**РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ**

**Факультет физико-математических и естественных наук**

**Кафедра прикладной информатики и теории вероятностей**

**ОТЧЕТ**

**по лабораторной работе № 2**

*дисциплина: Операционные системы*

Студент: Иванов Н.В.

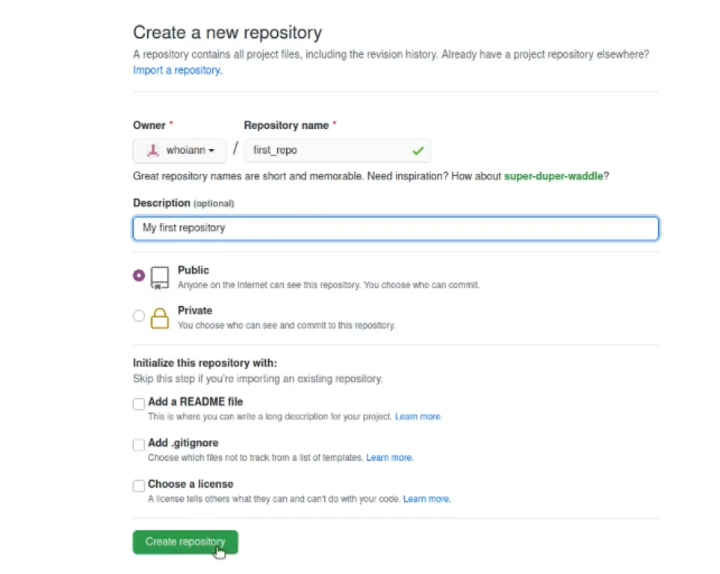
Группа: НБИбд-01-20

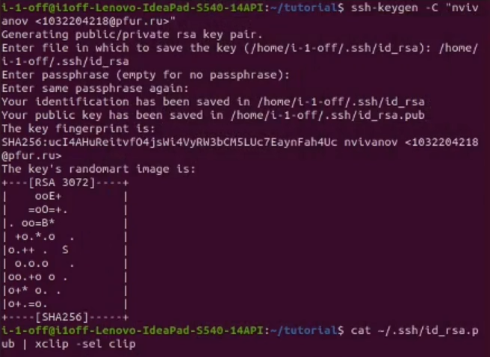
**МОСКВА**

2021 г.

**Цель работы:** Изучить идеологию и применение средств контроля версий.  
**Порядок выполнения работы:**

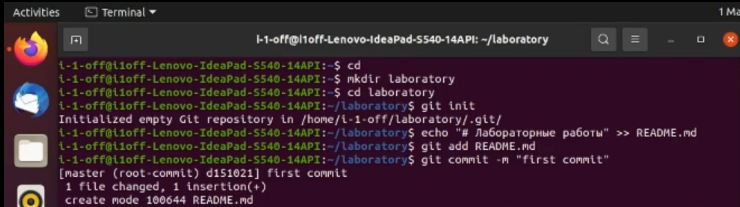
1. Настройка git
   1. Создал учётную запись на github.com и тестовый репозиторий
   2. Командой **ssh-keygen** сгенерировал ssh ключ и сохранил его в файле
   3. Командой **cat** и **xclip** скопировал его в буфер обмена и вставил в соответствующее поле в настройках github
   4. Командой **git** **remote add origin** подключил локальный репозиторий к удалённому



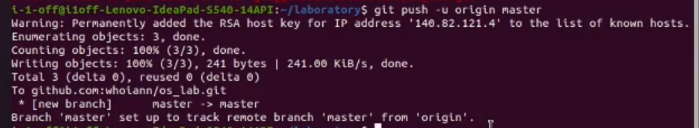




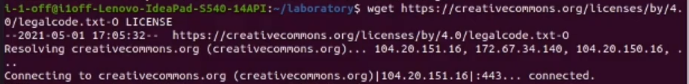
1. Подключение локального репозитория к Github.
   1. Командой **mkdir** создал каталог для лабораторных работ
   2. Командой **cd** переместился в этот каталог
   3. Командой **git init** инициализировал в этом каталоге git
   4. Командой **echo** создал файл и вписал в него текст
   5. Командой **git add** добавил файл в git
   6. Командой **git commit** создал первый коммит
   7. Командой **git remote add** подключил удалённый репозиторий
   8. Командой **git push** выложил все изменения в локальном репозитории на удалённый

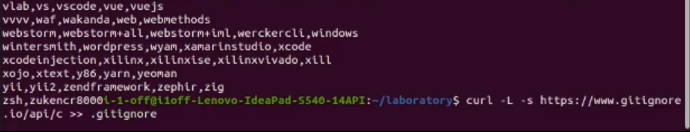


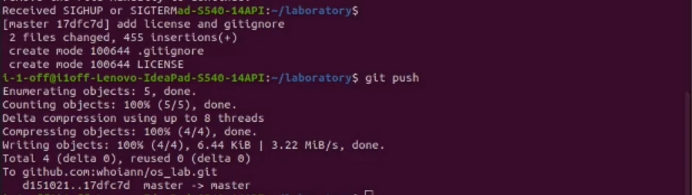


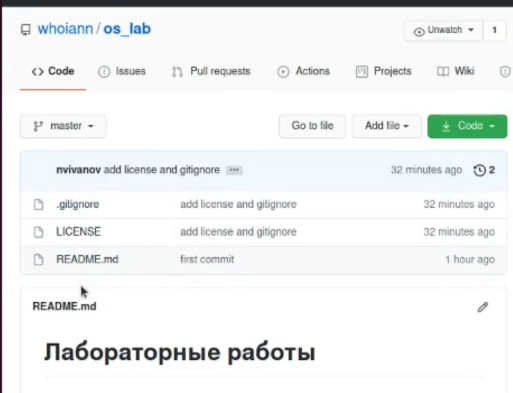


1. Настройка первичной конфигурации.
   1. Командой **wget** добавляю файл лицензии в локальный репозиторий
   2. Командой **curl** посмотрел и добавил шаблон игнорируемых файлов
   3. Командой **git add** добавил новые файлы в git
   4. Командой **git commit** выполнил коммит
   5. Командой **git push** отправил все изменения в удалённый репозиторий



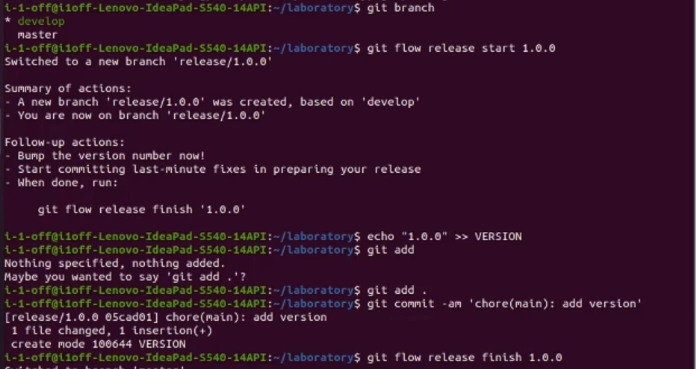


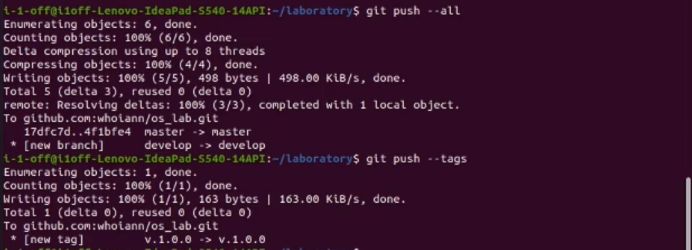




1. Настройка конфигурации git-flow
   1. Командой **git flow init** инициализирую git-flow в лабораторном каталоге
   2. Устанавливаю названия для веток разработки и префикс для версий
   3. Командой **git branch** проверил, на какой ветке сейчас нахожусь
   4. Командой **git flow** release start создал релиз версии 1.0.0
   5. Командой **echo** записал версию
   6. Командой **git add** и **git commit** добавил в индекс
   7. Командой **git flow release finish** слил релизную ветку с основной
   8. Командой **git push –all** и **git push –tags** отправил данные на github и создал релиз







**Выводы:** В ходе данной работы изучил идеологию и применение средств контроля версий.

**Контрольные вопросы:**

1. Системы контроля версий предназначаются для систематизации процесса поддержки и обновления программного обеспечения, своевременного выхода обновлений и исправления в нём критических ошибок.
2. Хранилище – сервер, удалённое место хранения и обмена рабочих файлов

Коммит – текстовое сопровождение изменений, загружаемых в хранилище. Служит для фиксации внесённых изменений, говорит о проделанной работе.

История – последовательность версий разрабатываемой программы/продукта, сохраняется в ветку master.

Рабочая копия – файл, загружаемый из удалённого репозитория на локальный для проведения над ним работ, внесения изменений.

1. Централизованные VCS – существует единственный центральный репозиторий, из которого берутся рабочие копии(subversion, CVS). Децентрализованные VCS – каждая подключённая рабочая машина может выступать в роли полноценного хранилища (git, mercurial).
2. Разработал программу, «запушил» все изменения в хранилище, доработал программу, «запушил» все изменения в хранилище.
3. Распределили роли в команде, скопировал нужную мне рабочую копию, доработал её, «запушил» изменения в хранилище, «слил» свою часть работы с работой всей команды, заново скопировал рабочую версию.
4. Контроль версий, возможность совместной системной работы над продуктом, выгрузка и загрузка данных в хранилище, сливание веток разработки.
5. git <действие>
6. Локальный репозиторий – рабочее пространство программиста, в котором он выполняет свою работу, например, пока совершает трансатлантический перелёт, во время которого нет доступа к удалённому репозиторию. Удалённый репозиторий – место обмена и слияния рабочих файлов, куда, например, наш программист загрузит результат длительной работы в самолёте.
7. Ветви – логическое разделение направлений разработки. Отдельная история разработки фичей, отдельная история багафиксов, и отдельная история готовых релизов. Ветви позволяют структурировать процесс разработки и не потеряться в гигабайтах исходников.
8. Хорошим тоном считается игнорирование бинарных (скомпилированных) файлов в разработки, так как они не несут никакой пользы для программистов. Реализуется посредством записи определённых расширений в файл .gitignore