

Отчет по лабораторной работе №2

Управление версиями

Кочкарев “sakochkarev” Станислав

Содержание

1	Цель работы	3
2	Задание	4
3	Выполнение лабораторной работы	5
4	Выводы	14
5	Ответы на контрольные вопросы	15

1 Цель работы

Изучение идеологии и применение средств контроля версий. Освоение умения по работе с Git.

2 Задание

- Создать базовую конфигурацию для работы с git.
- Создать ключ SSH.
- Создать ключ PGP.
- Настроить подписи git.
- Зарегистрироваться на GitHub.
- Создать локальный каталог для выполнения заданий по предмету.

3 Выполнение лабораторной работы

В пунктах 2.1 по 2.5 были изучены азы системы контроля версий Git, а также команды, используемые для работы.

В пункте 5 был основной процесс выполнения лабораторной работы №2.

Предварительно был создан аккаунт на GitHub (рис. 3.1).

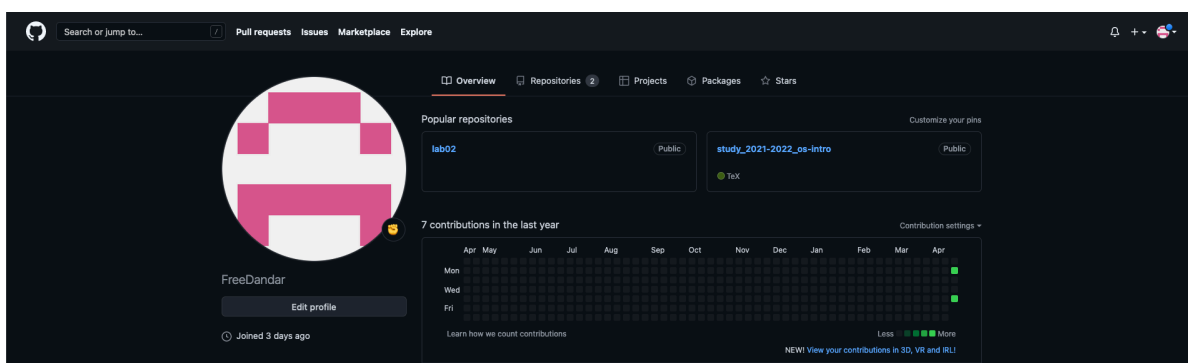


Рис. 3.1: Созданный аккаунт на GitHub

Также предварительно были скачены утилиты git, такие как git и gh. В процессе работы над лабораторной работой была также установлена утилита git-flow (рис. 3.2).

```
os-intro — squidass@sqmac — .темы/os-intro — zsh — 130x30
squidass@sqmac [17:52:38] [~/work/study/2021-2022/course-directory-student-template] [master]
-> % brew install git-flow
Running `brew update --preinstall`...
=> Auto-updated Homebrew!
Updated 6 taps (microsoft/git, homebrew/cask-versions, homebrew/core, homebrew/cask, homebrew/services and homebrew/cask-drivers).
=> New Formulae
sophus
=> Updated Formulae
Updated 104 formulae.
=> New Casks
descript
=> Updated Casks
Updated 62 casks.

=> Downloading https://ghcr.io/v2/homebrew/core/git-flow/manifests/0.4.1_1
##### 100.0%
=> Downloading https://ghcr.io/v2/homebrew/core/git-flow/blobs/sha256:cffa267a59238174b54b4058131b3fdf674d4fa79ff724dd7111f6bc773
=> Downloading from https://pkg-containers.githubusercontent.com/ghcr1/blobs/sha256:cffa267a59238174b54b4058131b3fdf674d4fa79ff72
##### 100.0%
=> Pouring git-flow--0.4.1_1.all.bottle.tar.gz
=> Caveats
To install Zsh completions:
  brew install zsh-completions
=> Summary
📦 /usr/local/Cellar/git-flow/0.4.1_1: 17 files, 112KB
=> Running `brew cleanup git-flow`...
Disable this behaviour by setting HOMEBREW_NO_INSTALL_CLEANUP.
Hide these hints with HOMEBREW_NO_ENV_HINTS (see `man brew`).
squidass@sqmac [17:56:32] [~/work/study/2021-2022/course-directory-student-template] [master]
-> % clear
```

Рис. 3.2: Установка утилиты git-flow

Далее была проведена базовая конфигурация утилиты git путем введения команд конфигурации (рис. 3.3, 3.4).

```
squidass@sqmac [18:33:31] [~/tmp/lab02] [master *]
-> % git config user.name "Кочкарев Станислав"
squidass@sqmac [18:33:54] [~/tmp/lab02] [master *]
-> % git config user.email "1032219994@pfur.ru"
```

Рис. 3.3: Установка имени и адреса почты

```
squidass@sqmac [18:43:10] [~/tmp/lab02] [hotfix/hotfix_branch *]
-> % git config init.defaultBranch master
squidass@sqmac [18:44:15] [~/tmp/lab02] [hotfix/hotfix_branch *]
-> % git config core.autocrlf input
squidass@sqmac [18:44:24] [~/tmp/lab02] [hotfix/hotfix_branch *]
-> % git config core.safecrlf warn
```

Рис. 3.4: Установка дополнительных конфигураций

После этого был создан ssh ключ (рис. 3.5) и добавлен в GitHub (рис. 3.6).

```
squidass@sqmac [18:20:10] [~/tmp/lab02] [master *]
-> % ssh-keygen -f ~/.ssh/id_rsa_rudn
Generating public/private rsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /Users/squidass/.ssh/id_rsa_rudn
Your public key has been saved in /Users/squidass/.ssh/id_rsa_rudn.pub
The key fingerprint is:
SHA256:nldumB4AGhJYiPn+PGu6mJ00J3qUWen8P0gpyY1gWAs squidass@sqmac
The key's randomart image is:
+---[RSA 3072]-----+
|.o                    |
|= .                   |
|.. ...               |
|E o.o. .             |
|. + B.   S   .       |
|o.* o   . o =        |
|++.* +   o = o       |
|.B*.B.+   o o        |
|+o**+o   . .         |
+----[SHA256]-----+
```

Рис. 3.5: Создание ssh ключа

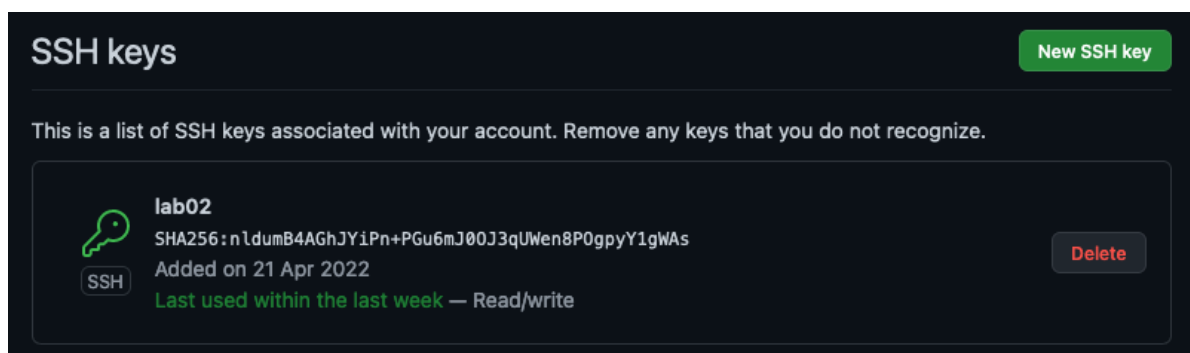


Рис. 3.6: Добавленный ssh ключ в GitHub

Следующим шагом было создание пары ключей gpg (рис. 3.7).

```
os-intro — squidass@sqmac — .темы/os-intro — -zsh — 130x58
squidass@sqmac [18:45:04] [~/tmp/lab02] [hotfix/hotfix_branch *]
-> % gpg --full-generate-key
gpg (GnuPG) 2.3.4; Copyright (C) 2021 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

gpg: directory '/Users/squidass/.gnupg' created
gpg: keybox '/Users/squidass/.gnupg/pubring.kbx' created
Please select what kind of key you want:
(1) RSA and RSA
(2) DSA and Elgamal
(3) DSA (sign only)
(4) RSA (sign only)
(9) ECC (sign and encrypt) *default*
(10) ECC (sign only)
(14) Existing key from card
Your selection? 1
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (3072) 4096
Requested keysize is 4096 bits
Please specify how long the key should be valid.
  0 = key does not expire
  <n> = key expires in n days
  <n>w = key expires in n weeks
  <n>m = key expires in n months
  <n>y = key expires in n years
Key is valid for? (0) 0
Key does not expire at all
Is this correct? (y/N) y

GnuPG needs to construct a user ID to identify your key.

Real name: sakochkarev
Email address: 1032219994@pfur.ru
Comment:
You selected this USER-ID:
"sakochkarev <1032219994@pfur.ru>"

Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit?
Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? o
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
gpg: /Users/squidass/.gnupg/trustdb.gpg: trustdb created
gpg: directory '/Users/squidass/.gnupg/openpgp-revocs.d' created
gpg: revocation certificate stored as '/Users/squidass/.gnupg/openpgp-revocs.d/ABEE67EA63924AE215EFB3CE6CD06708CA856A26.rev'
public and secret key created and signed.

pub   rsa4096 2022-04-21 [SC]
       ABEE67EA63924AE215EFB3CE6CD06708CA856A26
uid     sakochkarev <1032219994@pfur.ru>
sub   rsa4096 2022-04-21 [E]
```

Рис. 3.7: Создание пары GPG ключей

После этого созданный ключ был добавлен в Github (рис. 3.8, 3.9).


```
os-intro — squidass@sqmac — ..темы/os-intro — -zsh — 130x27
squidass@sqmac [18:48:22] [~/tmp/lab02] [hotfix/hotfix_branch *]
-> % gpg --list-secret-keys
gpg: checking the trustdb
gpg: marginals needed: 3 completes needed: 1 trust model: pgp
gpg: depth: 0 valid: 1 signed: 0 trust: 0-, 0q, 0n, 0m, 0f, 1u
/Users/squidass/.gnupg/pubring.kbx
-----
sec   rsa4096 2022-04-21 [SC]
      ABEE67EA63924AE215EFB3CE6CD06708CA856A26
uid    [ultimate] sakochkarev <1032219994@pfur.ru>
ssb    rsa4096 2022-04-21 [E]

squidass@sqmac [18:48:35] [~/tmp/lab02] [hotfix/hotfix_branch *]
-> %
squidass@sqmac [18:48:36] [~/tmp/lab02] [hotfix/hotfix_branch *]
-> % gpg --list-secret-keys --keyid-format LONG
/Users/squidass/.gnupg/pubring.kbx
-----
sec   rsa4096/6CD06708CA856A26 2022-04-21 [SC]
      ABEE67EA63924AE215EFB3CE6CD06708CA856A26
uid    [ultimate] sakochkarev <1032219994@pfur.ru>
ssb    rsa4096/99C9D8901A28F25D 2022-04-21 [E]

squidass@sqmac [18:48:48] [~/tmp/lab02] [hotfix/hotfix_branch *]
-> % gpg --armor --export 6CD06708CA856A26
-----BEGIN PGP PUBLIC KEY BLOCK-----
```

Рис. 3.8: Экспорт GPG ключа

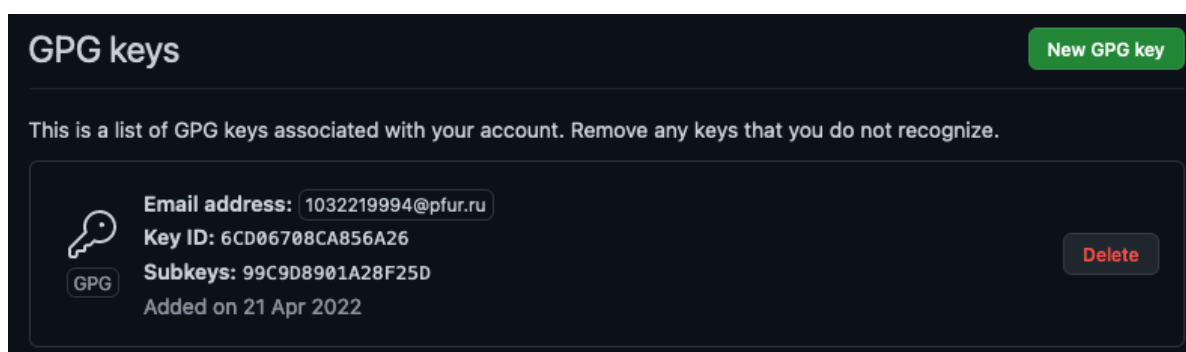


Рис. 3.9: Добавленный в GitHub GPG ключ

Конечным шагом после добавления GPG ключа в Github была дополнительная конфигурация git, чтобы все коммиты автоматически подписывались данным ключом (рис. 3.10).

```

squidass@sqmac [18:49:29] [~/tmp/lab02] [hotfix/hotfix_branch *]
-> % git config user.signingkey 6CD06708CA856A26
squidass@sqmac [18:50:27] [~/tmp/lab02] [hotfix/hotfix_branch *]
-> % git config commit.gpgsign true
squidass@sqmac [18:51:22] [~/tmp/lab02] [hotfix/hotfix_branch *]
-> % git config gpg.program /usr/local/bin/gpg

```

Рис. 3.10: Дополнительная конфигурация git для работы с GPG

После того, как все предварительные настройки были завершены, я перешел к созданию репозитория курса на основе шаблона.

Для этого были созданы рабочие директории (рис. 3.11).

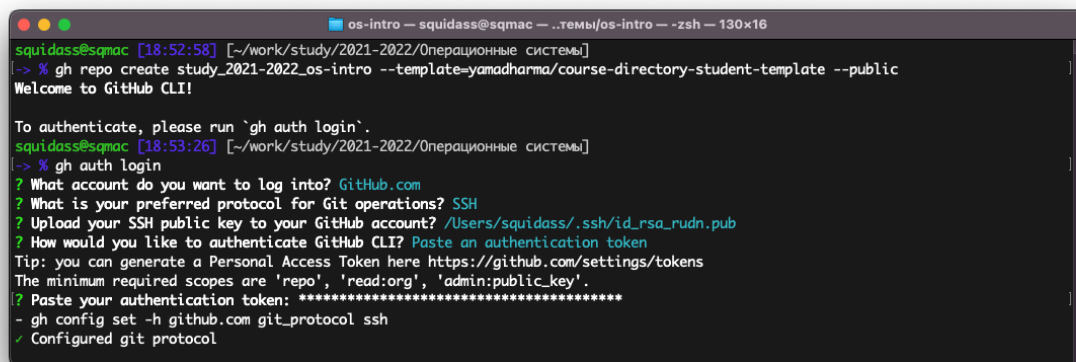
```

os-intro — squidass@sqmac — ../темы/os-intro — zsh — 130x16
squidass@sqmac [18:51:32] [~/tmp/lab02] [hotfix/hotfix_branch *]
-> % cd ~
squidass@sqmac [18:52:08] [~]
-> % cd work
squidass@sqmac [18:52:11] [~/work]
-> % cd study/2021-2022
squidass@sqmac [18:52:17] [~/work/study/2021-2022]
-> % ls
total 0
drwxr-xr-x  3 squidass  staff   96B Apr 19 22:30 .
drwxr-xr-x  3 squidass  staff   96B Apr 19 22:29 ..
drwxr-xr-x 14 squidass  staff  448B Apr 19 22:30 course-directory-student-template
squidass@sqmac [18:52:18] [~/work/study/2021-2022]
-> % mkdir "Операционные системы"
squidass@sqmac [18:52:33] [~/work/study/2021-2022]
-> % cd Операционные\ системы

```

Рис. 3.11: Рабочие директории

Далее было необходимо создать репозиторий используя приведенный репозиторий как шаблон, однако перед этим было необходимо дать доступ к Github утилите gh (рис. 3.12).

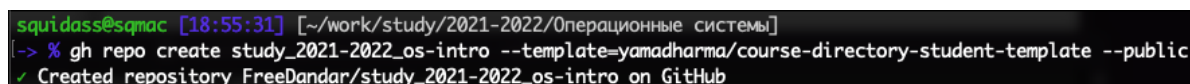
A terminal window titled 'os-intro — squidass@sqmac — ..темы/os-intro — -zsh — 130x16'. The prompt is 'squidass@sqmac [18:52:58] [~/work/study/2021-2022/Операционные системы]'. The user runs 'gh repo create study_2021-2022_os-intro --template=yamadharma/course-directory-student-template --public'. The terminal shows the GitHub CLI welcome message and prompts for authentication. The user runs 'gh auth login'. The terminal shows prompts for account selection, protocol selection (SSH), and SSH key upload. The user pastes an authentication token. The terminal shows the token being set and the git protocol being configured. The final output is 'Configured git protocol'.

```
os-intro — squidass@sqmac — ..темы/os-intro — -zsh — 130x16
squidass@sqmac [18:52:58] [~/work/study/2021-2022/Операционные системы]
-> % gh repo create study_2021-2022_os-intro --template=yamadharma/course-directory-student-template --public
Welcome to GitHub CLI!

To authenticate, please run `gh auth login`.
squidass@sqmac [18:53:26] [~/work/study/2021-2022/Операционные системы]
-> % gh auth login
? What account do you want to log into? GitHub.com
? What is your preferred protocol for Git operations? SSH
? Upload your SSH public key to your GitHub account? /Users/squidass/.ssh/id_rsa_rudn.pub
? How would you like to authenticate GitHub CLI? Paste an authentication token
Tip: you can generate a Personal Access Token here https://github.com/settings/tokens
The minimum required scopes are 'repo', 'read:org', 'admin:public_key'.
? Paste your authentication token: *****
- gh config set -h github.com git_protocol ssh
✓ Configured git protocol
```

Рис. 3.12: Выдача доступа к аккаунту утилите gh

После этого был скопирован и создан на примере приведенного репозитория собственный репозиторий для дальнейшей работы (рис. 3.13).

A terminal window showing the command to create a new repository. The prompt is 'squidass@sqmac [18:55:31] [~/work/study/2021-2022/Операционные системы]'. The user runs 'gh repo create study_2021-2022_os-intro --template=yamadharma/course-directory-student-template --public'. The terminal shows the command being executed and the repository being created. The final output is 'Created repository FreeDandar/study_2021-2022_os-intro on GitHub'.

```
squidass@sqmac [18:55:31] [~/work/study/2021-2022/Операционные системы]
-> % gh repo create study_2021-2022_os-intro --template=yamadharma/course-directory-student-template --public
✓ Created repository FreeDandar/study_2021-2022_os-intro on GitHub
```

Рис. 3.13: Создание репозитория на примере приведенного

Далее созданный репозиторий был скопирован на локальную машину (рис. 3.14).

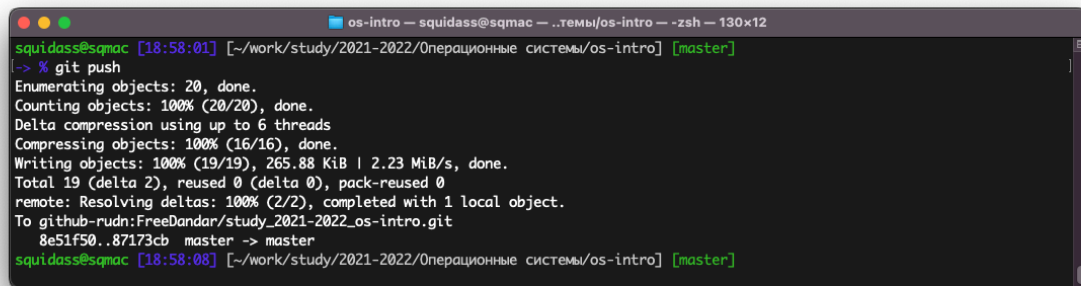
```
os-intro — squidass@sqmac — ..темы/os-intro — -zsh — 130x29
squidass@sqmac [18:55:37] [~/work/study/2021-2022/Операционные системы]
-> % git clone --recursive git@github-rudn:FreeDandar/study_2021-2022_os-intro.git os-intro
Cloning into 'os-intro'...
remote: Enumerating objects: 20, done.
remote: Counting objects: 100% (20/20), done.
remote: Compressing objects: 100% (18/18), done.
remote: Total 20 (delta 2), reused 15 (delta 2), pack-reused 0
Receiving objects: 100% (20/20), 12.49 KiB | 12.49 MiB/s, done.
Resolving deltas: 100% (2/2), done.
Submodule 'template/presentation' (https://github.com/yamadharma/academic-presentation-markdown-template.git) registered for path
'template/presentation'
Submodule 'template/report' (https://github.com/yamadharma/academic-laboratory-report-template.git) registered for path 'template/
report'
Cloning into '/Users/squidass/work/study/2021-2022/Операционные системы/os-intro/template/presentation'...
remote: Enumerating objects: 42, done.
remote: Counting objects: 100% (42/42), done.
remote: Compressing objects: 100% (34/34), done.
remote: Total 42 (delta 9), reused 40 (delta 7), pack-reused 0
Receiving objects: 100% (42/42), 31.19 KiB | 1.08 MiB/s, done.
Resolving deltas: 100% (9/9), done.
Cloning into '/Users/squidass/work/study/2021-2022/Операционные системы/os-intro/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
Receiving objects: 100% (78/78), 292.27 KiB | 2.16 MiB/s, done.
Resolving deltas: 100% (31/31), done.
Submodule path 'template/presentation': checked out '3eae6b7586f8a9aded2b506cd1018e625b228b93'
Submodule path 'template/report': checked out 'df7b2ef80f8def3b9a496f8695277469a1a7842a'
```

Рис. 3.14: Копирование репозитория на локальную машину

После была проведена процедура настройки каталога курса. Был удален лиш-
ний файл, далее с помощью приведенного документа были созданы дополни-
тельные каталоги и после этого все файлы были отправлены на сервер (рис. 3.15,
3.16).

```
os-intro — squidass@sqmac — ..темы/os-intro — -zsh — 130x16
squidass@sqmac [18:57:05] [~/work/study/2021-2022]
-> % cd Операционные\ системы
squidass@sqmac [18:57:20] [~/work/study/2021-2022/Операционные системы]
-> % ls
total 0
drwxr-xr-x  3 squidass  staff   96B Apr 21 18:56 .
drwxr-xr-x  4 squidass  staff  128B Apr 21 18:52 ..
drwxr-xr-x 14 squidass  staff  448B Apr 21 18:56 os-intro
squidass@sqmac [18:57:21] [~/work/study/2021-2022/Операционные системы]
-> % cd os-intro
squidass@sqmac [18:57:23] [~/work/study/2021-2022/Операционные системы/os-intro] [master]
-> % rm package.json
squidass@sqmac [18:57:28] [~/work/study/2021-2022/Операционные системы/os-intro] [master *]
-> % make COURSE=os-intro
squidass@sqmac [18:57:37] [~/work/study/2021-2022/Операционные системы/os-intro] [master *]
-> % git add .
```

Рис. 3.15: Процедура настройки каталога

A terminal window titled "os-intro — squidass@sqmac — ..темы/os-intro — -zsh — 130x12". The prompt is "squidass@sqmac [18:58:01] [~/work/study/2021-2022/Операционные системы/os-intro] [master]". The user enters "git push". The output shows the progress of pushing to the remote repository: "Enumerating objects: 20, done.", "Counting objects: 100% (20/20), done.", "Delta compression using up to 6 threads", "Compressing objects: 100% (16/16), done.", "Writing objects: 100% (19/19), 265.88 KiB | 2.23 MiB/s, done.", "Total 19 (delta 2), reused 0 (delta 0), pack-reused 0", "remote: Resolving deltas: 100% (2/2), completed with 1 local object.", "To github-rudn:FreeDandar/study_2021-2022_os-intro.git", "8e51f50..87173cb master -> master". The prompt returns to "squidass@sqmac [18:58:08] [~/work/study/2021-2022/Операционные системы/os-intro] [master]".

```
os-intro — squidass@sqmac — ..темы/os-intro — -zsh — 130x12
squidass@sqmac [18:58:01] [~/work/study/2021-2022/Операционные системы/os-intro] [master]
-> % git push
Enumerating objects: 20, done.
Counting objects: 100% (20/20), done.
Delta compression using up to 6 threads
Compressing objects: 100% (16/16), done.
Writing objects: 100% (19/19), 265.88 KiB | 2.23 MiB/s, done.
Total 19 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), completed with 1 local object.
To github-rudn:FreeDandar/study_2021-2022_os-intro.git
8e51f50..87173cb master -> master
squidass@sqmac [18:58:08] [~/work/study/2021-2022/Операционные системы/os-intro] [master]
```

Рис. 3.16: Отправка созданного финального каталога на сервер

4 Выводы

Была изучена идеология и применение средств контроля версий. Было освоено умение по работе с git. Дополнительно мы познакомились с платформой Github, а также в некоторых моментах была изучена дополнительная информация для решения встретившихся проблем.

5 Ответы на контрольные вопросы

1. Системы контроля версий — это набор программного обеспечения, которые предназначены для работы нескольких человек над одним проектом.
2.
 - Хранилище — сервер, на котором хранится вся история изменений проекта.
 - Commit — фиксация “дельта-изменений”, т.е. изменений с последнего commit’a с его последующей записью как версии в истории.
 - История — список всех изменений проекта с возможностью отката в любую точку истории.
 - Рабочая копия — все файлы проекта, с которыми происходит основная работа.
3. В централизованных VCS необходим центральный репозиторий для хранения файлов. Примером таковых могут служить CVS и Subversion. В децентрализованных VCS наличие центрального репозитория не обязательно. Децентрализованными VCS являются Git, Bazaar и Mercurial.
4. Инициализация системы управления версиями git через git init. Работа над проектом используя git-flow для отдельных частей проекта. Git commit для фиксации изменений. При необходимости использование удаленного сервера для хранения с помощью remote и git push. Удаленный сервер также позволяет работать с нескольких устройств с использованием git pull.
5. При существующей версии проекта в хранилище, скопировать его оттуда

- через `git pull`. Использовать `git-flow` для работы над частями проекта. После окончания работы зафиксировать изменения через `git commit` и загрузить в хранилище через `git push`.
6. Ведение истории изменений, фиксирование изменений, совмещение версий, веток и др., а также откат к прошлым версиям.
 7.
 - `git init` — инициализация проекта с системой контроля версий
 - `git add` — добавление файла/директории в систему контроля версий как отслеживаемое
 - `git commit` — фиксация изменений в отслеживаемых файлах
 - `git push` — загрузка локальной версии на сервер
 - `git pull` — выгрузка актуальной версии с сервера
 - `git fetch` — “часть” команды `git pull`, которая собирает актуальную версию, но не вносит её в работу
 - `git merge` — слияние веток
 8. При работе с локальным репозиторием все изменения хранятся локально и не выгружаются на удаленный сервер. Не требуется использование команд `push`, `pull`, `remote` и т.д. При работе с удаленным репозиторием для отображения изменения на удаленном репозитории и его актуализации, последние изменения должны быть загружены на удаленный сервер.
 9. Ветви позволяют “разделять” части работы и работать отдельно над каждой имплементацией. Использование ветвей дает возможность комфортной ревизии и обработки нововведений в основную ветвь, которая чаще всего является релизной.
 10. Игнорирование файлов при `commit` происходит с помощью `.gitignore` файла. В нем указываются пути, названия, расширения и другие идентификаторы нежелательных объектов которые не будут учитываться в `commit`. Это полезно для исключения как “мусорных” файлов, которые не являют-

ся значимой частью проекта, а также конфиденциальных файлов, которые содержат в себе приватную информацию, такую как пароли и токены.