

Лабораторная работа №3

Дисциплина: Операционные системы

Кондратьев Арсений Вячеславович

14.09.2022

Содержание

1	Цель работы	3
2	Задание	4
3	Теоретическое введение	5
3.1	Основные команды git	5
4	Выполнение лабораторной работы	6
5	Выводы	15
6	Контрольные вопросы	16

1 Цель работы

Научиться оформлять отчёты с помощью легковесного языка разметки Markdown.

2 Задание

Сделайте отчёт по предыдущей лабораторной работе в формате Markdown.

В качестве отчёта просьба предоставить отчёты в 3 форматах: pdf, docx и md (в архиве, поскольку он должен содержать скриншоты, Makefile и т.д.)

3 Теоретическое введение

3.1 Основные команды git

git init

git pull

git push

git status

git diff

git add

git rm

git commit

git checkout -b имя_ветки

4 Выполнение лабораторной работы

1. Создал учетную запись на Github(рис.4.1)

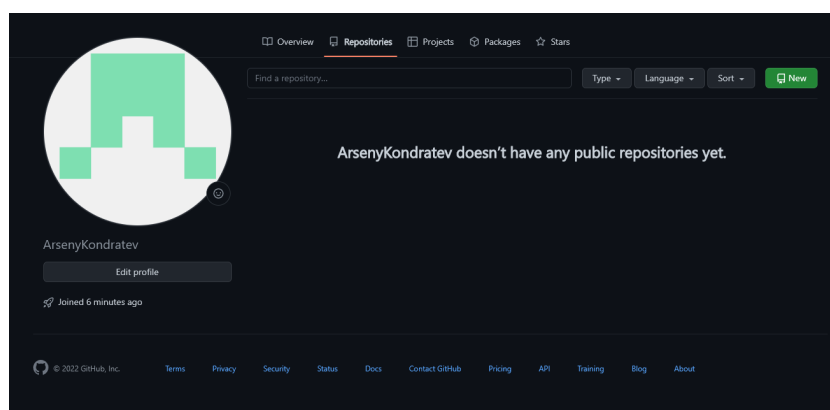


Figure 4.1: Рис. 1

2. Установил Git-flow(рис.4.2)

```
cd /tmp
wget --no-check-certificate -q https://raw.githubusercontent.com/petervanderdoes
❏ /gitflow/develop/contrib/gitflow-installer.sh
chmod +x gitflow-installer.sh
sudo ./gitflow-installer.sh install stable
```

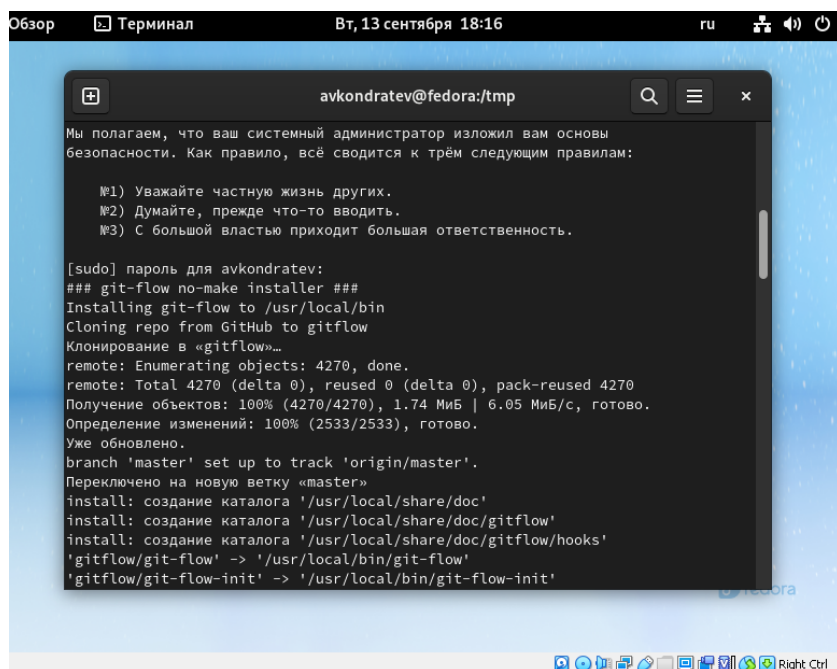


Figure 4.2: Рис. 2

3. Установил gh в Fedora Linux(рис.4.3)

`sudo dnf install gh`

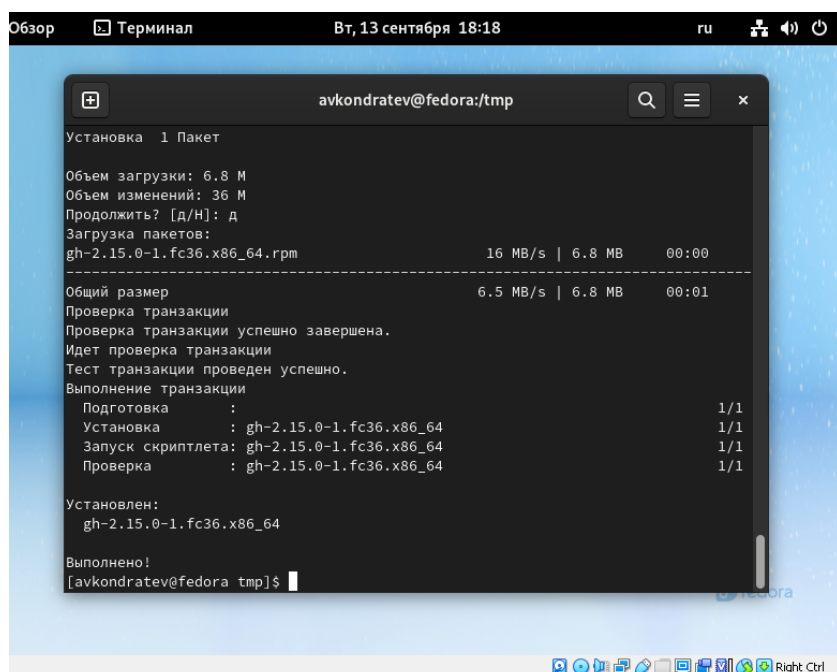


Figure 4.3: Рис. 3

4. Выполнил базовую настройку Git(рис.4.4)

```
git config --global user.name "KondratevArseny"  
git config --global user.email avk2200@yandex.ru  
git config --global core.quotepath false  
git config --global init.defaultBranch master  
git config --global core.autocrlf input > git config --global core.safecrlf warn
```

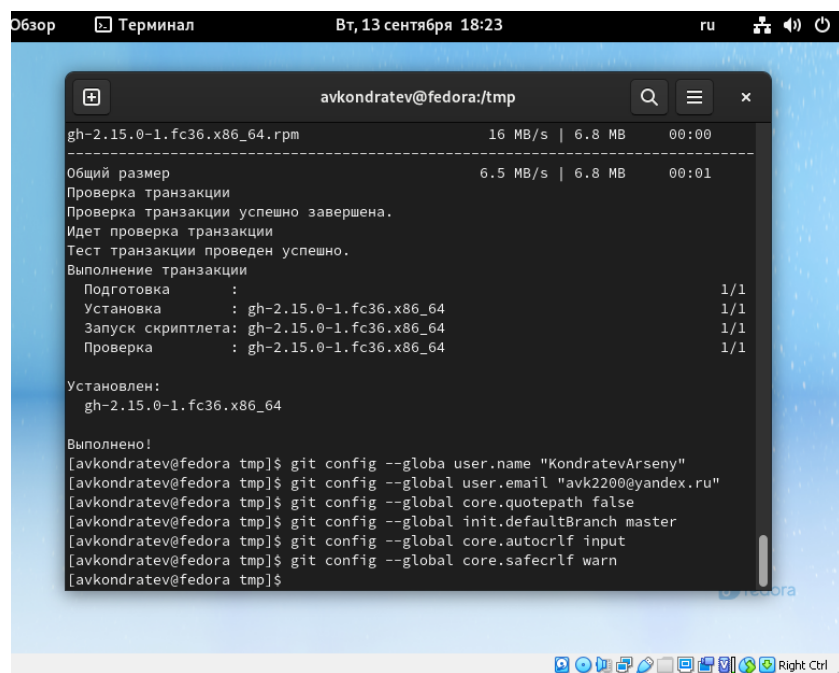
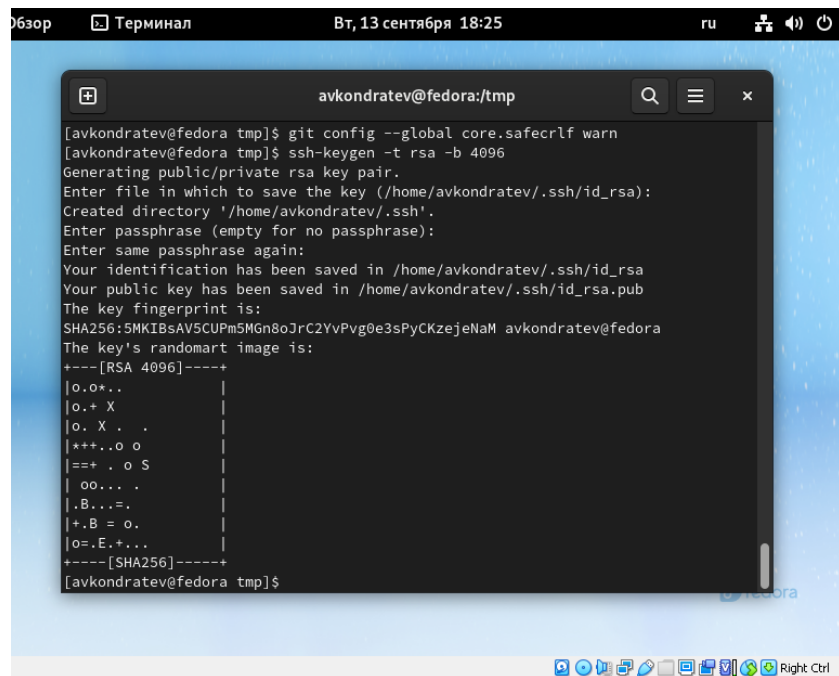


Figure 4.4: Рис. 4

5. Создал ключ SSH по алгоритму rsa(рис.4.5)

```
ssh-keygen -t rsa -b 4096
```

Обзор Терминал Вт, 13 сентября 18:25 ru

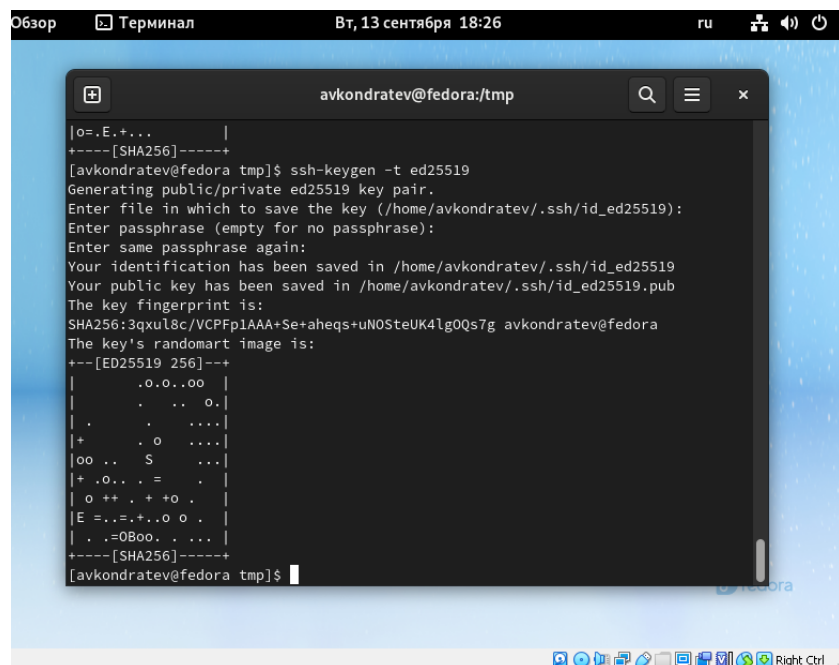
```
avkondratev@fedora/tmp
[avkondratev@fedora tmp]$ git config --global core.safecrlf warn
[avkondratev@fedora tmp]$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/avkondratev/.ssh/id_rsa):
Created directory '/home/avkondratev/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/avkondratev/.ssh/id_rsa
Your public key has been saved in /home/avkondratev/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:5MKIBsAV5CUPm5MGn8oJrC2YvPvg0e3sPyCKzejeNaM avkondratev@fedora
The key's randomart image is:
+---[RSA 4096]-----+
|o.o*..|
|o.+ X |
|o. X . .|
|+++..o o|
|==+ . o S|
| oo... .|
|.B...=|
|+.B = o.|
|o=.E.+...|
+---[SHA256]-----+
[avkondratev@fedora tmp]$
```

Right: Ctrl

Figure 4.5: Рис. 5

6. Создал ключ SSH по алгоритму ed25519(рис.4.6)

`ssh-keygen -t ed25519`



Обзор Терминал Вт, 13 сентября 18:26 ru

```
avkondratev@fedora/tmp
[avkondratev@fedora tmp]$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/avkondratev/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/avkondratev/.ssh/id_ed25519
Your public key has been saved in /home/avkondratev/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:3qxul8c/VCPFP1AAA+Se+ahqqs+uN0SteUK4lg0Qs7g avkondratev@fedora
The key's randomart image is:
+---[ED25519 256]---+
|      .o.o..oo |
|      . . . o |
|      . . ....|
|+      . o ....|
|oo .. S ....|
|+ .o.. = .|
| o ++ . + +o .|
|E =..=+..o o .|
| . . =0Boo. . ...|
+---[SHA256]-----+
[avkondratev@fedora tmp]$
```

Right: Ctrl

Figure 4.6: Рис. 6

7. Создал ключи pgr(рис.4.7)

```
gpg --full-generate-key
```

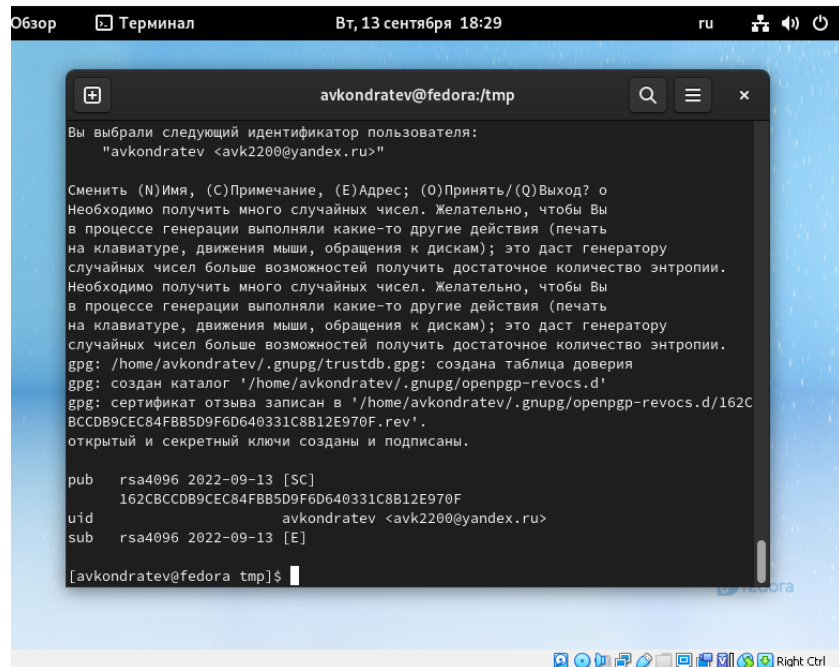


Figure 4.7: Рис. 7

8. Добавил ключ pgr в github(рис.4.8, рис.4.9)

```
gpg --armor --export <PGP Fingerprint> | xclip -sel clip
```

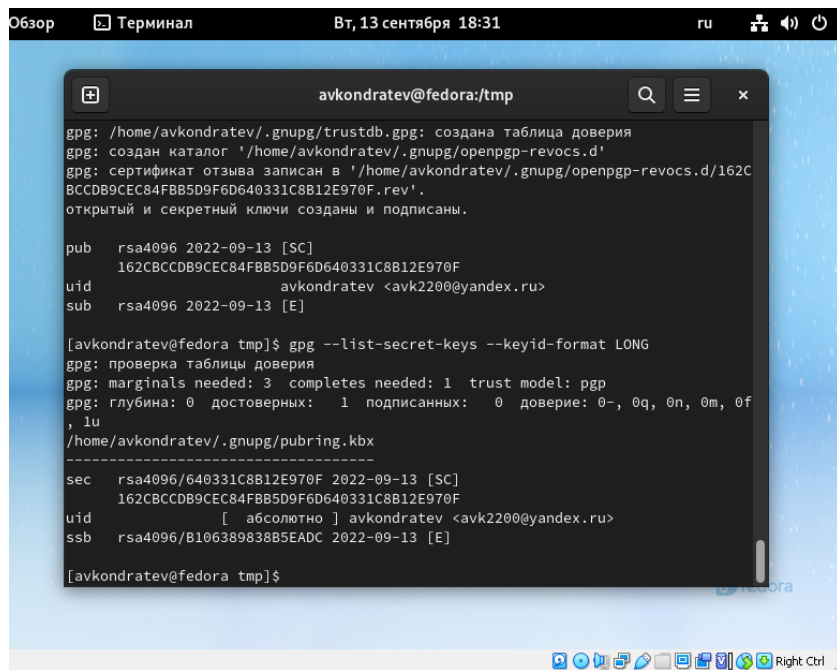


Figure 4.8: Рис. 8

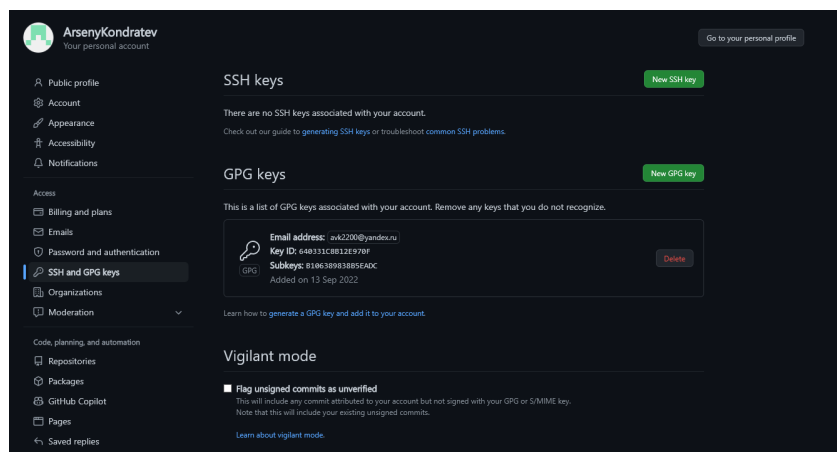


Figure 4.9: Рис. 9

9. Настроил автоматические подписи коммитов git(рис.4.10)

```

git config --global user.signingkey <PGP Fingerprint>
git config --global commit.gpgsign true
git config --global gpg.program $(which gpg2)

```

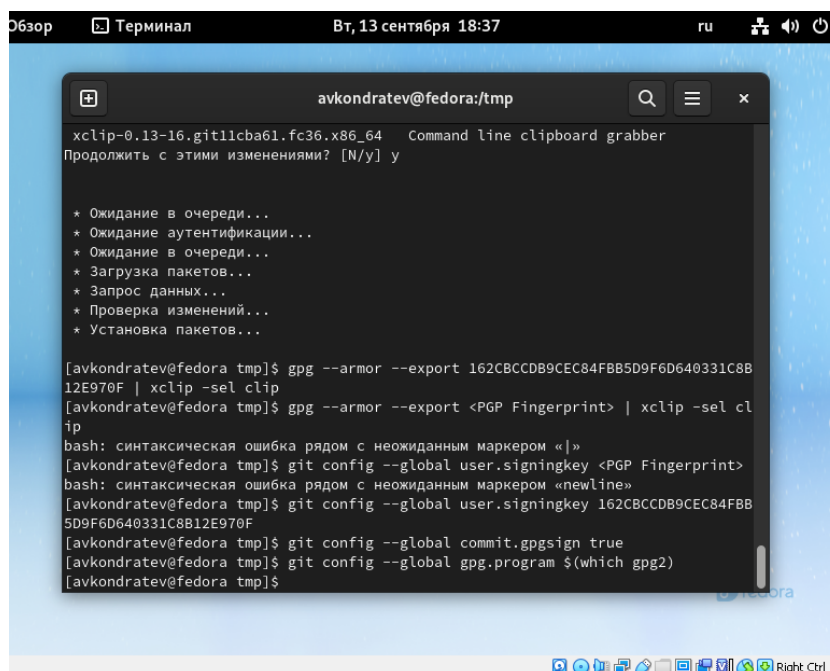


Figure 4.10: Рис. 10

10. Создал репозиторий курса на основе шаблона(рис.4.11)

```
mkdir -p ~/work/study/2021-2022/"Операционные системы"
```

```
cd ~/work/study/2021-2022/"Операционные системы"
```

```
gh repo create study_2021-2022_os-intro
```

```
❑ --template=yamadharma/course-directory-student-template --public
```

```
git clone --recursive
```

```
❑ git@github.com:ArsenyKondratev study_2021-2022_os-intro.git os-intro
```

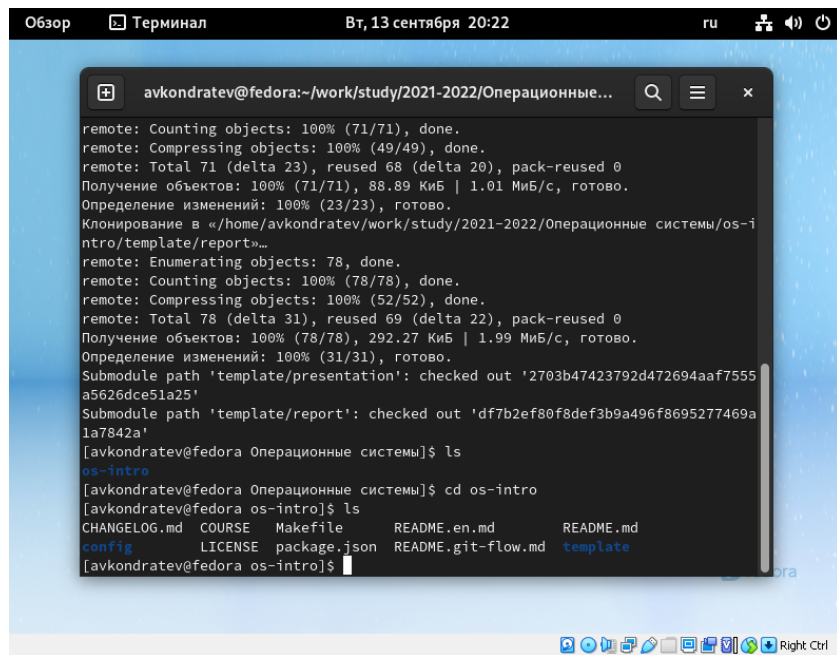


Figure 4.11: Рис. 11

11. Настроил каталог курса(рис.4.12)

Удалил лишние файлы и создал необходимый каталог

```
rm package.json
```

```
make COURSE=os-intro
```

```
Взор Терминал Вт, 13 сентября 20:48 ru

[avkondratev@fedora:~/work/study/2021-2022/Операционные...]
[avkondratev@fedora tmp]$ cat ~/.ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQCAQC7U17J/P2w/1qSh9IjiJYxjkGmyNYozcqW6tITVtaY
N4fF6n7B5eI22e9pK3MF2RC4l6FInrHmt2l3IBigZwENXPKNDwN5fy2f45Pj2BdZSGMstw7VE1a1dw3e
WYf6SrgiJLr5leRsbSHD02Hb6R+IkNiXSUCPmwNYFCLt0bbD4EVnGikHYLntDNv+hg6hLVBew0+bjid
deK5Mg2ZKxJxWHTENY5vucc5R6QVGhLL67wsBXUfIUUGYTTtyGYxqjyVAhjuV698rrURCEwr7bHs9X6c0
ZdJq+ZNjsiGCPqtVnsb77Rl4xZKIH5lpxfQMcZexs3oEDiiwTnIlefJ4UKuDfixM1hGCUtXImdiZwfkL
XlgSLebCQeGT3FcVvR54LEjLhKsNEIyDUhhbj8knTsDe0GbRvw17eDMg2EZmoTxf5lHsKWz4sFAwnJcp
lGKDPtaHwAR9G9NuxZpku8fuQckZkt0LaS1J8WPPTXKkKF39Hxh6iG73Ged5DRLVV46EhTyELDVR1sIv
QRh5hHeIZtxJc97N/vln2D59bdJJj2PyY5DjnVA5lNr18GVrIOChlpB5/0+mToVfpjEGYcrJeNzQ2VCU
JhUEv/tHzIpy70BbI3KpvTRKbIvyPKvUqPkb+eJ0PHkKpePJbVye7Fb4QG4LSTXoUP7lDwRHdo5BMWq
+w== avkondratev@fedora
[avkondratev@fedora tmp]$ cd ~/work/study/2021-2022/"Операционные системы"/os-in
tro
[avkondratev@fedora os-intro]$ git push
Перечисление объектов: 21, готово.
Подсчет объектов: 100% (21/21), готово.
Сжатие объектов: 100% (17/17), готово.
Запись объектов: 100% (20/20), 311.68 Киб | 2.21 Миб/с, готово.
Всего 20 (изменений 1), повторно использовано 0 (изменений 0), повторно использо
вано пакетов 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:ArseyKondratev/study_2021-2022_os-intro.git
554b71f..57c6e3b master -> master
[avkondratev@fedora os-intro]$
```

Figure 4.12: Рис. 12

5 Выводы

Я научился оформлять отчёты с помощью легковесного языка разметки Markdown.

6 Контрольные вопросы

1. Система контроля версий (Version Control System, VCS) — программное обеспечение для облегчения работы с изменяющейся информацией. VCS позволяет хранить несколько версий одного и того же документа, при необходимости возвращаться к более ранним версиям, определять, кто и когда сделал то или иное изменение.
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 нескольким разработчикам с удобством работать над одним проектом. Возможность получать изменения, внесенные другим человеком и откатываться на прошлые версии в случае ошибок.

7.

- a. 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)
9. Это простой перемещаемый указатель на один из таких коммитов. Они нужны для того, чтобы разделять код. Например одна ветка у нас может быть основная для разработки. Если мы делаем новый функционал, то мы создаем новую ветку под него, а после окончания работы сливаем то, что мы сделали в основную ветку.
10. Во время работы могут появляться временные файлы, не несущие смысла для проекта. Их лучше не отправлять при использовании `commit`.