

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: Кашкин Иван

Группа: НБИбд-01-21

Ст. билет №: 1032212958

Москва

2022 г.

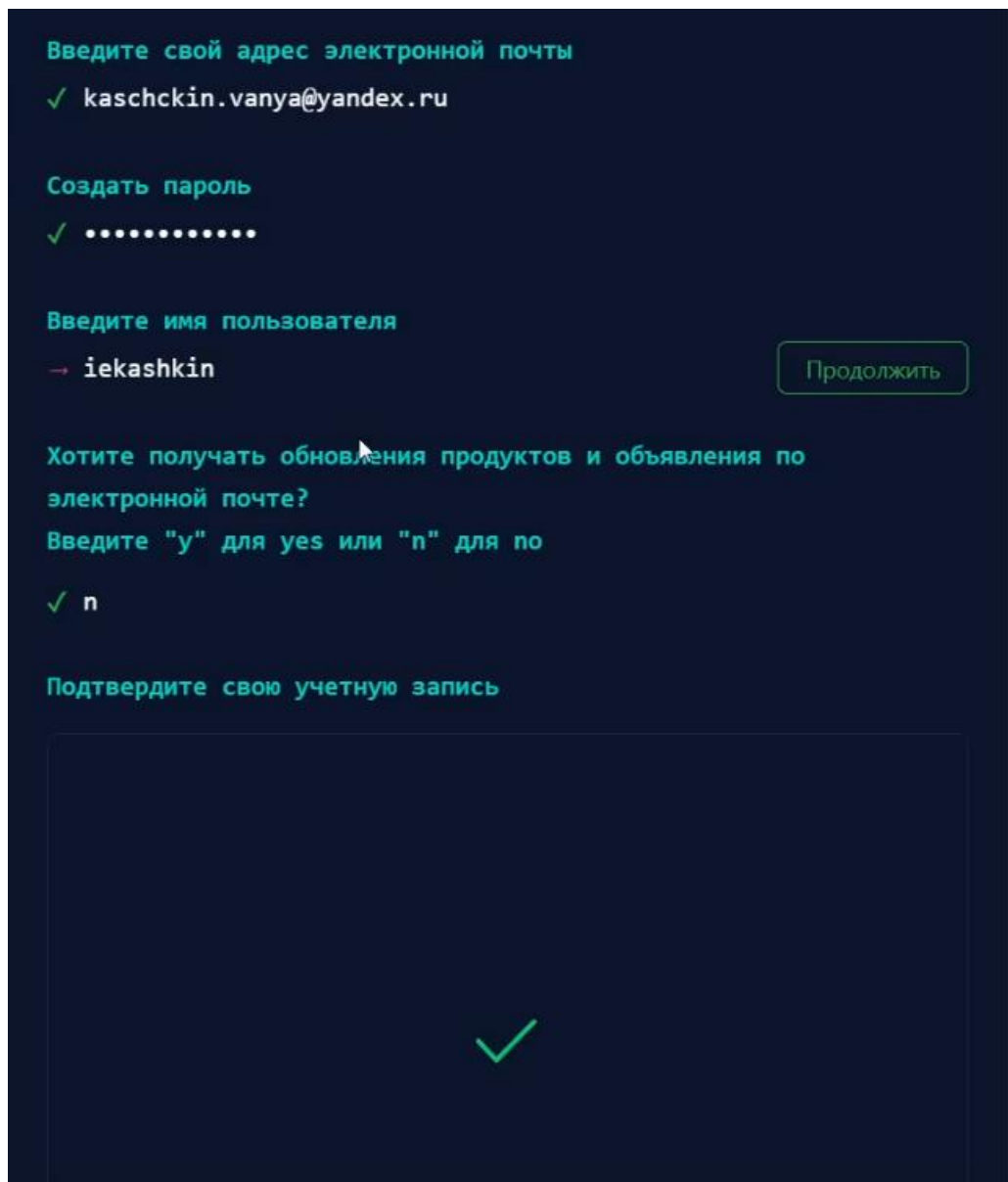
Цель:

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

Ход работы:

1) Настройка GitHub:

- Создайте учётную запись на <https://github.com>.
- Заполните основные данные на <https://github.com> (рис. 1.1; рис 1.2)



Введите свой адрес электронной почты

✓ kaschckin.vanya@yandex.ru

Создать пароль

✓

Введите имя пользователя

→ iekashkin

Продолжить

Хотите получать обновления продуктов и объявления по электронной почте?

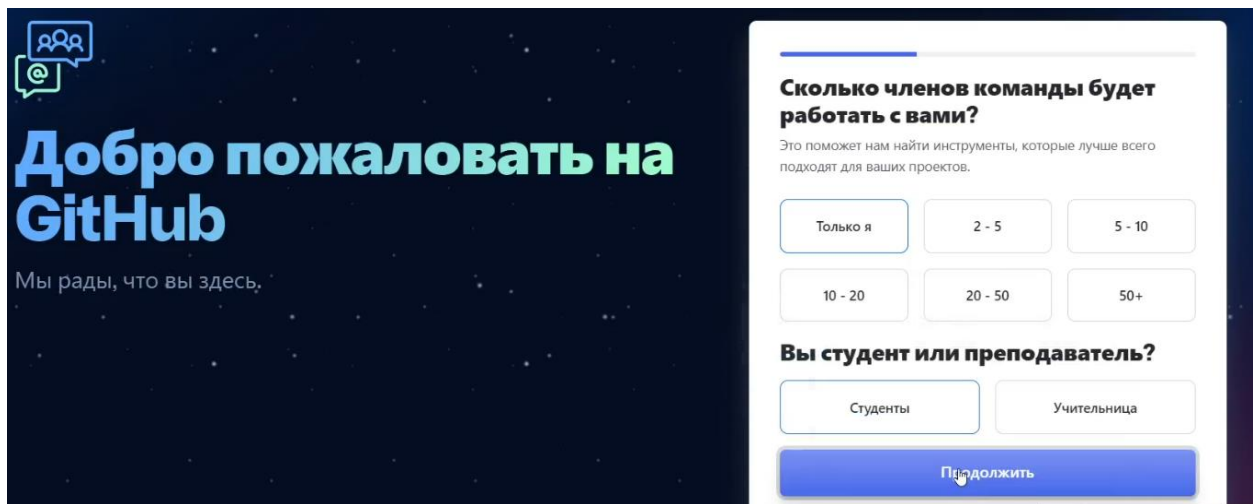
Введите "y" для yes или "n" для no

✓ n

Подтвердите свою учетную запись

✓

(Рис.1.1)



(Рис. 1.2)

2) Установка программного обеспечения:

Мы начали выполнять этот пункт с установки git-flow н нашу виртуальную машину. Это программное обеспечение удалено из репозитория. Необходимо устанавливать его вручную: (рис. 2.1; рис. 2.2)

1. «cd /tmp»
2. «wget --no-check-certificate -q
<https://raw.githubusercontent.com/petervanderdoes/gitflow/develop/contrib/gitflow-installer.sh>»
3. «chmod +x gitflow-installer.sh»
4. «sudo ./gitflow-installer.sh install stable»

```
ivanekashkin@fedora:tmp — sudo ./gitflow-installer.sh install stable

[ivanekashkin@fedora tmp]$ mkdir tmp
[ivanekashkin@fedora tmp]$ cd /tmp
[ivanekashkin@fedora tmp]$ wget --no-check-certificate -q https://raw.githubusercontent.com/petervanderdoes/gitflow/develop/contrib/gitflow-installer.sh
[ivanekashkin@fedora tmp]$ chmod +x gitflow-installer.sh
chmod: невозможно получить доступ к 'gitflow-installer.sh': Нет такого файла или каталога
[ivanekashkin@fedora tmp]$ cd
[ivanekashkin@fedora ~]$ wget --no-check-certificate -q https://raw.githubusercontent.com/petervanderdoes/gitflow/develop/contrib/gitflow-installer.sh
[ivanekashkin@fedora ~]$ chmod +x gitflow-installer.sh
chmod: невозможно получить доступ к 'gitflow-installer.sh': Нет такого файла или каталога
[ivanekashkin@fedora ~]$ cd /tmp
[ivanekashkin@fedora tmp]$ wget --no-check-certificate -q https://raw.githubusercontent.com/petervanderdoes/gitflow/develop/contrib/gitflow-installer.sh
[ivanekashkin@fedora tmp]$ chmod +x gitflow-installer.sh
[ivanekashkin@fedora tmp]$ sudo ./gitflow-installer.sh install stable

Мы полагаем, что ваш системный администратор изложил вам основы безопасности. Как правило, всё сводится к трём следующим правилам:

#1) Уважайте частную жизнь других.
#2) Думайте, прежде что-то вводить.
#3) С большой властью приходит большая ответственность.

[sudo] пароль для ivanekashkin:
sudo: ./gitflow-installer.sh: command not found
[ivanekashkin@fedora tmp]$ sudo ./gitflow-installer.sh install stable
## 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 | 3.05 МБ/с, готово.
Определение изменений: 90% (2280/2533)
```

(Рис.2.1)

```
'gitflow/hooks/filter-flow-release-finish-tag-message' -> '/usr/local/share/doc/gitflow/hooks/filter-flow-release-finish-tag-message'
'gitflow/hooks/filter-flow-release-start-version' -> '/usr/local/share/doc/gitflow/hooks/filter-flow-release-start-version'
'gitflow/hooks/post-flow-bugfix-delete' -> '/usr/local/share/doc/gitflow/hooks/post-flow-bugfix-delete'
'gitflow/hooks/post-flow-bugfix-finish' -> '/usr/local/share/doc/gitflow/hooks/post-flow-bugfix-finish'
'gitflow/hooks/post-flow-bugfix-publish' -> '/usr/local/share/doc/gitflow/hooks/post-flow-bugfix-publish'
'gitflow/hooks/post-flow-bugfix-pull' -> '/usr/local/share/doc/gitflow/hooks/post-flow-bugfix-pull'
'gitflow/hooks/post-flow-bugfix-start' -> '/usr/local/share/doc/gitflow/hooks/post-flow-bugfix-start'
'gitflow/hooks/post-flow-bugfix-track' -> '/usr/local/share/doc/gitflow/hooks/post-flow-bugfix-track'
'gitflow/hooks/post-flow-feature-delete' -> '/usr/local/share/doc/gitflow/hooks/post-flow-feature-delete'
'gitflow/hooks/post-flow-feature-finish' -> '/usr/local/share/doc/gitflow/hooks/post-flow-feature-finish'
'gitflow/hooks/post-flow-feature-publish' -> '/usr/local/share/doc/gitflow/hooks/post-flow-feature-publish'
'gitflow/hooks/post-flow-feature-pull' -> '/usr/local/share/doc/gitflow/hooks/post-flow-feature-pull'
'gitflow/hooks/post-flow-feature-start' -> '/usr/local/share/doc/gitflow/hooks/post-flow-feature-start'
'gitflow/hooks/post-flow-feature-track' -> '/usr/local/share/doc/gitflow/hooks/post-flow-feature-track'
'gitflow/hooks/post-flow-hotfix-delete' -> '/usr/local/share/doc/gitflow/hooks/post-flow-hotfix-delete'
'gitflow/hooks/post-flow-hotfix-finish' -> '/usr/local/share/doc/gitflow/hooks/post-flow-hotfix-finish'
'gitflow/hooks/post-flow-hotfix-publish' -> '/usr/local/share/doc/gitflow/hooks/post-flow-hotfix-publish'
'gitflow/hooks/post-flow-hotfix-start' -> '/usr/local/share/doc/gitflow/hooks/post-flow-hotfix-start'
'gitflow/hooks/post-flow-release-branch' -> '/usr/local/share/doc/gitflow/hooks/post-flow-release-branch'
'gitflow/hooks/post-flow-release-delete' -> '/usr/local/share/doc/gitflow/hooks/post-flow-release-delete'
'gitflow/hooks/post-flow-release-finish' -> '/usr/local/share/doc/gitflow/hooks/post-flow-release-finish'
'gitflow/hooks/post-flow-release-publish' -> '/usr/local/share/doc/gitflow/hooks/post-flow-release-publish'
'gitflow/hooks/post-flow-release-start' -> '/usr/local/share/doc/gitflow/hooks/post-flow-release-start'
'gitflow/hooks/post-flow-release-track' -> '/usr/local/share/doc/gitflow/hooks/post-flow-release-track'
'gitflow/hooks/pre-flow-feature-delete' -> '/usr/local/share/doc/gitflow/hooks/pre-flow-feature-delete'
'gitflow/hooks/pre-flow-feature-finish' -> '/usr/local/share/doc/gitflow/hooks/pre-flow-feature-finish'
'gitflow/hooks/pre-flow-feature-publish' -> '/usr/local/share/doc/gitflow/hooks/pre-flow-feature-publish'
'gitflow/hooks/pre-flow-feature-pull' -> '/usr/local/share/doc/gitflow/hooks/pre-flow-feature-pull'
'gitflow/hooks/pre-flow-feature-start' -> '/usr/local/share/doc/gitflow/hooks/pre-flow-feature-start'
'gitflow/hooks/pre-flow-feature-track' -> '/usr/local/share/doc/gitflow/hooks/pre-flow-feature-track'
'gitflow/hooks/pre-flow-hotfix-delete' -> '/usr/local/share/doc/gitflow/hooks/pre-flow-hotfix-delete'
'gitflow/hooks/pre-flow-hotfix-finish' -> '/usr/local/share/doc/gitflow/hooks/pre-flow-hotfix-finish'
'gitflow/hooks/pre-flow-hotfix-publish' -> '/usr/local/share/doc/gitflow/hooks/pre-flow-hotfix-publish'
'gitflow/hooks/pre-flow-hotfix-start' -> '/usr/local/share/doc/gitflow/hooks/pre-flow-hotfix-start'
'gitflow/hooks/pre-flow-release-branch' -> '/usr/local/share/doc/gitflow/hooks/pre-flow-release-branch'
'gitflow/hooks/pre-flow-release-delete' -> '/usr/local/share/doc/gitflow/hooks/pre-flow-release-delete'
'gitflow/hooks/pre-flow-release-finish' -> '/usr/local/share/doc/gitflow/hooks/pre-flow-release-finish'
'gitflow/hooks/pre-flow-release-publish' -> '/usr/local/share/doc/gitflow/hooks/pre-flow-release-publish'
'gitflow/hooks/pre-flow-release-start' -> '/usr/local/share/doc/gitflow/hooks/pre-flow-release-start'
'gitflow/hooks/pre-flow-release-track' -> '/usr/local/share/doc/gitflow/hooks/pre-flow-release-track'
[ivanekashkin@fedora tmp]$
```

(Рис.2.2)

После этого нам нужно установить gh в Fedora Linux с помощью команды:
«sudo dnf install gh»(рис. 3)

```
[ivanekashkin@fedora tmp]$ sudo dnf install gh
Последняя проверка окончания срока действия метаданных: 1:07:11 назад, Чт 21 апр 2022 09:34:08.
Зависимости разрешены.
=====
Пакет                Архитектура          Версия                Репозиторий           Размер
=====
Установка:
gh                    x86_64                2.7.0-1.fc35          updates                6.8 М
=====
Результат транзакции
=====
Установка 1 Пакет
=====
Объем загрузки: 6.8 М
Объем изменений: 32 М
Продолжить? [д/Н]: д
Загрузка пакетов:
gh-2.7.0-1.fc35.x86_64.rpm                                4.7 MB/s | 6.8 MB  00:01
=====
Общий размер
gh-2.7.0-1.fc35.x86_64.rpm                                3.1 MB/s | 6.8 MB  00:02
=====
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                1/1
Запуск скрипта  : gh-2.7.0-1.fc35.x86_64                1/1
Проверка        : gh-2.7.0-1.fc35.x86_64                1/1
Установлен:
gh-2.7.0-1.fc35.x86_64
Выполнено!
[ivanekashkin@fedora tmp]$
```

(Рис. 3)

Далее по лабораторной работе мы устанавливали базовые настройки git с помощью программ: (рис. 4)

1. «git config --global user.name "Name Surname"»
2. «git config --global user.email "work@mail"»

Эти две команды выше задают имя и email владельца репозитория.

Настроим utf-8 в выводе сообщений git с помощью: (рис.4)

1. «git config --global core.quotePath false»

Настроим верификацию и подписание коммитов git. Зададим имя начальной ветки master: (рис. 4)

1. «git config --global init.defaultBranch master»

Настраиваем параметры autocrlf и safecrlf: (рис. 4)

1. «git config --global core.autocrlf input»
2. «git config --global core.safecrlf warn»

```
[ivanekashkin@fedora tmp]$ git config --global user.name "iekashkin777"
[ivanekashkin@fedora tmp]$ git config --global user.email "kachckin.vanya@yandex.ru"
[ivanekashkin@fedora tmp]$ git config --global core.quotePath false
[ivanekashkin@fedora tmp]$ git config --global init.defaultBranch master
[ivanekashkin@fedora tmp]$ git config --global core.autocrlf input
[ivanekashkin@fedora tmp]$ git config --global core.safecrlf warn
[ivanekashkin@fedora tmp]$
```

(Рис. 4)

От этих действий мы перешли к созданию ключа ssh. По алгоритму rsa с ключём размером 4096 бит, а после по алгоритму ed25519, пишем комнды: (рис.5.1; рис.5.2)

1. «ssh-keygen -t rsa -b 4096»
2. «ssh-keygen -t ed25519»

```
[ivanekashkin@fedora tmp]$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/ivanekashkin/.ssh/id_rsa):
Created directory '/home/ivanekashkin/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ivanekashkin/.ssh/id_rsa
Your public key has been saved in /home/ivanekashkin/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:24Rg8YfVTNt5FoGUQa/wvi4X7ac34dfL0WXoc6J8qR4 ivanekashkin@iekashkin
The key's randomart image is:
+---[RSA 4096]---+
|      . o=Xoo   |
|      o o+.*   |
|      o + . .o  |
|      . . = .o  |
|      S +o . o|
|      =. ....o|
|      . ooE.+++|
|      . .o.o0++|
|      +o.** =. |
+-----[SHA256]-----+
[ivanekashkin@fedora tmp]$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/ivanekashkin/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
```

(Рис.5.1)

```
[ivanekashkin@fedora tmp]$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/ivanekashkin/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ivanekashkin/.ssh/id_ed25519
Your public key has been saved in /home/ivanekashkin/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:brBZWZ/NX6YvcW6zRa1xYg3I9TRiRD3tlZou8aCQxJE ivanekashkin@iekashkin
The key's randomart image is:
+---[ED25519 256]---+
|      ..o   o=o.+|
|      E   ..o.**|
|      . . . o +.+|
|      o o + * oo|
|      . S . B * B|
|      * . . +.Xo|
|      o o   . o+o|
|      .      .o+|
|      ++|
+-----[SHA256]-----+
```

(Рис.5.2)

После создаем ключ gpg: (рис.6)

1. «gpg --full-generate-key»


```
[ivanekashkin@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/ivanekashkin/.gnupg'
gpg: создан щит с ключами '/home/ivanekashkin/.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.

Ваше полное имя: Ivan
Имя не должно быть короче 5 символов
Ваше полное имя: Ivan Kashkin
Адрес электронной почты: kaschckin.vanya@yandex.ru
Примечание:
Вы выбрали следующий идентификатор пользователя:
  "Ivan Kashkin <kaschckin.vanya@yandex.ru>"

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

(Рис.6)

Добавим этот ключ в GitHub (рис.7.1 рис.7.2 рис.7.3 рис.7.4) (<PGP Fingerprint> | xclip -sel clip – Эта часть команды у меня не работал и я писал свой «Отпечаток_ключа»)

```
[ivanekashkin@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/ivanekashkin/.gnupg/pubring.kbx
-----
sec   rsa4096/DF8CA12E5F73745F 2022-04-21 [SC]
      CE8802B7B05A8948F2B1FB46DF8CA12E5F73745F
uid   [ абсолютно ] Ivan Kashkin <kaschckin.vanya@yandex.ru>
ssb   rsa4096/F7C22B1D88BA3EF8 2022-04-21 [E]

[ivanekashkin@fedora tmp]$ gpg --armor █
```

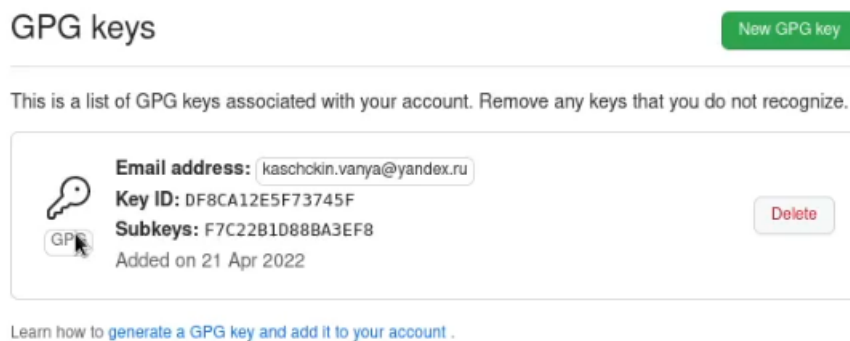
(Рис.7.1)

```
[ivanekashkin@fedora tmp]$ gpg --armor --export DF8CA12E5F73745F
-----BEGIN PGP PUBLIC KEY BLOCK-----
I
mQINBGJhDIcBEACVYiiCuwcaQzum4aehM99YtDcH9nqhKILAZTOHQTYbeiFLBj9I
YCoWCpu/99B0U1GE91Wpl+h16HzN+6e3MH/+M9UDKxji47QCLklygr6bTulqHFeC
CDwmDXyMItl7It4Z6JzMTV51mDkyYj1kCKG/KBgXxb8EX5yKk6cgUwLMY84dX+Ms
nNmpamcyHtgA844VG/Da9xeUmB6MyepTm8mXgZT0ke+YNBjVfsyIbmZj03NAGNd5
9ZLxVYPdwFt9jMGWQ5oyAH3+4G97mnsJvpA/M4nacDw/WryLXsW1F2KPtRy/UoU5
S530njGYGC4nzEs7E4iJ1CgJo4gyl3ZQ3u348bX2TbPbrRRZR10VZhQ52CozIfZ1
```

(Рис.7.2)

```
tChJdmFuIEthc2hraW4gPGthc2NoY2tpb152YW55YU85YW5kZXgucnU+IQJSBBMB
CAA8FiEEzogCt7BaiUjysftG34yhLL9zdF8FAMJhDlCgGwMFCwkIBwIDIGtBBHUK
CQgLAGQWAgMBAh4HAheAAoJEN+MoS5fc3RfEskQAIRGkxbCFdQ060InBs6N8KG1
zYDo4w+bsBphaFMV2Sjsu7P01Rg9h3KkXfwwiB5zRQjc9YttsnmRRE351KxT/Hk2
ZCBsB6wFLmIxp0BAFgpXvekB353CKr5/1oHtPsnthnYNRtFY5FSvd8a/vWe5QrTE
ttcMe0yFiTvfAZaxZLmplU9iwwt6EKxzZbHuiyig5XM/5903k4BkiHVF2WD459FC
Mef4JrJb2FSxzIIldJn2cbXNGjPul/EK6c6uv9Yl95CtThxzmV89gVLQ3I+dEqz9
k29Ky9j9b9FZfy5HYlRdEaJDUPtH4X0f0Gc8oMuD3WTJlc9BMo99dQyb7fM049q
eLtwPAngMChmqtLfc1u8+K+QX/qWD9fk2MYFHgmihwc/YTOuWZRxRgtw35q/H0vd
+yvYP+2BievuICQIAxPYRla8b/6CX/VdRgLXJFFlRoKuzs7LTPlo/rhliXh8N49P
G8J0EGHgTUViVW84viIQK5PagsBahfIFqYl02NSgVskoZJZcVCKpjTefLbjgwSV6
Q4sG2HQ1pE33WzoA2AeYwZ3gygrJg2EQnfDqkj7/gw6CF1oUL3wTrmqmKffFjeoq
4LS7G+S+rG0dcBuR+RoCw3U1hDoQYI4pE02rKdsBSXnHkG1P5MuWfSvffKvX1E60
eWxH03C9PRIwJ5qmKUWouQINBGJhDlCBEAC8/8LecxSLVmLfmoQwbFwcYpQVwM8y
8nwsBfrEY83/ZYfELbljCs1PC7NRT8lMfDsDY6LDuUioXT8iRl1mqhNUBna/z7Z
ApM4KDQWCW/0k7WoaYkQ172WQnNp+XCrGSU1i5IBY2xkpGZBbBH3Q0STLCuD54y3
0BSUcvo4tEUBCA19ihBxTHmQ6NzC6rvYignRTAESeaBUV5skLAX3++v2NxxKstxz
fle46nxiAJkGaxp2IsQU6J3jD80lgqctZCnw1k3zkjcCiF6mbhREsy6kPWJBAXHg
lTdu0x1N0o0X4Uc1t48Vs7Gy9Frk709qx5QPhnUa5pKPU5ifyG7BH4E207mZngw8
VTll15hZiUCyU0A0o90k3aew0lyCeCro2aa1Q32rLh+kgerqDVHZMntc/nP3WS3Z
QebR5nuD29D7+80kqMESEtQivPwW3Kpl9bDlIkmb0ZJ6+j9U0GhybiWoTzxaDTSF
KXuR3+juJWp9dY4rr4RVvP4bhHMoMwa5kWCXBPK/EGNv99hw729nbj0kqbZIWLEZ
i92zvc/128hbk+ljmY6eWavBRyFJjWcTWmhJfmMmdzV5YtNAQDyAbY2YuZ8Qkjxo
NLZ0inYMJCa2dK5lWylr4K+YR0FFXIdtbHYE7AwbV/OvuglXCr47iWGzHTU9yAGT
SekMchPYbrJ2EQARAQABiQI2BBgBCAAgFiEEzogCt7BaiUjysftG34yhLL9zdF8F
AMJhDlCgGwACgkQ34yhLL9zdF+mWA/5ASnFs4bCJtz0XKmPskqf9up8NoipLlJJ
oE3mLmJhQBeszqPGRwV7rfvL/SCJikFIk8MJ7QDUYINfVeEJVCBSW0aCoP7zf5k
65c4N+dZ6F2JcDV+fAa9oU6fzePhEJikIE0HLKf55hvc7p2cGIQTQz7TtyWkDtQ/Q
y4KJv1ldgCJkK1nsRkRiwC+Eh59TFthc2Q7krf1IC72TrHpa5+Gm184inUhsDlKY
RVA7qRoegZg6tp6G0f4pPqbzKW4qZpoJpwQefNfjPH9AyqoeMKTM0uxYUz7QC7Gz
Wf1+4FI9kF3NKgsA7HJ0ZibIEyybbEqcfn9dzbILv4SossnV+hAecX0G2DY2q21H
jqPEJlHg0kndobK8y5aHTAmnSePP2Up22LFP0LLvFCu5/wY28Qrk5EN/6JsnF8Ky
sie9+VTSwoR/o/Q2FVvAh7bj572uKUEd0u0ALLPA9FLTPC6ohIGbgLFLqKj3Hxr
7AmsG+jf+/Dg4oUVYelFm+TU40/lqVtx5EcZn0IFs/NLTwfD7s2Ns/J0DZjlAqG
6ypjS/21mFZWz9+/IODsibr22KU4G8PAuwocvNWkeaq+S1db03J8x4I8erg3NMm
8I8rcLunR5c824IUhUbc/k0A24gorOfxmQ110ZamZPzuxKwVTirxJz7+azMe35x7
04Sb9cNUZqQ=
=sviX
-----END PGP PUBLIC KEY BLOCK-----
[ivanekashkin@fedora tmp]$
```

(Puc. 7.3)



(Puc. 7.4)

Далее идет настройка автоматических подписей коммитов git. Мы используем введённый email, укажем Git применять его при подписи коммитов: (рис.8)

2. «git config --global user.signingkey «Отпечаток_ключа»»
3. «git config --global commit.gpgsign true»
4. «git config --global gpg.program \$(which gpg2)»


```
[ivanekashkin@fedora tmp]$ git config --global user.signingkey DF8CA12E5F73745F
[ivanekashkin@fedora tmp]$ git config --global user.gpgsign true
[ivanekashkin@fedora tmp]$ git config --global gpg.program $(which gpg2)
[ivanekashkin@fedora tmp]$
```

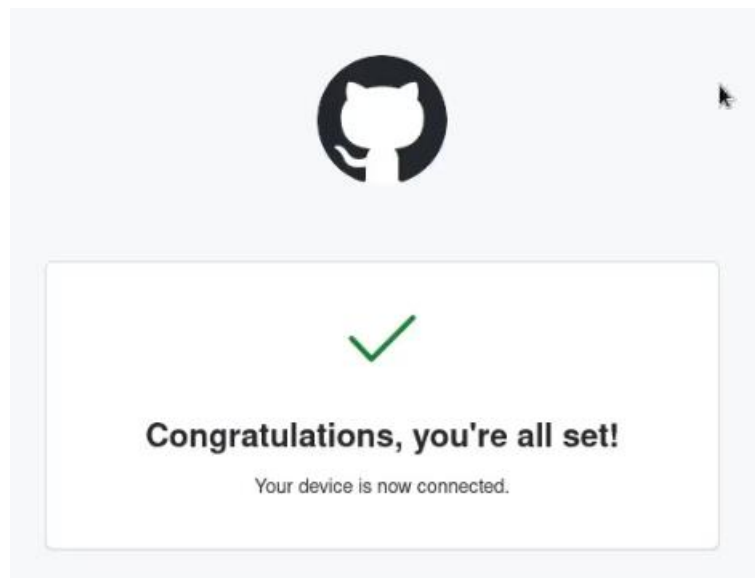
(Рис.8)

Настройка gh и авторизация с помощью консоли (рис.9.1 рис9.2)

```
[ivanekashkin@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: C65A-8E8E
Press Enter to open github.com in your browser...
█
```

(Рис.9.1)



(Рис.9.2)

Мы создаем репозиторий курса на основе шаблона (рис.10.1 рис.10.2)

1. `mkdir -p ~/work/study/2021-2022/"Операционные системы"`
2. `cd ~/work/study/2021-2022/"Операционные системы"`
3. `gh repo create study_2021-2022_os-intro --template=yamadharma/course-directory-student-template --public`
4. `git clone --recursive git@github.com:<iekashkin777>/study_2021-2022_os-intro.git os-intro`

```
[ivanekashkin@fedora tmp]$ mkdir -p ~/work/study/2021-2022/"Операционные системы"
[ivanekashkin@fedora Операционные системы]$ gh repo create study_2021-2022_os-intro --template=yamadharma/course-directory-student-template --
[ivanekashkin@fedora Операционные системы]$ gh repo create study_2021-2022_os-intro --template=yamadharma/course-directory-student-template --
public
unknown flag: --public

Usage: gh repo create [<name>] [flags]

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
```

(Рис.10.1)

```
[ivanekashkin@fedora Операционные системы]$ git clone https://git@github.com/iekashkin777/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 КиБ | 3.12 МБ/с, готово.
Определение изменений: 100% (2/2), готово.
[ivanekashkin@fedora Операционные системы]$
```

(Рис.10.2)

И в конце идет настройка каталога курса (рис11.1 рис 11.2 рис 11.3 рис 11.4)

1. `cd ~/work/study/2021-2022/"Операционные системы"/os-intro`
2. `rm package.json`
3. `make COURSE=os-intro`

Пояснение для этих команд: первая, мы заходим в каталог курса, вторая, удаляем не нужные файлы, третья, создаем необходимые каталоги

```
[ivanekashkin@fedora os-intro]$ rm package.json
```

(Рис.11.1)

```

[ivaneekashkin@fedora os-intro]$ cd template
[ivaneekashkin@fedora template]$ cd presentation
[ivaneekashkin@fedora presentation]$ mkdir presentation
[ivaneekashkin@fedora presentation]$ ls
presentation
[ivaneekashkin@fedora presentation]$ cd
[ivaneekashkin@fedora ~]$ cd ~/work/study/2021-2022/"Операционные системы"/os-intro
[ivaneekashkin@fedora os-intro]$ make COURSE=os-intro
cp: не удалось выполнить stat для 'template/report/report': Нет такого файла или каталога
cp: не удалось выполнить stat для 'template/report/report': Нет такого файла или каталога
cp: не удалось выполнить stat для 'template/report/report': Нет такого файла или каталога
cp: не удалось выполнить stat для 'template/report/report': Нет такого файла или каталога
cp: не удалось выполнить stat для 'template/report/report': Нет такого файла или каталога
cp: не удалось выполнить stat для 'template/report/report': Нет такого файла или каталога
cp: не удалось выполнить stat для 'template/report/report': Нет такого файла или каталога
cp: не удалось выполнить stat для 'template/report/report': Нет такого файла или каталога
cp: не удалось выполнить stat для 'template/report/report': Нет такого файла или каталога
cp: не удалось выполнить stat для 'template/report/report': Нет такого файла или каталога
cp: не удалось выполнить stat для 'template/report/report': Нет такого файла или каталога
cp: не удалось выполнить stat для 'template/report/report': Нет такого файла или каталога
cp: не удалось выполнить stat для 'template/report/report': Нет такого файла или каталога
cp: не удалось выполнить stat для 'template/report/report': Нет такого файла или каталога
cp: не удалось выполнить stat для 'template/report/report': Нет такого файла или каталога
cp: не удалось выполнить stat для 'template/report/report': Нет такого файла или каталога
cp: не удалось выполнить stat для 'template/report/report': Нет такого файла или каталога
cp: не удалось выполнить stat для 'template/report/report': Нет такого файла или каталога
[ivaneekashkin@fedora os-intro]$ git

```

(Рис.11.2)

```

[ivaneekashkin@fedora os-intro]$ cd
[ivaneekashkin@fedora ~]$ cd report
bash: cd: report: Нет такого файла или каталога
[ivaneekashkin@fedora ~]$ cd ~/work/study/2021-2022/"Операционные системы"/os-intro
[ivaneekashkin@fedora os-intro]$ cd report
bash: cd: report: Нет такого файла или каталога
[ivaneekashkin@fedora os-intro]$ mkdir report
[ivaneekashkin@fedora os-intro]$ cd report
[ivaneekashkin@fedora report]$ mkdir report
[ivaneekashkin@fedora report]$ ls
report
[ivaneekashkin@fedora report]$ cd
[ivaneekashkin@fedora ~]$ cd ~/work/study/2021-2022/"Операционные системы"/os-intro
[ivaneekashkin@fedora os-intro]$ make COURSE=os-intro
make: Цель «all» не требует выполнения команд.
[ivaneekashkin@fedora os-intro]$

```

(Рис.11.3)

```

[ivaneekashkin@fedora os-intro]$ ls
config  labs  LICENSE  Makefile  os-intro  project-personal  README.en.md  README.git-flow.md  README.md  report  structure  template
[ivaneekashkin@fedora os-intro]$ ls
config  labs  LICENSE  Makefile  os-intro  project-personal  README.en.md  README.git-flow.md  README.md  report  structure  template
[ivaneekashkin@fedora os-intro]$

```

(Рис.11.4)

Контрольные вопросы:

- 1) Система контроля версий Git представляет собой набор программ командной строки. Доступ к ним можно получить из терминала посредством ввода команды git с различными опциями. 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. Пример - Wikipedia.

В децентрализованных системах каждый узел принимает свое собственное решение. Конечное поведение системы является совокупностью решений отдельных узлов. Пример — Bitcoin.

В классических системах контроля версий используется централизованная модель, предполагающая наличие единого репозитория для хранения файлов. Выполнение большинства функций по управлению версиями осуществляется специальным сервером.

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 quotepath false
```

Для инициализации локального репозитория, расположенного, например, в каталоге ~/tutorial, 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 сохраняться 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) У Git две основных задачи: первая — хранить информацию о всех изменениях в вашем коде, начиная с самой первой строчки, а вторая — обеспечение удобства командной работы над кодом.

7) Основные команды git:

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 изменения: 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 имена_файлов – удалить файл и/или каталог из индекса репозитория (при этом файл и/или каталог остаётся в локальной директории): git rm имена_файлов – 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 стекущим 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) Использование `git` при работе с локальными репозиториями (добавления
текстового документа в локальный репозиторий):

```
git add hello.txt
```

```
git commit -am 'Новый файл'
```

9) Проблемы, которые решают ветки `git`:

- нужно постоянно создавать архивы с рабочим кодом
- сложно "переключаться" между архивами
- сложно перетаскивать изменения между архивами
- легко что-то напутать или потерять

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

Затем скачать шаблон, например, для C и C++

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

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

Вывод:

Я изучил изучить идеологию и применение средств контроля версий. Освоил умения по работе с git.