

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ
Факультет физико-математических и естественных наук
Кафедра прикладной математики и информатики

ОТЧЁТ
ПО ЛАБАРАТОРНОЙ РАБОТЕ №3

Студент: Смирнов Дмитрий Романович

Группа: НММбд-03-22

МОСКВА

2022 г

Цель работы:

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

Ход работы:

Для начала проведу базовую настройку git (рис 1.0)

```
[smirnovd_03_22@10 ~]$ git config --global user.name "Smirnov_Dmitry"
[smirnovd_03_22@10 ~]$ git config --global user.email "1132221813@pfur.ru"
[smirnovd_03_22@10 ~]$ git config --global core.quotepath false
[smirnovd_03_22@10 ~]$ git config --global init.defaultBranch master
[smirnovd_03_22@10 ~]$ git config --global core.autocrlf input
[smirnovd_03_22@10 ~]$ git config --global core.safecrlf warn
[smirnovd_03_22@10 ~]$
```

Рис 1.0

Сгенерирую ключ для последующей идентификации пользователя на сервере репозитория (рис 1.1)

```
[smirnovd_03_22@10 ~]$ ssh-keygen -C "Dmitriy Smirnov 1132221813@pfur.ru"
Generating public/private rsa key pair.
Enter file in which to save the key (/home/smirnovd_03_22/.ssh/id_rsa):
Created directory '/home/smirnovd_03_22/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/smirnovd_03_22/.ssh/id_rsa
Your public key has been saved in /home/smirnovd_03_22/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:c2DXIvoJmB45DWxs80DIP8aw0rlZk4gHcKfH9D3H9I Dmitriy Smirnov 1132221813@pfur.ru
The key's randomart image is:
+---[RSA 3072]-----+
|      oo          |
| .   . . . .   .  |
| o.o...o.+ o .    |
| =.oo0.=.+..     |
| ..o+B.O So.E     |
| ..+o.o.= +o .    |
| .+=o.  o .       |
| ..=+             |
| . ..             |
+---[SHA256]-----+
[smirnovd_03_22@10 ~]$
```

Рис 1.1

Скопирую ssh ключ и вставлю на GitHub (рис 1.2, рис 1.3)

```
[smirnovd_03_22@10 ~]$ cat ~/.ssh/id_rsa.pub | xclip -sel clip
[smirnovd_03_22@10 ~]$
```

Рис 1.2

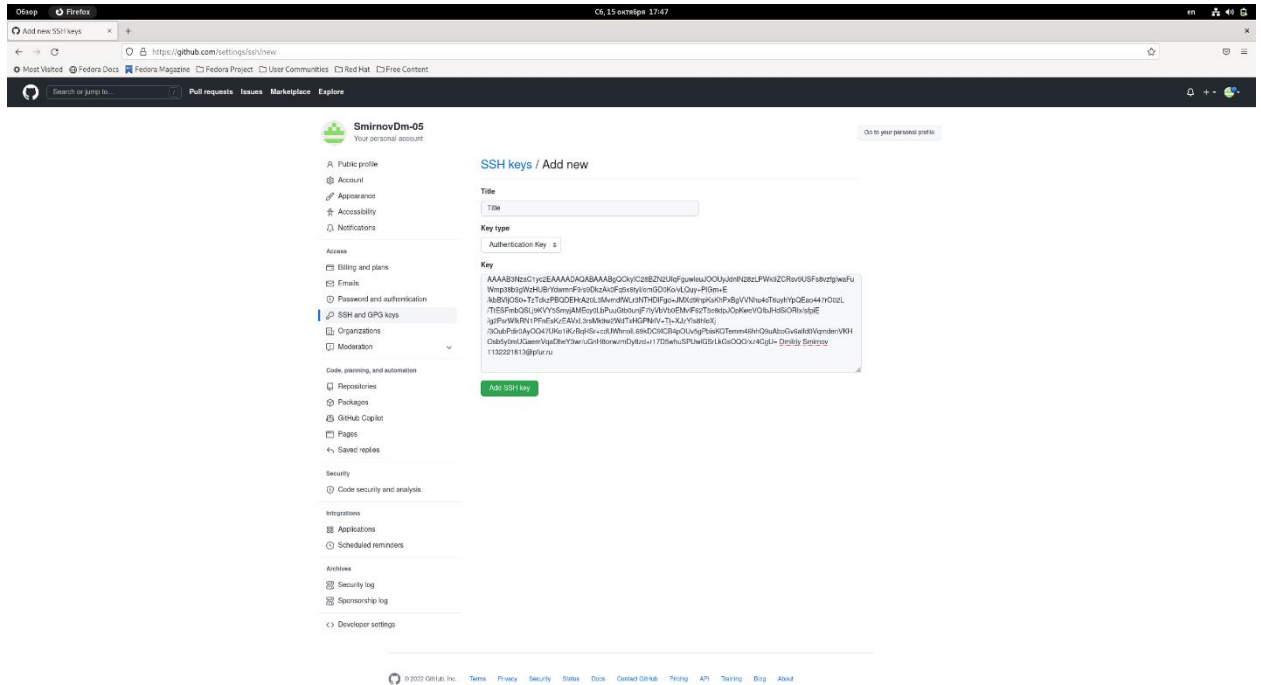


Рис 1.3

Создам каталог для предмета «Архитектура компьютера» (рис 1.4)

```
[smirnovd_03_22@10 ~]$ mkdir -p ~/work/study/2022-2023/"Архитектура компьютера"
[smirnovd_03_22@10 ~]$
```

Рис 1.4

Создам репозиторий на основе предоставленного шаблона.

Скопирую созданный репозиторий (рис 1.5)

```
[smirnovd_03_22@10 Архитектура компьютера]$ git clone --recursive git@github.com:SmirnovDm-05/study_2022-2023_arh-pc.git
Клонирование в «study_2022-2023_arh-pc»...
The authenticity of host 'github.com (140.82.121.4)' can't be established.
ED25519 key fingerprint is SHA256:+DiY3wvV6TuJJhbpZisF/zLDA0zPMSvHdkr4UvC0qU.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'github.com' (ED25519) to the list of known hosts.
remote: Enumerating objects: 26, done.
remote: Counting objects: 100% (26/26), done.
remote: Compressing objects: 100% (25/25), done.
remote: Total 26 (delta 0), reused 17 (delta 0), pack-reused 0
Получение объектов: 100% (26/26), 16.39 Киб | 16.39 Миб/с, готово.
Подмодуль «template/presentation» (https://github.com/yamadharma/academic-presentation-markdown-template.git) зарегистрирован по пути «template/presentation»
Подмодуль «template/report» (https://github.com/yamadharma/academic-laboratory-report-template.git) зарегистрирован по пути «template/report»
Клонирование в «/home/smirnovd_03_22/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arh-pc/template/presentation»...
remote: Enumerating objects: 71, done.
remote: Counting objects: 100% (71/71), done.
remote: Compressing objects: 100% (49/49), done.
remote: Total 71 (delta 23), reused 68 (delta 20), pack-reused 0
Получение объектов: 100% (71/71), 88.89 Киб | 892.00 Киб/с, готово.
Определение изменений: 100% (23/23), готово.
Клонирование в «/home/smirnovd_03_22/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arh-pc/template/report»...
remote: Enumerating objects: 78, done.
remote: Counting objects: 100% (78/78), done.
remote: Compressing objects: 100% (52/52), done.
remote: Total 78 (delta 31), reused 69 (delta 22), pack-reused 0
Получение объектов: 100% (78/78), 292.27 Киб | 1.32 Миб/с, готово.
Определение изменений: 100% (31/31), готово.
Submodule path 'template/presentation': checked out '2703b47423792d472694aaf7555a5626dce51a25'
Submodule path 'template/report': checked out 'df7b2ef80f8def3b9a496f8695277469a1a7842a'
[smirnovd_03_22@10 Архитектура компьютера]$
```

Рис 1.5

Перейду в каталог курса. Удалю файл package.json и создам COURSE (рис 1.6)

```
[smirnovd_03_22@10 ~]$ cd work/study/2022-2023/"Архитектура компьютера"/study_2022-2023_arh-pc
[smirnovd_03_22@10 study_2022-2023_arh-pc]$ ls
CHANGELOG.md  config  COURSE  LICENSE  Makefile  package.json  README.en.md  README.git-flow.md  README.md  template
[smirnovd_03_22@10 study_2022-2023_arh-pc]$ rm package.json
[smirnovd_03_22@10 study_2022-2023_arh-pc]$ echo arch-pc > COURSE
[smirnovd_03_22@10 study_2022-2023_arh-pc]$ make
[smirnovd_03_22@10 study_2022-2023_arh-pc]$ ls
CHANGELOG.md  config  COURSE  labs  LICENSE  Makefile  prepare  README.en.md  README.git-flow.md  README.md  template
[smirnovd_03_22@10 study_2022-2023_arh-pc]$
```

Рис 1.6

Отправляю файлы на сервер с помощью команд:

git add .

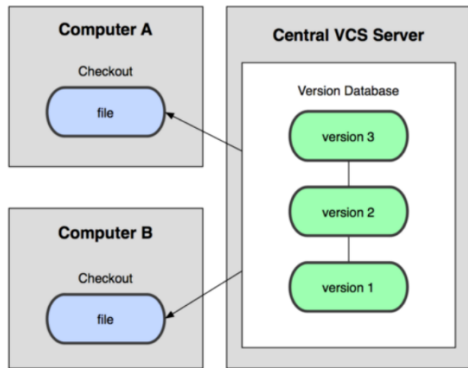
git commit -am 'feat(main): make course structure'

git push

Контрольные вопросы для самопроверки

1. Что такое системы контроля версий (VCS) и для решения каких задач они предназначаются?
Система контроля версий — это система, регистрирующая изменения в одном или нескольких файлах с тем, чтобы в дальнейшем была возможность вернуться к определённым старым версиям этих файлов. Применяются при работе нескольких человек над одним проектом.
2. Объясните следующие понятия VCS и их отношения: хранилище, commit, история, рабочая копия.
Хранилище, или репозиторий, — место хранения всех версий и служебной информации.
Commit - зафиксированный набор изменений, который показывает, какие файлы изменились и что именно в них изменилось.
Рабочая копия — текущее состояние файлов проекта, основанное на версии, загруженной из хранилища

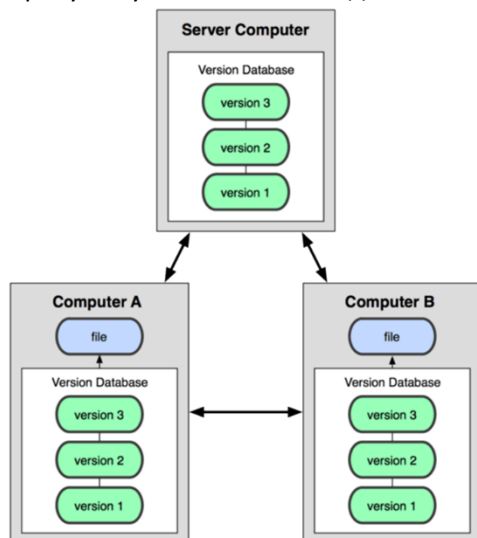
3. Что представляют собой и чем отличаются централизованные и децентрализованные VCS?
Приведите примеры VCS каждого вида.
Централизованные VCS.
Одно основное хранилище всего проекта.
Каждый пользователь копирует себе необходимые ему файлы из этого репозитория,
изменяет и, затем, добавляет свои изменения обратно



Децентрализованные VCS.

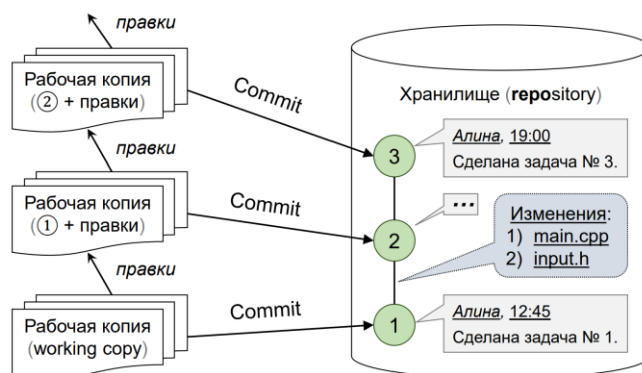
У каждого пользователя свой вариант (возможно не один) репозитория.

Присутствует возможность добавлять и забирать изменения из любого репозитория



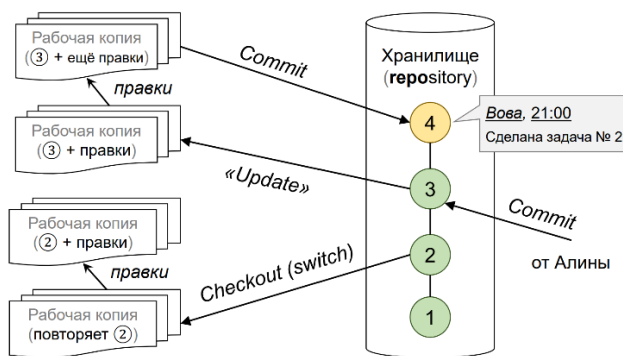
4. Опишите действия с VCS при единоличной работе с хранилищем.

Единоличная работа с VCS



5. Опишите порядок работы с общим хранилищем VCS.

Работа с общим хранилищем



6. Каковы основные задачи, решаемые инструментальным средством git?

У Git две основные задачи: первая — хранить информацию о всех изменениях в вашем коде, начиная с самой первой строчки, а вторая — обеспечение удобства командной работы над кодом.

7. Назовите и дайте краткую характеристику командам git.

`git init` создание основного дерева репозитория

`git pull` получение обновлений (изменений) текущего дерева из центрального репозитория

`git push` отправка всех произведённых изменений локального дерева в центральный репозиторий

`git status` просмотр списка изменённых файлов в текущей директории

`git diff` просмотр текущих изменений

`git add .` добавить все изменённые и/или созданные файлы и/или каталоги

`git add имена_файлов` добавить конкретные изменённые и/или созданные файлы и/или каталоги

`git rm имена_файлов` удалить файл и/или каталог из индекса репозитория (при этом файл и/или каталог остаётся в локальной директории)

`git commit -am 'Описание коммита'` сохранить все добавленные изменения и все изменённые файлы

`git checkout -b имя_ветки` создание новой ветки, базирующейся на текущей

`git checkout имя_ветки` переключение на некоторую ветку (при переключении на ветку, которой ещё нет в локальном репозитории, она будет создана и связана с удалённой)

`git push origin имя_ветки` отправка изменений конкретной ветки в центральный репозиторий

`git merge --no-ff имя_ветки` слияние ветки с текущим деревом

`git branch -d имя_ветки` удаление локальной уже слитой с основным деревом ветки

`git branch -D имя_ветки` принудительное удаление локальной ветки

`git push origin :имя_ветки` удаление ветки с центрального репозитория

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

Команда `git fetch` связывается с удалённым репозиторием и забирает из него все изменения, которых у вас пока нет и сохраняет их локально.

Команда `git pull` вначале забирает изменения из указанного удалённого репозитория, а затем пытается слить их с текущей веткой.

Команда `git push` используется для установления связи с удалённым репозиторием, вычисления локальных изменений отсутствующих в нём, и собственно их передачи в вышеупомянутый репозиторий.

Вывод: Я изучил идеологию и применение средств контроля версий, а также приобрел практические навыки по работе с системой git.