

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: Лебедева Алёна

Группа: НБИбд-02-21

Ст. билет №: 1032212267

Москва

2022 г.

Цель работы:

Изучить идеологию и применение средств контроля версий и освоить умения по работе с git.

Ход работы:

- 1) Создаю учётную запись на <https://github.com> и заполняем основные данные (рис. 1)

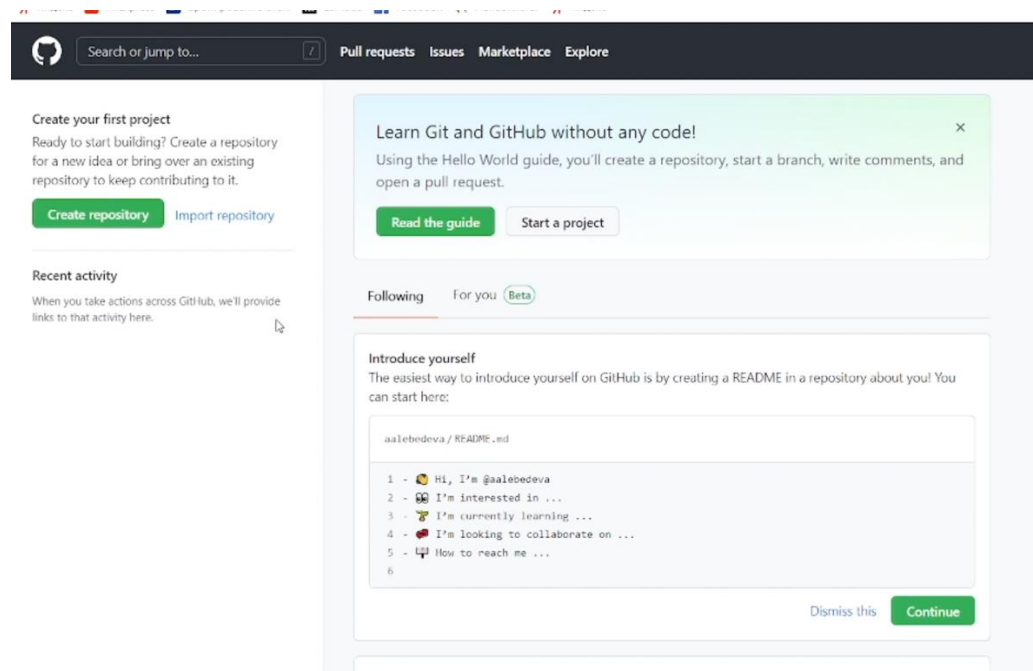


Рис. 1

- 2) Устанавливаем программное обеспечение git-flow в Fedora Linux. Приходится устанавливать его вручную, так как это программное обеспечение удалено из репозитория (рис. 2)

```
aalebedeva@fedora/tmp

ВЭ) С большой властью приходит большая ответственность.

[sudo] пароль для aalebedeva:
sudo: ./gitflow-installer.sh: command not found
[aalebedeva@fedora tmp]$ wget --no-check-certificate -q https://raw.githubusercontent.com/petervanderdoes/gitflow/develop/contrib/gitflow-installer.sh
[aalebedeva@fedora tmp]$ chmod +x gitflow-installer.sh
chmod: невозможно получить доступ к 'gitflow-installer.sh': Нет такого файла или каталога
[aalebedeva@fedora tmp]$ sudo ./gitflow-installer.sh install stable
sudo: ./gitflow-installer.sh: command not found
[aalebedeva@fedora tmp]$ wget --no-check-certificate -q https://raw.githubusercontent.com/petervanderdoes/gitflow/develop/contrib/gitflow-installer.sh
[aalebedeva@fedora tmp]$ chmod +x gitflow-installer.sh
[aalebedeva@fedora tmp]$ sudo ./gitflow-installer.sh install stable
[sudo] пароль для aalebedeva:
## git-flow no-make installer ##
Installing git-flow to /usr/local/bin
Cloning repo from GitHub to gitflow
Клонирование в «gitflow»...
remote: Enumerating objects: 4270, done.
remote: Total 4270 (delta 0), reused 0 (delta 0), pack-reused 4270
Получение объектов: 100% (4270/4270), 1.74 МБ | 1.48 МБ/с, готово.
Определение изменений: 100% (2533/2533), готово.
Уже обновлено.
Ветка «master» отслеживает внешнюю ветку «master» из «origin».
Переключено на новую ветку «master»
install: создание каталога '/usr/local/share/doc'
install: создание каталога '/usr/local/share/doc/gitflow'
install: создание каталога '/usr/local/share/doc/gitflow/hooks'
'gitflow/git-flow' -> '/usr/local/bin/git-flow'
'gitflow/git-flow-init' -> '/usr/local/bin/git-flow-init'
'gitflow/git-flow-feature' -> '/usr/local/bin/git-flow-feature'
'gitflow/git-flow-bugfix' -> '/usr/local/bin/git-flow-bugfix'
'gitflow/git-flow-hotfix' -> '/usr/local/bin/git-flow-hotfix'
'gitflow/git-flow-release' -> '/usr/local/bin/git-flow-release'
'gitflow/git-flow-support' -> '/usr/local/bin/git-flow-support'
'gitflow/git-flow-version' -> '/usr/local/bin/git-flow-version'
'gitflow/gitflow-common' -> '/usr/local/bin/gitflow-common'
'gitflow/gitflow-shFlags' -> '/usr/local/bin/gitflow-shFlags'
'gitflow/git-flow-config' -> '/usr/local/bin/git-flow-config'
```

Рис. 2

3) Устанавливаю gh в Fedora Linux (рис. 3)

```
[aalebedeva@fedora tmp]$ sudo dnf install gh
Последняя проверка окончания срока действия метаданных: 2:04:14 назад, Пт 22 апр 2022 21:52:56.
Зависимости разрешены.
=====
Пакет      Архитектура      Версия      Репозиторий
=====
Установка:
gh         x86_64           2.7.0-1.fc35 updates
=====
Результат транзакции
=====
Установка 1 Пакет

Объем загрузки: 6.6 М
Объем изменений: 32 М
Продолжить? [д/н]: д
Загрузка пакетов:
gh-2.7.0-1.fc35.x86_64.rpm                                     2.7
Общий размер
Проверка транзакции
Проверка транзакции успешно завершена.
Идет проверка транзакции
Тест транзакции проведен успешно.
Выполнение транзакции
Подготовка      :
Установка       : gh-2.7.0-1.fc35.x86_64
Запуск скрипта  : gh-2.7.0-1.fc35.x86_64
Проверка        : gh-2.7.0-1.fc35.x86_64
Установлен:
gh-2.7.0-1.fc35.x86_64
Выполнено!
[aalebedeva@fedora tmp]$
```

Рис. 3

4) Далее осуществляем базовую настройку git. (Зададим имя и email владельца репозитория, Настроим utf-8 вывод сообщений git, Настроим верификацию и подписание коммитов git, Зададим имя начальной ветки (будем называть ее master), параметр autocrlf, параметр safecrlf) (рис. 4-5)

```
[aalebedeva@fedora tmp]$ git config --global user.name "Alyona Lebedeva"
[aalebedeva@fedora tmp]$ git config --global user.email "alena_lebedeva_04777@mail.ru"
```

Рис. 4

```
[aalebedeva@fedora tmp]$ git config --global user.name "Alyona Lebedeva"
[aalebedeva@fedora tmp]$ git config --global user.email "alena_lebedeva_04777@mail.ru"
[aalebedeva@fedora tmp]$ git config --global core.quotepath false
[aalebedeva@fedora tmp]$ git config --global init.defaultBranch master
[aalebedeva@fedora tmp]$ git config --global core.autocrlf input
[aalebedeva@fedora tmp]$ git config --global core.safecrlf warn
```

Рис. 5

- 5) Создаём ключи ssh по алгоритму rsa с ключом размером 4096 бит и по алгоритму ed25519 (рис. 6-7)

```
[aalebedeva@fedora tmp]$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/aalebedeva/.ssh/id_rsa):
Created directory '/home/aalebedeva/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Passphrases do not match. Try again.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Passphrases do not match. Try again.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/aalebedeva/.ssh/id_rsa
Your public key has been saved in /home/aalebedeva/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:x7zoehrH0M1J7iWx6E2lGrqetoZ705xf1hLk44Xbw aalebedeva@fedora
The key's randomart image is:
+---[RSA 4096]-----+
|
| ..
| ..
| .#
| o+ * + o o .
| o. = $ O B E
| . + * + 0 .
| .. . * * . 0
| .o# . #.o
| o+oo+ o+o
+---[SHA256]-----+
[aalebedeva@fedora tmp]$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/aalebedeva/.ssh/id_ed25519): ssh
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Passphrases do not match. Try again.
```

Рис. 6

```
SHA256:x7zoehrH0M1J7iWx6E2lGrqetoZ705xf1hLk44Xbw aalebedeva@fedora
The key's randomart image is:
+---[RSA 4096]-----+
|
| ..
| ..
| .#
| o+ * + o o .
| o. = $ O B E
| . + * + 0 .
| .. . * * . 0
| .o# . #.o
| o+oo+ o+o
+---[SHA256]-----+
[aalebedeva@fedora tmp]$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/aalebedeva/.ssh/id_ed25519): ssh
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Passphrases do not match. Try again.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in ssh
Your public key has been saved in ssh.pub
The key fingerprint is:
SHA256:g7vsYJbRk8RLMzAXjixUv62dL/gyU9JY9lsyJ73KNV aalebedeva@fedora
The key's randomart image is:
+---[ED25519 256]---+
|
| .+o
| o+8..
| ..oo8 o
| ..+ = . #
| . 0 $ B +
| # # # #
| # 0 . *..
| o o + *ooEo
| .+ +oo..
+---[SHA256]-----+
I
```

Рис. 7

- 6) Создаём ключи gpg
 - а. Генерируем ключ
 - б. Выбираем нужные нам опции
 - с. Далее gpg запросит личную информацию, которая сохранится в ключе. Надо её предоставить.

(рис. 8)

```
[aalebedeva@fedora tmp]$ gpg --full-generate-key
gpg (GnuPG) 2.3.2; 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: создан каталог '/home/aalebedeva/.gnupg'
gpg: создан шит с ключами '/home/aalebedeva/.gnupg/pubring.kbx'
Выберите тип ключа:
  (1) RSA and RSA
  (2) DSA and Elgamal
  (3) DSA (sign only)
  (4) RSA (sign only)
  (9) ECC (sign and encrypt) «default»
 (10) ECC (только для подписи)
 (14) Existing key from card
Ваш выбор? 1
длина ключей RSA может быть от 1024 до 4096.
Какой размер ключа Вам необходим? (3072) 4096
Запрошенный размер ключа - 4096 бит
Выберите срок действия ключа.
  0 = не ограничен
  <n> = срок действия ключа - n дней
  <n>w = срок действия ключа - n недель
  <n>m = срок действия ключа - n месяцев
  <n>y = срок действия ключа - n лет
Срок действия ключа? (0) 0
Срок действия ключа не ограничен
Все верно? (y/N) y

GnuPG должен составить идентификатор пользователя для идентификации ключа.
```

Рис. 8

- 7) Добавление PGP ключа в GitHub
 - а. Выводим список ключей и копируем отпечаток приватного ключа (рис. 9)
 - б. Копируем наш сгенерированный PGP ключ в буфер обмена
 - с. Переходим в настройки GitHub (<https://github.com/settings/keys>), нажимаем кнопку *New GPG key* и вставляем полученный ключ в поле ввода(рис. 10)

```
[aalebedeva@fedora tmp]$ gpg --list-secret-keys --keyid-format LONG
gpg: проверка таблицы доверия
gpg: marginals needed: 3 completes needed: 1 trust model: pgp
gpg: глубина: 0 достоверных: 1 подписанных: 0 доверие: 0-, 0q, 0n, 0m, 0f, 1u
/home/aalebedeva/.gnupg/pubring.kbx
-----
sec   rsa4096/4C148165D1EE5677 2022-04-22 [SC]
      D421DEB8FC1D7DEC61159FCE4C148165D1EE5677
uid   [ абсолютно ] Alyona <alena_lebedeva_04777@mail.ru>
ssb   rsa4096/3CB43232D0108A89 2022-04-22 [E]
```

Рис. 9

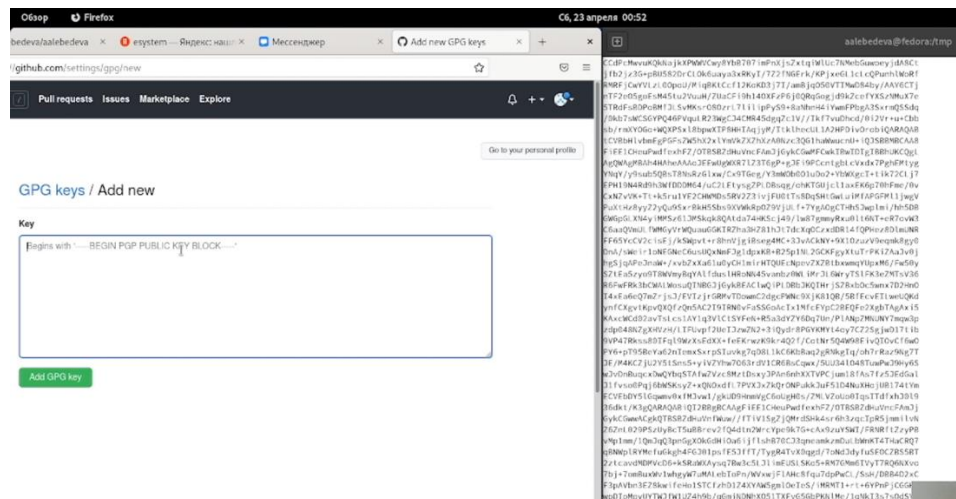


Рис. 10

- 8) Настраиваем автоматические подписи коммитов git
- 9) Настраиваем gh
 - а. Авторизовываемся (рис. 11)

```
[aalebedeva@fedora tmp]$ gh auth login
? What account do you want to log into? GitHub.com
? What is your preferred protocol for Git operations? HTTPS
? Authenticate Git with your GitHub credentials? Yes
? How would you like to authenticate GitHub CLI? Login with a web browser

! First copy your one-time code: 5BBF-047F
Press Enter to open github.com in your browser...
✓ Authentication complete.
- gh config set -h github.com git_protocol https
✓ Configured git protocol
✓ Logged in as aalebedeva
[aalebedeva@fedora tmp]$ gh auth login
? What account do you want to log into? GitHub.com
? You're already logged into github.com. Do you want to re-authenticate? No
```

Рис. 11

- 10) Создаём репозиторий курса на основе шаблона

Создаём шаблон рабочего пространства (рис. 12)


```
Flags:
-c, --clone Clone the new repository to the current directory
-d, --description string Description of the repository
--disable-issues Disable issues in the new repository
--disable-wiki Disable wiki in the new repository
-g, --gitignore string Specify a gitignore template for the repository
-h, --homepage URL Repository home page URL
--internal Make the new repository internal
-l, --license string Specify an Open Source License for the repository
--private Make the new repository private
--public Make the new repository public
--push Push local commits to the new repository
-r, --remote string Specify remote name for the new repository
-s, --source string Specify path to local repository to use as source
-t, --team name The name of the organization team to be granted access
-p, --template repository Make the new repository based on a template repository

[aalebedeva@fedora Операционные системы]$ gh repo create study_2021-2022_os-intro --template=yamadharma/course-directory-student-template --public
/ Created repository aalebedeva/study_2021-2022_os-intro on GitHub
[aalebedeva@fedora Операционные системы]$ git clone --recursive git@github.com:owner>/study_2021-2022_os-intro.git os-intro
bash: owner: Нет такого файла или каталога
[aalebedeva@fedora Операционные системы]$ git clone --recursive git@github.com:aalebedeva/study_2021-2022_os-intro.git os-intro
bash: aalebedeva: Нет такого файла или каталога
[aalebedeva@fedora Операционные системы]$ git clone --recursive git@github.com:aalebedeva/study_2021-2022_os-intro.git os-intro
Клонирование в «os-intro»...
The authenticity of host 'github.com (140.82.121.4)' can't be established.
ED25519 key fingerprint is SHA256:D1Y3wvV6TuJ3hpbZisF/zLDA0zPM5VHdkr4UvC0qu.
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.
git@github.com: Permission denied (publickey).
fatal: Не удалось прочитать из внешнего репозитория.

Удостоверьтесь, что у вас есть необходимые права доступа
и репозиторий существует.
[aalebedeva@fedora Операционные системы]$ git clone https://github.com/aalebedeva/study_2021-2022_os-intro.git os-intro
Клонирование в «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
Получение объектов: 100% (20/20), 12.49 КБ | 12.49 МБ/с, готово.
Определение изменений: 100% (2/2), готово.
[aalebedeva@fedora Операционные системы]$ ls
```

Рис. 12

- 11) Настраиваем каталог
 - а. Переходим в каталог курса и удаляем лишние файлы
 - б. Создаём необходимые каталоги
 - с. Отправляем файлы на сервер

(рис. 13)

```
aalebedeva@fedora:~/work/study/2021-2022/Операционные системы/os-intro
[aalebedeva@fedora Операционные системы]$ cd os-intro
[aalebedeva@fedora os-intro]$ cd template
bash: cd: template: Нет такого файла или каталога
[aalebedeva@fedora os-intro]$ ls
config  labs  LICENSE  Makefile  README.en.md  README.git-flow.md  README.md  template
[aalebedeva@fedora os-intro]$ cd template
[aalebedeva@fedora template]$ cd report
[aalebedeva@fedora report]$ mkdir report
[aalebedeva@fedora report]$ ls
report
[aalebedeva@fedora report]$ cd os-intro
bash: cd: os-intro: Нет такого файла или каталога
[aalebedeva@fedora report]$ cd ~/os-intro
bash: cd: /home/aalebedeva/os-intro: Нет такого файла или каталога
[aalebedeva@fedora report]$ cd -
bash: cd -: command not found...
[aalebedeva@fedora report]$ cd -
/home/aalebedeva/work/study/2021-2022/Операционные системы/os-intro/template
[aalebedeva@fedora template]$ ls
presentation  report
[aalebedeva@fedora template]$ cd -
/home/aalebedeva/work/study/2021-2022/Операционные системы/os-intro/template/report
[aalebedeva@fedora report]$ cd -
/home/aalebedeva/work/study/2021-2022/Операционные системы/os-intro/template
[aalebedeva@fedora template]$ cd ~/work/study/2021-2022/Операционные системы/os-intro
[aalebedeva@fedora os-intro]$ make COURSE=so-intro
Makefile:8: config/course/so-intro: Нет такого файла или каталога
make: *** Нет правила для сборки цели «config/course/so-intro». Останов.
[aalebedeva@fedora os-intro]$ make COURSE=os-intro
[aalebedeva@fedora os-intro]$ ls
config  labs  LICENSE  Makefile  project-personal  README.en.md  README.git-flow.md  README.md  structure  template
[aalebedeva@fedora os-intro]$ git add .
[aalebedeva@fedora os-intro]$ git commit -am 'feat(main): make course structure'
[master 3d43a17] feat(main): make course structure
2 files changed, 14 deletions(-)
delete mode 100644 package.json
create mode 100644 structure
[aalebedeva@fedora os-intro]$ git push
Перечисление объектов: 4, готово.
Подсчет объектов: 100% (4/4), готово.
Сжатие объектов: 100% (2/2), готово.
Запись объектов: 100% (3/3), 962 байта | 962.00 КБ/с, готово.
Всего 3 (изменения 1), повторно использовано 0 (изменений 0), повторно использо
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/aalebedeva/study_2021-2022_os-intro.git
30efc0..3d43a17 master -> master
[aalebedeva@fedora os-intro]$
```

Рис. 13

Вывод:

Изучила идеологию и применение средств контроля версий и освоила умения по работе с git.

Контрольные вопросы:

- 1) Система контроля версий представляет собой набор программ командной строки. Системы контроля версий (Version Control System, VCS) применяются при работе нескольких человек над одним проектом.
- 2) 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 неполную версию изменённых файлов, а производить так называемую дельта-компрессию—сохранять только изменения между последовательными версиями, что позволяет уменьшить объём хранимых данных.
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.
- 3) Централизованные системы — это системы, которые используют архитектуру клиент / сервер, где один или несколько клиентских узлов напрямую подключены к центральному серверу. Например - Wikipedia.
В децентрализованных системах каждый узел принимает свое собственное решение. Конечное поведение системы является совокупностью решений отдельных узлов. Например — Bitcoin.

4) Создадим локальный репозиторий.

Сначала сделаем предварительную конфигурацию, указав имя и email владельца репозитория:

```
git config --global user.name "Имя Фамилия"
```

```
git config --global user.email "work@mail"
```

и настроив utf-8 в выводе сообщений git:

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

Для инициализации локального репозитория необходимо ввести в командной строке:

```
cd
```

```
mkdir tutorial
```

```
cd tutorial
```

```
git init
```

5) Для последующей идентификации пользователя на сервере репозитория необходимо сгенерировать пару ключей (приватный и открытый):

```
ssh-keygen -C "Имя Фамилия <work@mail>"
```

Ключи сохраняются в каталоге ~/.ssh/.

Скопировав из локальной консоли ключ в буфер обмена

```
cat ~/.ssh/id_rsa.pub | xclip -sel clip
```

вставляем ключ в появившееся на сайте поле.

6) Git хранит информацию о всех изменениях в вашем коде, начиная с самой первой строчки, и обеспечивает удобства командной работы над кодом.

7) **git add**

команда `git add` добавляет содержимое рабочего каталога в индекс (staging area) для последующего коммита. По умолчанию `git commit` использует лишь этот индекс, так что вы можете использовать `git add` для сборки слежка вашего следующего коммита.

git status

Команда `git status` показывает состояния файлов в рабочем каталоге и индексе: какие файлы изменены, но не добавлены в индекс; какие ожидают коммита в индексе. Вдобавок к этому выводятся подсказки о том, как изменить состояние файлов.

git diff

Команда `git diff` используется для вычисления разницы между любыми двумя Git деревьями. Это может быть разница между вашей рабочей копией и индексом (собственно `git diff`), разница между индексом и последним

коммитом (`git diff --staged`), или между любыми двумя коммитами (`git diff master branchB`).

git difftool

Команда `git difftool` просто запускает внешнюю утилиту сравнения для показа различий в двух деревьях, на случай если вы хотите использовать что-либо отличное от встроенного просмотрщика `git diff`.

git commit

Команда `git commit` берёт все данные, добавленные в индекс с помощью `git add`, и сохраняет их слепок во внутренней базе данных, а затем сдвигает указатель текущей ветки на этот слепок.

git reset

Команда `git reset`, как можно догадаться из названия, используется в основном для отмены изменений. Она изменяет указатель `HEAD` и, опционально, состояние индекса. Также эта команда может изменить файлы в рабочем каталоге при использовании параметра `--hard`, что может привести к потере наработок при неправильном использовании, так что убедитесь в серьёзности своих намерений прежде чем использовать его.

git rm

Команда `git rm` используется в Git для удаления файлов из индекса и рабочей копии. Она похожа на `git add` с тем лишь исключением, что она удаляет, а не добавляет файлы для следующего коммита.

git mv

Команда `git mv` — это всего лишь удобный способ переместить файл, а затем выполнить `git add` для нового файла и `git rm` для старого.

git clean

Команда `git clean` используется для удаления мусора из рабочего каталога. Это могут быть результаты сборки проекта или файлы конфликтов слияний.

- 8) Использование `git` при работе с локальными репозиториями

```
git add hello.txt
```

```
git commit -am 'Новый файл'
```

Добавление текстового документа в репозитории

- 9) Ветвь — это направление разработки, независимое от других. Ветвь представляет собой копию части хранилища, в которую можно вносить изменения, не влияющие на другие ветви.

- a. Не нужно постоянно создавать архивы с рабочим кодом

- b. Легко "переключаться" между архивами

- c. Легко перетаскивать изменения между архивами

- 10) Во время работы над проектом так или иначе могут создаваться файлы, которые не требуется добавлять в последствии в репозиторий.

Можно прописать шаблоны игнорируемых при добавлении в репозиторий типов файлов в файл .gitignore с помощью сервисов.

а. Для этого сначала нужно получить списки меняющихся шаблонов: `curl -L -s https://www.gitignore.io/api/list`

б. Затем скачать шаблон, например, для С и С++

```
curl -L -s https://www.gitignore.io/api/c >> .gitignore
```

```
curl -L -s https://www.gitignore.io/api/c++ >> .gitignore
```