

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ № 2.

дисциплина: Архитектура компьютера

Студент: Иванова Анастасия

Группа: НКАбд-07-25

МОСКВА

2025 г.

Содержание

1. Цель работы	3
2. Теоретическое введение.....	4
2.1. Системы контроля версий	4
2.2. Системы контроля версий Git	4
2.3. Основные команды git	6
3. Порядок выполнения работы	10
3.1. Настройка github	10
3.2. Базовая настройка git	10
3.3. Создание SSH-ключа	15
3.4. Создание рабочего пространства и репозитория курса	17
3.5. Создание репозитория курса.....	20
3.6. Настройка каталога курса.....	21
4. Задание для самостоятельной работы	25
5. Заключение.....	28

1. Цель работы

Целью работы является изучение идеологии и применения средств контроля версий, приобретение практических навыков по работе с системой контроля версий git.

2. Теоретическое введение

2.1. Системы контроля версий.

Общие понятия Системы контроля версий (Version Control System, VCS) применяются при работе нескольких человек над одним проектом. Обычно основное дерево проекта хранится в локальном или удалённом репозитории, к которому настроен доступ для участников проекта. При внесении изменений в содержание проекта система контроля версий позволяет их фиксировать, совмещать изменения, произведённые разными участниками проекта, производить откат к любой более ранней версии проекта, если это требуется.

В классических системах контроля версий используется централизованная модель, предполагающая наличие единого репозитория для хранения файлов. Выполнение большинства функций по управлению версиями осуществляется специальным сервером. Участник проекта (пользователь) перед началом работы посредством определённых команд получает нужную ему версию файлов. После внесения изменений пользователь размещает новую версию в хранилище. При этом предыдущие версии не удаляются из центрального хранилища и к ним можно вернуться в любой момент. Сервер может сохранять не полную версию изменённых файлов, а производить так называемую дельта-компрессию — сохранять только изменения между последовательными версиями, что позволяет уменьшить объём хранимых данных.

Системы контроля версий поддерживают возможность отслеживания и разрешения конфликтов, которые могут возникнуть при работе нескольких человек над одним файлом. Можно объединить (слить) изменения, сделанные разными участниками (автоматически или вручную), вручную выбрать нужную версию, отменить изменения вовсе или заблокировать файлы для изменения. В зависимости от настроек блокировка не позволяет другим пользователям получить рабочую копию или препятствует изменению рабочей копии файла средствами файловой системы ОС, обеспечивая, таким образом, привилегированный доступ только одному пользователю, работающему с файлом.

Системы контроля версий также могут обеспечивать дополнительные, более гибкие функциональные возможности. Например, они могут поддерживать работу с несколькими версиями одного файла, сохраняя общую историю изменений до точки ветвления версий и собственные истории изменений каждой ветви. Кроме того, обычно доступна информация о том, кто из участников, когда и какие изменения вносил. Обычно такого рода информация хранится в журнале изменений, доступ к которому можно ограничить.

В отличие от классических в распределённых системах контроля версий центральный репозиторий не является обязательным. Среди классических VCS

наиболее известны CVS, Subversion, а среди распределённых — Git, Bazaar, Mercurial. Принципы их работы схожи, отличаются они в основном синтаксисом используемых в работе команд.

2.2. Система контроля версий Git

Система контроля версий Git представляет собой набор программ командной строки. Доступ к ним можно получить из терминала посредством ввода команды git с различными опциями. Благодаря тому, что Git является распределённой системой контроля версий, резервную копию локального хранилища можно сделать простым копированием или архивацией.

2.3. Основные команды git

Наиболее часто используемые команды git представлены в таблице.

Основные команды git

Команда	Описание
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

3. Порядок выполнения работы

3.1. Настройка github

Существует несколько доступных серверов репозитория с возможностью бесплатного размещения данных. Например: <http://bitbucket.org/>, <https://github.com/> и <https://gitflic.ru>. Для выполнения лабораторных работ предлагается использовать Github. Создайте учётную запись на сайте <https://github.com/> и заполните основные данные (Рисунок 3.1).

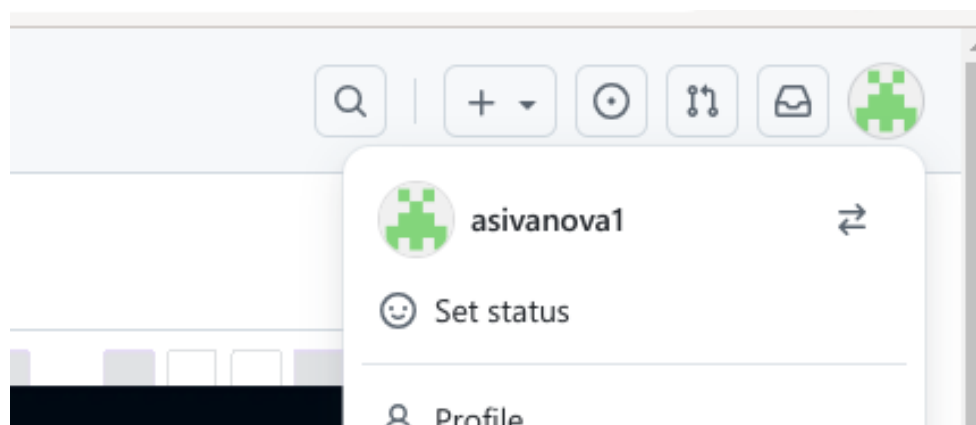


Рисунок 3.1. - Создание профиля в github

3.2. Базовая настройка git

Сначала сделаем предварительную конфигурацию git. Откроем терминал и введем следующие команды, указав имя и e-mail владельца репозитория (Рисунок 3.2):

```
git config --global user.name "<Name Surname>"
git config --global user.email "<work@mail>"
```

Настроим utf-8 в выводе сообщений git:

```
git config --global core.quotePath false
```

Зададим имя начальной ветки (будем называть её master):

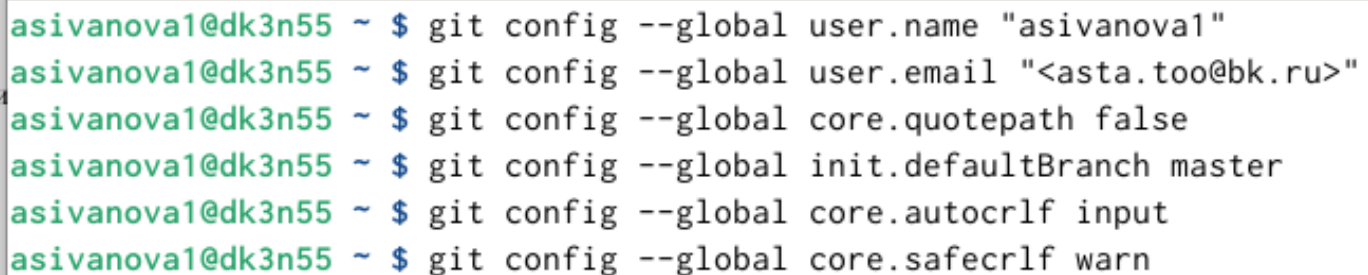
```
git config --global init.defaultBranch master
```

Параметр autocrlf:

```
git config --global core.autocrlf input
```

Параметр safecrlf:

```
git config --global core.safecrlf warn
```

A screenshot of a terminal window showing a series of git configuration commands being executed. The prompt is 'asivanova1@dk3n55 ~ \$'. The commands are: 'git config --global user.name "asivanova1"', 'git config --global user.email "<asta.too@bk.ru>"', 'git config --global core.quotePath false', 'git config --global init.defaultBranch master', 'git config --global core.autocrlf input', and 'git config --global core.safecrlf warn'. Each command is preceded by the prompt and followed by a tilde '~' and a dollar sign '\$' on the same line.

```
asivanova1@dk3n55 ~ $ git config --global user.name "asivanova1"
asivanova1@dk3n55 ~ $ git config --global user.email "<asta.too@bk.ru>"
asivanova1@dk3n55 ~ $ git config --global core.quotePath false
asivanova1@dk3n55 ~ $ git config --global init.defaultBranch master
asivanova1@dk3n55 ~ $ git config --global core.autocrlf input
asivanova1@dk3n55 ~ $ git config --global core.safecrlf warn
```

Рисунок 3.2. — Предварительная настройка git

3.3. Создание SSH-ключа

Для последующей идентификации пользователя на сервере репозитория сгенерируем ключ (Рисунок 3.3):

```

asivanova1@dk3n55 ~ $ ssh-keygen -C "Anastasia Ivanova<asta.too@bk.ru>"
Generating public/private ed25519 key pair.
Enter file in which to save the key (/afs/.dk.sci.pfu.edu.ru/home/a/s/asivanova1/.ssh/id_ed25519):
/afs/.dk.sci.pfu.edu.ru/home/a/s/asivanova1/.ssh/id_ed25519 already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /afs/.dk.sci.pfu.edu.ru/home/a/s/asivanova1/.ssh/id_ed25519
Your public key has been saved in /afs/.dk.sci.pfu.edu.ru/home/a/s/asivanova1/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:e/OEN84JBkJ7KUw4SxaVESYIsj8lebE/ha/xzEMlTsU Anastasia Ivanova<asta.too@bk.ru>
The key's randomart image is:
+--[ED25519 256]--+
|o. .+. =+ ..      |
|.... B.. .E       |
|. o O + + .       |
| . * B * +        |
| o . O S          |
| . % o .          |
| . * * +          |
|      + O o       |
|      =          |
+-----[SHA256]-----+

```

Рисунок 3.3. – Создание ключа

Далее нам необходимо загрузить сгенерированный ключ. Для этого зайдём на сайт <http://github.org/> под своей учётной записью и перейдём в меню **Setting** . После этого выберем в боковом меню **SSH and GPG keys** и нажмём кнопку **New SSH key** . Копируем из локальной консоли ключ в буфер обмена, и вставляем ключ в появившееся на сайте поле, и указываем для ключа имя (key) (Рисунок 3.4):

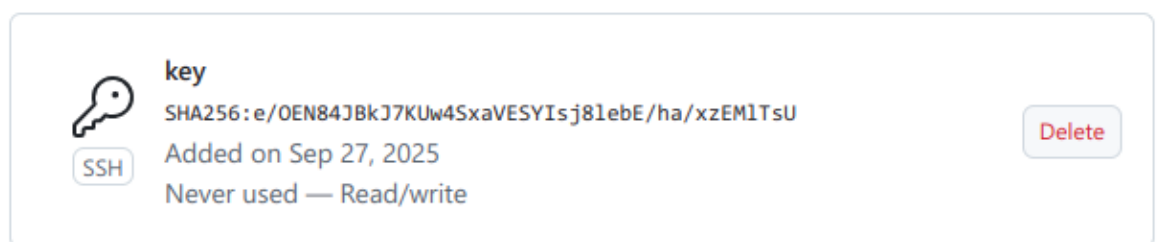


Рисунок 3.4. - Ключ

3.4. Создание рабочего пространства и репозитория курса

При выполнении лабораторных работ следует придерживаться структуры рабочего пространства. Откроем терминал и создадим каталог для предмета

«Архитектура компьютера» (Рисунок 3.5:

```
asivanova1@dk3n55 ~ $ mkdir -p ~/work/study/2025-2026/"Архитектура компьютера"
```

Рисунок 3.5. — Создание каталога для «Архитектура компьютера»

3.5. Создание репозитория курса

Репозиторий на основе шаблона можно создать через web-интерфейс github. Перейдем на страницу репозитория с шаблоном курса <https://github.com/yamadharm/course-directory-student-template> и далее выберите Use this template (Рисунок 3.6):

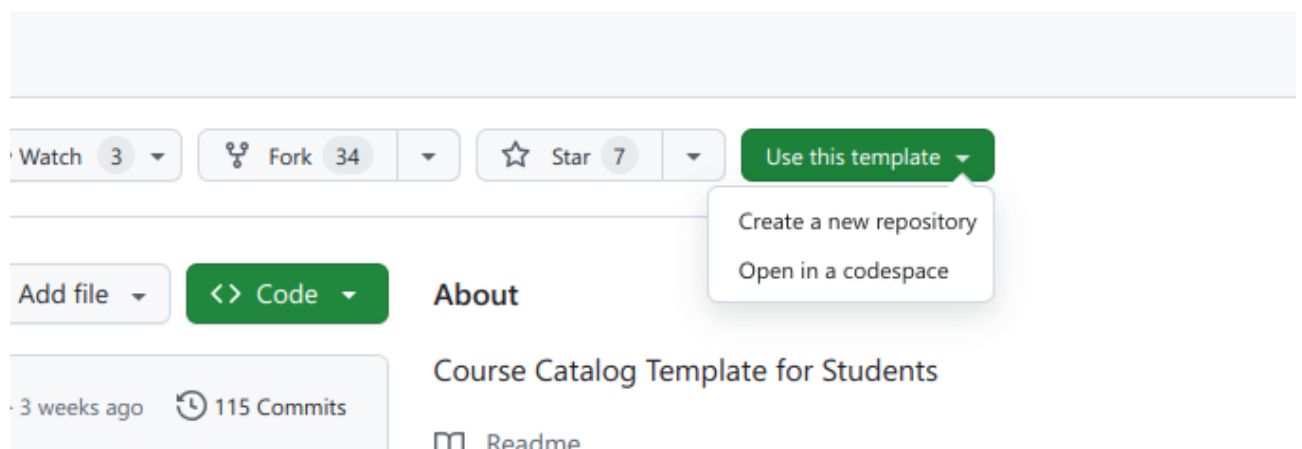


Рисунок 3.6. – Use this template

В открывшемся окне зададим имя репозитория (Repository name) study_2025–2026_arh-рс и создадим репозиторий (кнопка Create repository from template) (Рисунок 3.7):

Start with a template

Templates pre-configure your repository with files.

yamadharma/course-directory-student-template

Include all branches

Off

If enabled, all branches from the template repository will be included.

1

General

Owner *

asivanova1

Repository name *

study_2025-2026_arh-pc.git

Your new repository will be created as study_2025-2026_arh-pc.

✓

The repository name can only contain ASCII letters, digits, and the characters ., -, and _.

The .git extension will be removed.

Great repository names are short and memorable. How about jubilant-octo-funicular?

Description

0 / 350 characters

2

Configuration

Choose visibility *

Public

Choose who can see and commit to this repository

Рисунок 3.7. – Создание репозитория

Ссылку для клонирования можно скопировать на странице созданного репозитория Code -> SSH(Рисунок 3.8):

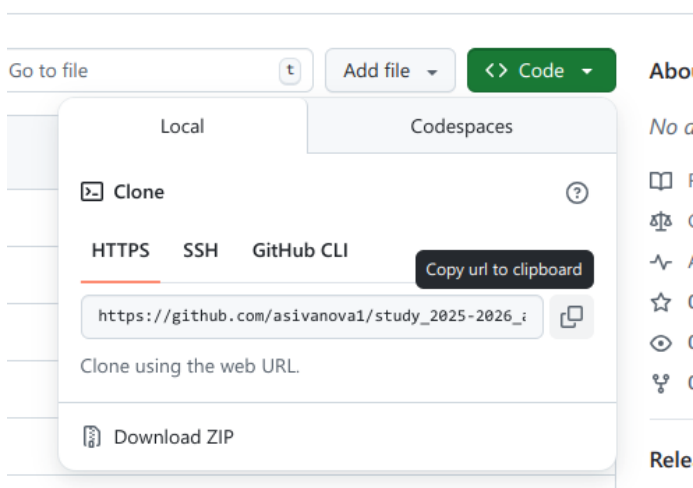


Рисунок 3.8. – Копирование репозитория

Клонируем созданный репозиторий (Рисунок 3.9 – 3.10)

```

asivanova1@dk3n55 ~/work/study/2025-2026/Архитектура компьютера $ git clone --recursive git@github.com:asivanova1/study_2025-2026_arh-pc.git arch-pc
Клонирование в «arch-pc»...
The authenticity of host 'github.com (140.82.121.3)' can't be established.
ED25519 key fingerprint is SHA256:+DiY3wvvV6TuJJhbpZisF/zLDA0zPMSvHdkr4UvCQqU.
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: 38, done.
remote: Counting objects: 100% (38/38), done.
remote: Compressing objects: 100% (36/36), done.
remote: Total 38 (delta 1), reused 26 (delta 1), pack-reused 0 (from 0)
Получение объектов: 100% (38/38), 23.45 КиБ | 585.00 КиБ/с, готово.
Определение изменений: 100% (1/1), 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 «/afs/.dk.sci.pfu.edu.ru/home/a/s/asivanova1/work/study/2025-2026/Архитектура компьютера/arch-pc/template/presentation»...
remote: Enumerating objects: 161, done.
remote: Counting objects: 100% (161/161), done.
remote: Compressing objects: 100% (111/111), done.
remote: Total 161 (delta 60), reused 142 (delta 41), pack-reused 0 (from 0)
Получение объектов: 100% (161/161), 2.65 МиБ | 7.97 МиБ/с, готово.
Определение изменений: 100% (60/60), готово.
Клонирование в «/afs/.dk.sci.pfu.edu.ru/home/a/s/asivanova1/work/study/2025-2026/Архитектура компьютера/arch-pc/template/report»...
remote: Enumerating objects: 221, done.
remote: Counting objects: 100% (221/221), done.
remote: Compressing objects: 100% (152/152), done.
remote: Total 221 (delta 98), reused 180 (delta 57), pack-reused 0 (from 0)
Получение объектов: 100% (221/221), 765.46 КиБ | 4.01 МиБ/с, готово.
Определение изменений: 100% (98/98), готово.
Submodule path 'template/presentation': checked out '6efd5c4ee78e4456caff3dc7062cfcad26058ca6'
Submodule path 'template/report': checked out '89a9622199b4df88227b9b3fa3d4714c85f68dd2'

```

Рисунок 3.9. – Клонирование репозитория

```

asivanova1@dk3n55 ~ $ cd ~/work/study/2025-2026/"Архитектура компьютера"
asivanova1@dk3n55 ~/work/study/2025-2026/Архитектура компьютера $

```

Рисунок 3.10. – Переход в каталог курса

3.6. Настройка каталога курса

Выполним следующие действия (Рисунок 3.11):

- 1) перейдем в каталог курса и создадим необходимые каталоги:

```

echo arch-pc > COURSE
make prepare

```

- 2) отправим файлы на сервер:

```

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

```

```

asivanova1@dk3n55 ~/work/study/2025-2026/Архитектура компьютера/arch-pc $ echo arch-pc > COURSE
asivanova1@dk3n55 ~/work/study/2025-2026/Архитектура компьютера/arch-pc $ make prepare
asivanova1@dk3n55 ~/work/study/2025-2026/Архитектура компьютера/arch-pc $ git add .
asivanova1@dk3n55 ~/work/study/2025-2026/Архитектура компьютера/arch-pc $ git commit -am 'feat(main): make course structure'
[master a092a84] feat(main): make course structure
211 files changed, 8073 insertions(+), 207 deletions(-)
delete mode 100644 CHANGELOG.md
create mode 100644 labs/README.md
create mode 100644 labs/README.ru.md
create mode 100644 labs/lab01/presentation/.gitignore
create mode 100644 labs/lab01/presentation/.marksmen.toml
create mode 100644 labs/lab01/presentation/.projectile
create mode 100644 labs/lab01/presentation/Makefile
create mode 100644 labs/lab01/presentation/_quarto.yml

create mode 100644 presentation/report/bib/cite.bib
create mode 100644 presentation/report/image/solvay.jpg
asivanova1@dk3n55 ~/work/study/2025-2026/Архитектура компьютера/arch-pc $ git push
Перечисление объектов: 66, готово.
Подсчет объектов: 100% (66/66), готово.
При сжатии изменений используется до 6 потоков
Сжатие объектов: 100% (52/52), готово.
Запись объектов: 100% (64/64), 700.25 КиБ | 5.26 МиБ/с, готово.
Total 64 (delta 22), reused 1 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (22/22), completed with 1 local object.
To github.com:asivanova1/study_2025-2026_arh-pc.git
9bbc6f2..a092a84 master -> master

```

Рисунок 3.11. – Выполнение команд

Проверим правильность создания иерархии рабочего пространства в локальном репозитории и на странице github (Рисунок 3.12):



	CHANGELOG.md	initial commit	30 minutes ago
	COURSE	feat(main): make course structure	2 minutes ago

Рисунок 3.12. – Проверка выполнения команд

4. Задания для самостоятельной работы

1. Создадим отчет по выполнению лабораторной работы в соответствующем каталоге рабочего пространства (labs/lab02/report):

```

asivanova1@dk3n55 ~ $ mkdir -p labs/lab02/report

```

Рисунок 4.1. – Создание каталога рабочего пространства

2. Скопируем отчеты по выполнению предыдущих лабораторных работ в соответствующие каталоги созданного рабочего пространства и загрузим файлы на github (Рисунок 4.2 – 4.3):

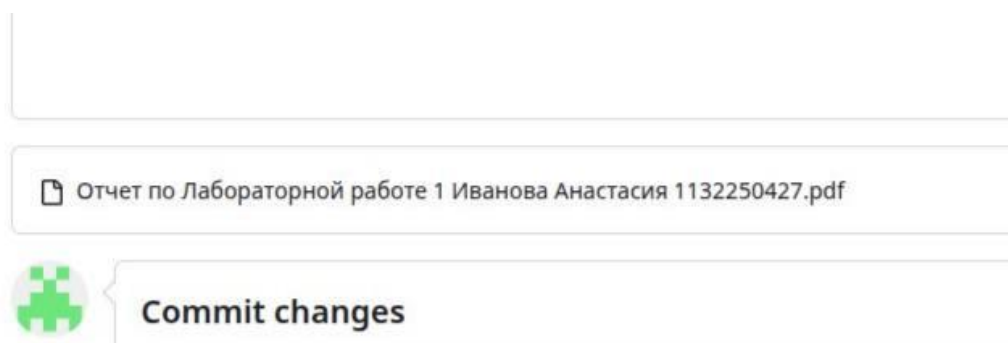


Рисунок 4.2. – Прикрепление на сайт отчета по 1 лабораторной работе

Рисунок 4.3. - Прикрепление на сайт отчета по 2 лабораторной работе

Сайт, в котором мы работали, выполняя лабораторную работу:
<https://github.com/>

5. Заключение

Мы изучили идеологию и применение средств контроля версий, а также приобрели практические навыки по работе с системой контроля версий git.