

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: Балханова Алтана Юрьевна

Группа: НПМбд-03-21

МОСКВА

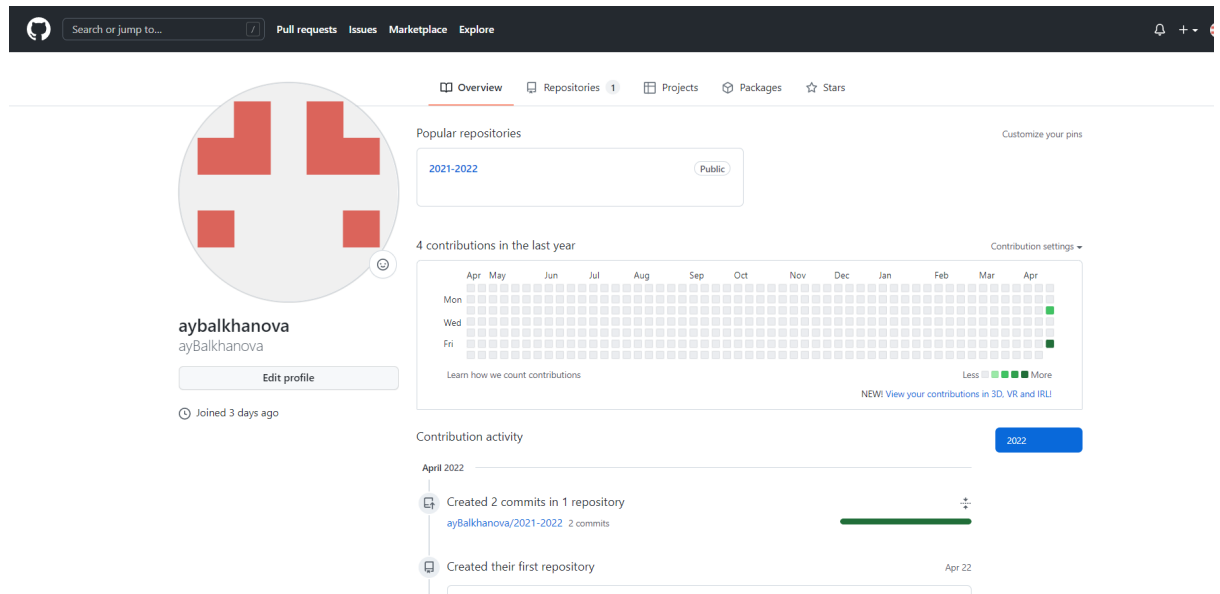
2022 г.

Цель работы:

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

Ход работы:

1. Создала учётную запись на github:



2. Задала имя и email своего репозитория:

```
aybalkhanova@dk4n58 ~ $ git config --global user.name "aybalkhanova"
aybalkhanova@dk4n58 ~ $ git config --global user.email "AltanaBalkhanova@yandex.ru"
```

3. Настроила utf-8 в выводе сообщений git:

```
aybalkhanova@dk4n58 ~ $ git config --global core.quotePath false
```

4. Настроила верификацию и подписание коммитов git, задала имя начальной ветки (будем называть её master):

```
aybalkhanova@dk4n58 ~ $ git config --global init.defaultBranch master
```

5. Задала параметры autocrlf и safecrlf:

```
aybalkhanova@dk4n58 ~ $ git config --global core.autocrlf input
```

```
aybalkhanova@dk4n58 ~ $ git config --global core.safecrlf warn
```

6. Создала ключ ssh – по алгоритму rsa с ключём размером 4096 бит:

```
aybalkhanova@dk4n58 ~ $ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
```

```

Enter file in which to save the key (/afs/.dk.sci.pfu.edu.ru/home/a/y/aybalkhanova/.ssh/id_rsa):
Created directory '/afs/.dk.sci.pfu.edu.ru/home/a/y/aybalkhanova/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /afs/.dk.sci.pfu.edu.ru/home/a/y/aybalkhanova/.ssh/id_rsa
Your public key has been saved in /afs/.dk.sci.pfu.edu.ru/home/a/y/aybalkhanova/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:J4vdrFKBqoteLgkpNrVkr6GNNcLSZK0FSB+tTmyen/Y aybalkhanova@dk4n58
The key's randomart image is:
+---[RSA 4096]-----+
|o.=B.+          |
|.o=o0o.         |
|.o+o. .         |
| 0 .. .         |
|.0 +. S..        |
|+o =. o.*        |
|+ oo. o.o o      |
|++ +. .          |
|o.oo. .E.        |
+----[SHA256]-----+

```

7. Создала ключ ssh – по алгоритму ed25519:

```
aybalkhanova@dk4n58 ~ $ ssh-keygen -t ed25519
```

```

Generating public/private ed25519 key pair.
Enter file in which to save the key (/afs/.dk.sci.pfu.edu.ru/home/a/y/aybalkhanova/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /afs/.dk.sci.pfu.edu.ru/home/a/y/aybalkhanova/.ssh/id_ed25519
Your public key has been saved in /afs/.dk.sci.pfu.edu.ru/home/a/y/aybalkhanova/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:QECi6CyPCi4PAEC/e3aY0jDYxsJ1RdNnmcThGi0W1yA aybalkhanova@dk4n58
The key's randomart image is:
+--[ED25519 256]--+
|o...o.oE..oo+   |
|o... . .+.+*    |
|+ . o+ +oo      |
|+ .....o +     |
|==... S.        |
|=*+ . o         |
|+o+. = .        |
|=. .+ .         |
|oo...          |
+----[SHA256]-----+

```

8. Создала ключи pgr:

```
aybalkhanova@dk4n58 ~ $ gpg --full-generate-key
gpg (GnuPG) 2.2.27; 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.
```

-Выбрала тип ключа RSA и RSA:

```
Выберите тип ключа:
  (1) RSA и RSA (по умолчанию)
  (2) DSA и Elgamal
  (3) DSA (только для подписи)
  (4) RSA (только для подписи)
 (14) Имеющийся на карте ключ
Ваш выбор? 1
```

-Выбрала размер ключа 4096:

```
длина ключей RSA может быть от 1024 до 4096.
Какой размер ключа Вам необходим? (3072) 4096
Запрошенный размер ключа - 4096 бит
```

-Выбрала срок действия - неограничен

```
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
```

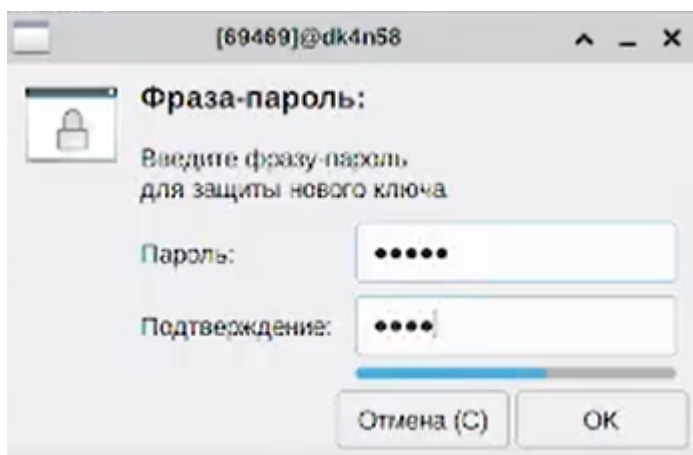
-Для составления идентификатора пользователя я ввела своё имя и адрес электронной почты:

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

Ваше полное имя: aybalkhanova
Адрес электронной почты: AltanaBalkhanova@yandex.ru
Примечание:
Вы выбрали следующий идентификатор пользователя:
  "aybalkhanova <AltanaBalkhanova@yandex.ru>"

Сменить (N)Имя, (C)Примечание, (E)Адрес; (O)Принять/(Q)Выход?
```

-Ввела фразу-пароль для защиты ключа:



9. Вывела список ключей и пыталась скопировать отпечаток приватного ключа:

```
aybalkhanova@dk4n58 ~ $ 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
/afs/.dk.sci.pfu.edu.ru/home/a/y/aybalkhanova/.gnupg/pubring.kbx
-----

sec  rsa4096/4497780EE1162BA5 2022-04-21 [SC]
      4D99D3F843AB4639C48C93614497780EE1162BA5
uid          [ абсолютно ] aybalkhanova <AltanaBalkhanova@yandex.ru>
ssb  rsa4096/566904C763E50997 2022-04-21 [E]

aybalkhanova@dk4n58 ~ $ gpg --armor --export <4497780EE1162BA5> | xclip -sel clip
bash: синтаксическая ошибка рядом с неожиданным маркером «|»
```

-Но у меня это не получилось. Во время выполнения отчёта я поняла, что проблема была в том, что я не убрала угловые скобки <>, когда писала свой PGP Fingerprint. Но во время выполнения лабораторной работы, я зашла на github, где нашла, как вывести отпечаток ключа:

- 12 Paste the text below, substituting in the GPG key ID you'd like to use. In this example, the GPG key ID is 3AA5C343715678D2 :

```
$ gpg --armor --export 3AA5C343715678D2
# Prints the GPG key ID, in ASCII armor format
```

-Затем, я ввела эту команду, меняя данный пример отпечатка ключа на свой и скопировала выведенный ключ:


```
Терминал - aybalkhanova@dk4n58:~
Файл Правка Вид Терминал Вкладки Справка
orUBoEAsUqQ40AgKaW1o65FS4dBK07MHZqEGDsnahz0KivjwZrmFmXGUFdx+rpWv
hc8/SmhhQdToqQG/XcrjZ+ZgTUp1YhLbsItEkVm0Ait0kDi5zW263OxXi01P8MDR
Qbn7MzmcIZjR5YhMQ4glD+keCOaKFlsFL9A5dZuqdpZQQZzQULXjqt2x1TVnQipf
EGxI1UBa2pYiRNDeg9eysRMn0+/P5TJVL1zeypt3QobpRHmi5DCUMmv/dIfrvL40
3N07v45rz5vuUSSuHBTUilGqJOp4M03z1aU0FukzcAR3Ij1pIVsDnFPazaxYrdP1
+XRxt/xy7jOqmuICLjx03nM09BJs5Wwh4gwyiKoKi/2KoAsFwwrArYyxWrpGySKv
GTB9Z6RQ2TUA46P4YOFu5iQrVoChB+dHvBom/2VX8uke5EISHPy192tsHEtDUptC
gZNE915S1QARAQABiQI2BBgBCAAgFiEETZnT+EOrRjnEjJNhrJd4DuEWK6UFAMJh
VPsCGwWACgkQRJd4DuEWK6X5mA/5Aa5FhW2Y5EI4KtuhAR1AT2gt3fNLnx1jDpMF
CsZBZxkFQRYnxkiiyCoT5M0ik0dSWJfeN90Ym8dYw72i091+3CH+aYdCeB9gBnxy
+z2UuUZ1201+dR5g8hFfdHVhoxnvcZQo9F7n28ORY0lk/3Hyqq7dIYEYIV0Abfcs
KHGeNpicRzi5XPrU07yhYcMYuQIkZiUEZ/dDnazhayWhu34bTUUZJicHkoKcm07e
ccA4xjJmR8YiFfY7TpebRL+6SSJaEY72Bpw3aF7dh0YyuWALLRCOUOxsQdBx5s/f
EOZ7T7sHwrV9Z3KouWbZ1I8lw/nGayXv9NI7jIGBnWzYXAwNv/DXwyI/zf7oxeBs
KX6u2Iz3DSdOS/bot/768bykM/MKbTsSi37PSE7dgKxTN6uXwK3tHrgrS4zZHhti
M4UI0bRjVzpb90vKqAVGbbQlFMnUtnZSlgC9uE4auHOyHzEgRVuIvR6bxTk1LLIAB
wxos+K7Tks0dsEUK5FYPH74G1kcN2HeXTtptzoTjBMgrhL6sKEBz3OpRS9lo29/
SoWiD8NxxnfdE3vbnXb2FjHNNG9YWzN1uCOqXeZFX9nsJHCetF5u99JdMOZfisIF
pehgthDcIz2NkORFsBf9Nr/QggCzQ6Lx81R7FzfkXCpkxPiucikMExyzVU9R+n6
TmeGCfw=
=pWdw
-----END PGP PUBLIC KEY BLOCK-----
aybalkhanova@dk4n58 ~ $ ^C
aybalkhanova@dk4n58 ~ $
```

-После этого я перешла в настройки GitHub, нажала на кнопку New GPG key и вставила скопированный ключ в поле ввода




-Так он выглядит:

GPG keys

New GPG key

This is a list of GPG keys associated with your account. Remove any keys that you do not recognize.



Email address: `AltanaBalkhanova@yandex.ru`
Key ID: `4497780EE1162BA5`
Subkeys: `566904C763E50997`
Added on 21 Apr 2022

Delete

[Learn how to generate a GPG key and add it to your account.](#)

10. Затем я попыталась настроить автоматические подписи коммитов git, но не убрала угловые скобки `<>`, из-за чего операция не удалась. Это я поняла только во время выполнения отчёта:

```
aybalkhanova@dk4n58 ~ $ git config --global user.signingkey <4497780EE1162BA5>
bash: синтаксическая ошибка рядом с неожиданным маркером «newline»
```

Вывод:

Я изучила идеологию и применение средств контроля версий, освоила умения по работе с git.

Контрольный вопросы:

1. Системы контроля версий (VCS) представляет собой программное обеспечение, которое позволяет отслеживать изменения в документах, при необходимости производить их откат, определять, кто и когда внес исправления.
2. Хранилище — это содержимое скрытой папки `.git`. В этой папке хранятся все версии рабочей области и служебная информация. Этим версиям система автоматически даёт название, состоящее из букв и цифр. Коммиты — основные конструктивные элементы временной шкалы проекта Git. 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. Примером является CVS, Subversion. Децентрализованная система означает, что у каждого разработчика есть личный репозиторий проекта с полным набором всех версий. Примером является Git.
4. Создадим тестовый текстовый файл `hello.txt` и добавим его в локальный репозиторий:
`echo 'hello world' > hello.txt`

```
git add hello.txt
```

```
git commit -am 'Новый файл'
```

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

```
ssh-keygen -C "Имя Фамилия "
```

Ключи сохраняются в каталоге ~/.ssh/

6. Хранение информации об изменениях в коде. Благодаря тому, что Git является распределённой системой контроля версий, резервную копию локального хранилища можно сделать простым копированием или архивацией.

7. Наиболее часто используемые команды git:

- создание основного дерева репозитория: `git init`

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

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

- просмотр списка изменённых файлов в текущей директории: `git status`

- просмотр текущих изменений: `git diff`

- сохранение текущих изменений:

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

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

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

- сохранение добавленных изменений:

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

- сохранить добавленные изменения с внесением комментария через встроенный редактор: `git commit`

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

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

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

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

- удаление ветки:

- удаление локальной уже слитой с основным деревом ветки: `git branch -d имя_ветки`
- принудительное удаление локальной ветки: `git branch -D имя_ветки`
- удаление ветки с центрального репозитория: `git push origin :имя_ветки`

8. Создаём тестовый текстовый файл `hello.txt` и добавляем его в локальный репозиторий:
`echo 'hello world' > hello.txt`
`git add hello.txt`
`git commit -am 'Новый файл'`
9. Ветка в Git — 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.
Это дает нам возможность легко откатывать код, если вдруг мы передумаем его сливать в основную ветку, либо делать несколько различных изменений в разных ветках.
10. Во время работы над проектом так или иначе могут создаваться файлы, которые не требуется добавлять в последствии в репозиторий. Например, временные файлы, создаваемые редакторами, или объектные файлы, создаваемые компиляторами. Можно прописать шаблоны игнорируемых при добавлении в репозиторий типов файлов в файл `.gitignore` с помощью сервисов.