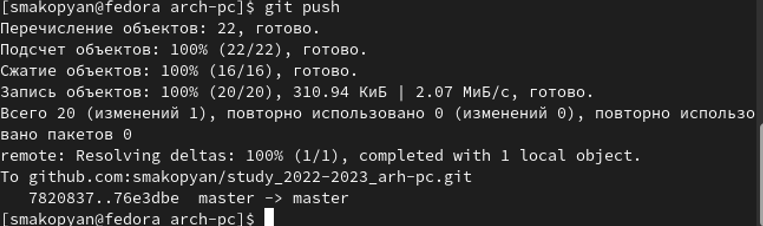
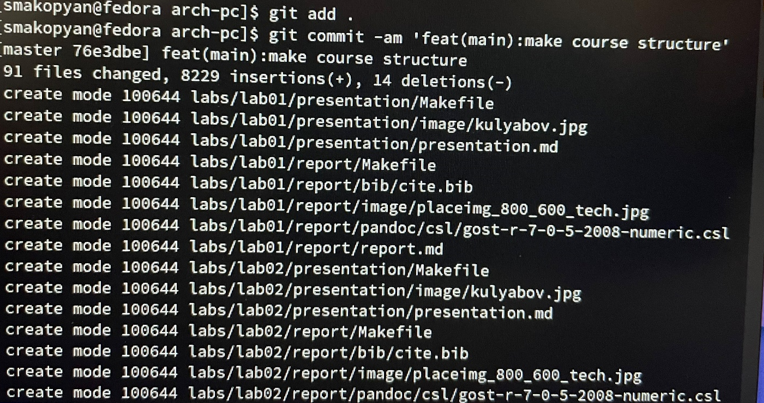
1. Настройка github. Создаем учётную запись на сайте https://github.com/ и заполняем основные данные. (рис. **¿fig:fig1?**)  { #fig:fig1 width=70% }
2. Базовая настройка git Сделаем предварительную конфигурацию git, а так же настроим utf-8 в выводе сообщений git и зададим имя начальной ветки (будем называть её master). (рис. **¿fig:fig2?**)  { #fig:fig2 width=70% }
3. Создание SSH ключа

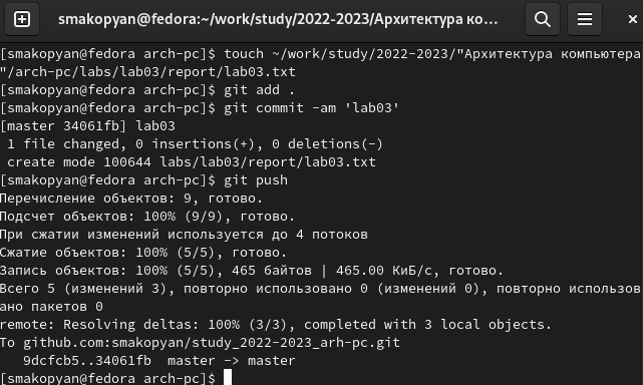
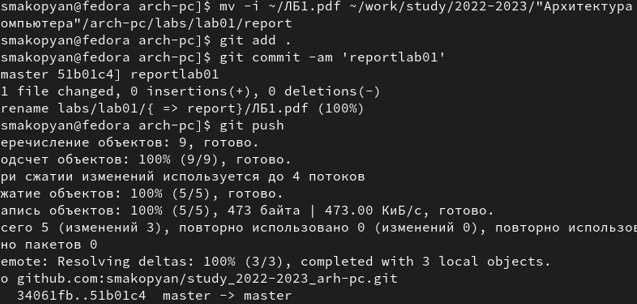
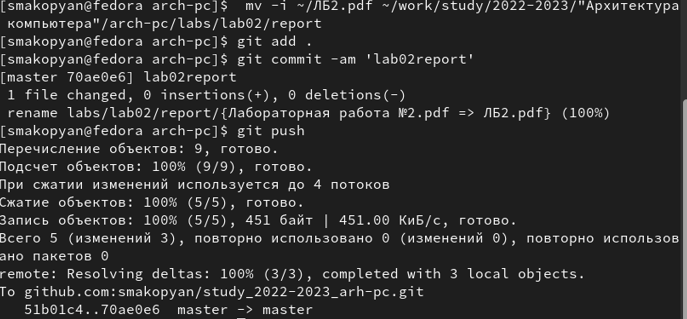
3.1 Для последующей идентификации пользователя на сервере репозиториев необходимо сгенерировать пару ключей (приватный и открытый (рис. **¿fig:fig3?**)  { #fig:fig3 width=70% } 3.2 Далее необходимо загрузить на сайт сгенерённый открытый ключ (рис. **¿fig:fig4?**)  { #fig:fig4 width=70% }

1. Создание рабочего пространства и репозитория курса на основе шаблона. Открываем терминал и создаем каталог для предмета «Архитектура компьютера» (рис. **¿fig:fig5?**)  { #fig:fig5 width=70% }
2. Создание репозитория курса на основе шаблона

5.1 Репозиторий на основе шаблона можно создать через web-интерфейс github, для этого переходим на страницу репозитория с шаблоном курса, задаем имя репозитория study\_2022–2023\_arh-pc и создаем его.(рис. **¿fig:fig6?**)  { #fig:fig6 width=70% }

5.2 Далее открываем терминал и переходим в каталог курса “Архитектура компьютера”.(рис. **¿fig:fig7?**) рисунок 7 { #fig:fig7 width=70% } 5.3 Клонируем созданный репозиторий(рис. **¿fig:fig8?**) рисунок 8 { #fig:fig8 width=70% } 6. Настройка каталога курса.

6.1 Переходим в каталог курса, удаляем лишние файлы и создаем необходимые каталоги.(рис.9)  { #fig:fig9 width=70% } 6.2 Отправляем файлы на сервер. (рис.10, рис.11)  { #fig:fig10 width=70% }  { #fig:fig11 width=70% } # Задание для самостоятельной работы

1. Создайте отчет по выполнению лабораторной работы в соответствующем каталоге рабочего пространства (labs>lab03>report). (рис.13)  { #fig:fig13 width=70% }
2. Скопируйте отчеты по выполнению предыдущих лабораторных работ в соответствующие каталоги созданного рабочего пространства.
3. Загрузите файлы на github. (рис.14, рис.15)  { #fig:fig14 width=70% }  { #fig:fig15 width=70% } # Выводы

В результате выполнения данной лабораторной работы я изучила идеологию и применение средств контроля версий, приобрела практические навыки по работе с системой git.