МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ   
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ»

Факультет физико-математических и естественных наук

ОТЧЕТ

По лабораторной работе №2

«Управление версиями»

Выполнил:

Студент группы: НПИбд-02-21

Студенческий билет: № 1032217060

ФИО студента: Королев Адам Маратович

Дата выполнения: 21.04.2022

Москва 2022

Цель работы:

- Изучить идеологию и применение средств контроля версий.

- Освоить умения по работе с git.

Задание:

- Создать базовую конфигурацию для работы с git.

- Создать ключ SSH.

- Создать ключ PGP.

- Настроить подписи git.

- Создать локальный каталог для выполнения заданий по предмету.

Теоретическое введение:

Git – это распределенная система контроля версий, которая дает возможность разработчикам отслеживать изменения в файлах и коллективно работать над одним проектом. Она была разработана в 2005 году Линусом Торвальдсом, создателем Linux, чтобы другие разработчики могли вносить свой вклад в ядро Linux.

GitHub – это сервис онлайн-хостинга репозиториев, который обладает всеми функциями распределенного контроля версий и функциональностью управления исходным кодом.

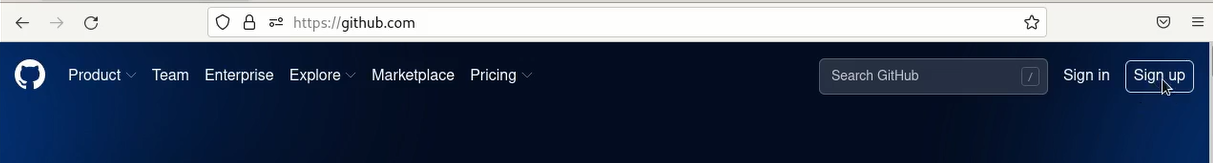
SSH ключ – это специальный код, который позволяет удаленному компьютеру определить пользователя и понять, какими правами на компьютере он обладает.

Ключ SSH в свою очередь разделен на две части. Одна называется приватной и хранится только на вашем компьютере. Вторая часть называется публичной и эту часть необходимо отправлять на другие устройства. При подключении к удаленному устройству сравнивается публичная часть, которую вы предоставили, с приватной частью, хранящейся у вас. Если части совпадают, то вы получаете необходимый доступ. На компьютере может быть создано сколько угодно SSH-ключей.

PGP ключ – это криптографическая программа, которая позволяет зашифровать информацию таким образом, чтобы никто не мог ни прочитать, ни изменить данные. Это обеспечивает надежную безопасность файлов и гарантирует их секретность. Также PGP ключ может использоваться для того, чтобы обозначить ваше авторство, то есть, чтобы никто не мог присвоить себе ваш труд.

Выполнение лабораторной работы:

1. Необходимо создать учетную запись на <https://github.com>



Рисунок

1. Установить git-flow в Fedora Linux. Устанавливается оно в ручную, так как оно удалено из репозитория.



Рисунок

1. Необходимо установить gh в Fedora Linux



Рисунок

1. Необходимо произвести базовую настройку git. То есть, задать имя и email владельца репозитория.



Рисунок

1. Настроим utf-8 в выводе сообщений git.



Рисунок

Настроим верификацию и подписание коммитов git:

1. Зададим имя начальной ветки (будем называть ее master).



Рисунок

1. Установим параметр autocrlf.



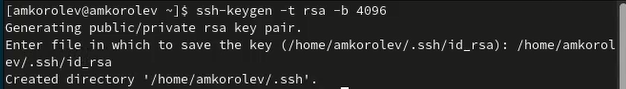
Рисунок

1. Установим параметр safecrlf.



Рисунок

1. По алгоритму rsa создадим ключ ssh с размером 4096 бит.



Рисунок

1. По алгоритму ed25519 создадим ключ ssh.



Рисунок

1. Создадим ключ pgp путем генерации.

Из предложенных опций выбираем:

- тип RSA and RSA;

- размер 4096;

- устанавливаем срок действия, параметр 0 (не истекает никогда);

Далее GPG запросит личную информацию, которая сохранится в ключе:

- Имя;

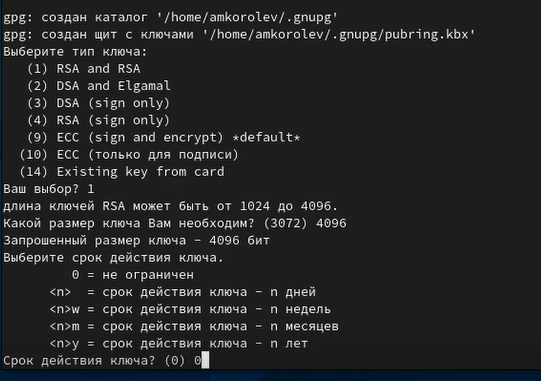
- Адрес электронной почты;

- Email (необходимо ввести email, используемый на GitHub)

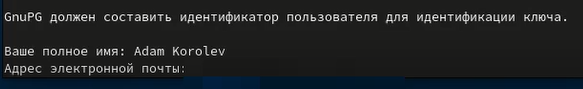
- Комментарий. Можно оставить пустым.



Рисунок



Рисунок



Рисунок

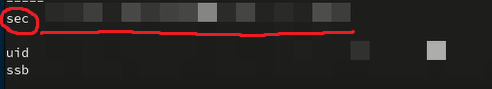


Рисунок

13. Выводим список ключей и копируем отпечаток приватного ключа.

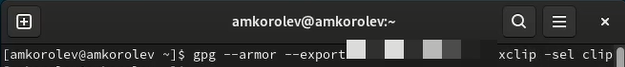


Рисунок



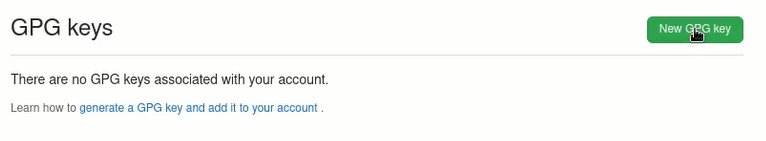
Рисунок

14. Скопируем наш сгенерированный PGP ключ в буфер обмена.

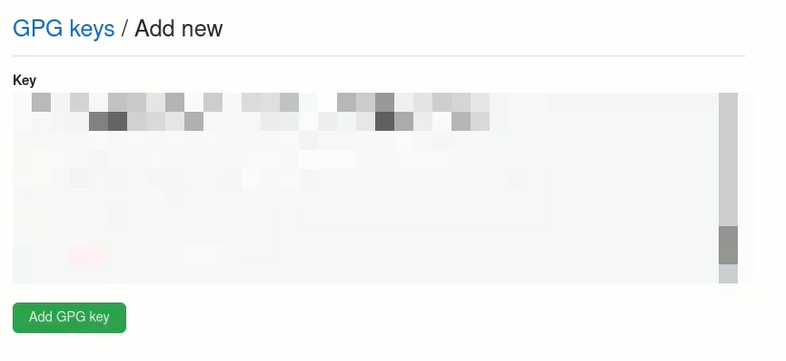


Рисунок

15. Перейдем в настройки GitHub, нажмем на кнопку New GPG key и вставим полученный ключ в поле ввода.



Рисунок



Рисунок

16. Используя введенный email, укажем Git применять его при подписи коммитов.



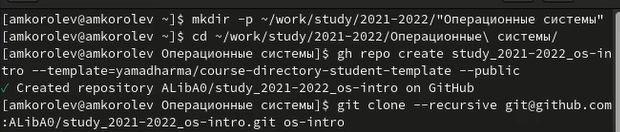
Рисунок

17. Произведем авторизацию в GitHub.



Рисунок

18. Создадим шаблон рабочего пространства.



Рисунок

19. Перейдем в каталог курса.



Рисунок

20. Удалим лишние файлы.



Рисунок

21. Создадим необходимые каталоги.



Рисунок

20. Отправим файлы на сервер.



Рисунок



Рисунок

Выводы:

В процессе выполнения задания были приобретены умения по работе с git. Была изучена идеология и применения средств контроля версий.

Ответы на контрольные вопросы:

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

Ответ: Системы контроля версий (VCS) применяются при работе нескольких человек над одним проектом, совместная работа путем изменения файлов в одном репозитории.

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

Ответ: Хранилище – общее пространство для хранения файлов

Commit – команда для записи индексированных изменений в репозиторий

История - в истории сохраняются все коммиты, по которым можно отследить автора сообщения, дату и хэш коммита

Рабочая копия - все файлы кроме .git/ называются рабочей копией, и принадлежат пользователю (пользователям)

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

Ответ: Централизованные системы контроля версий – сохраняют проект и его файлы на один общий север, децентрализованные системы контроля версий – при каждом копировании данных удаленного репозитория, происходит полное копирование данных в локальный репозиторий. Пример ЦСКВ – SVN, MS TFS, ClearCase; ДСКВ – Git, Mercurial, Bazaar.

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

Ответ: 1. Создаем репозиторий, именуем его

2. Добавляем файлы в репозиторий

3. Фиксируем с помощью коммитов

4. Изменяем файлы репозитория и фиксируем изменения

1. Опишите порядок работы с общим хранилищем VCS.

Ответ: 1. Создаем репозиторий, именуем его или присоединяемся к нему в качестве contributor

2. Добавляем файлы в репозиторий

3. Фиксируем с помощью коммитов.

4. Изменяем файлы репозитория и фиксируем изменения

5. Ждем проверки коммитов при участии других пользователей в общем репозитории

1. Каковы основные задачи, решаемые инструментальным средством git?

Ответ: Систематизация, параллельность разработки программного обеспечения, единое место для хранения файлов проекта.

1. Назовите и дайте краткую характеристику командам git.

Ответ: Создание репозитория (git init). Клонирование репозитория (git clone). Добавление изменений в индекс (git add). Удаление изменений из индекса (git reset). Коммиты (git commit). Удаление файла (git rm).

1. Приведите примеры использования при работе с локальным и удаленным репозиториями.

Ответ: Для написания черновых работ по лабораторным работам используем локальные репозитории, для их распространения или для оценивания используем удаленный репозиторий git.

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

Ответ: Ветви служат для параллельной разработки программного обеспечения, тестирования, отладки и улучшения.

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

Ответ: Игнорирование можно установить для проекта, компьютера и репозитория. Цель игнорирования заключается в том, чтобы не отслеживать файлы служебного типа, например временные файлы сборных утилит для проектов или только те файлы, которые полезны при взаимодействии только с очень ограниченным программным обеспечением.