

Лабораторная работа №1

Дисциплина: Основы информационной безопасности

Тихонова Екатерина Андреевна

Содержание

Цель работы

Часть 1: установка на виртуальную машину системы Rocky Linux и настройка необходимых сервисов. Часть 2: освоение git и применение средств контроля версий. Часть 3: оформление отчётов с помощью языка разметки Markdown.

Задание

Часть 1 1.Установка и настройка Виртуальной машины VirtualBox (ОС Linux)

Часть 2 1.Создать базовую конфигурацию для работы с git. 2.Создать ключ SSH. 3.Создать ключ PGP. 4.Настроить подписи git. 5.Зарегистрироваться на GitHub. 6.Создать локальный каталог для выполнения заданий по предмету.

Часть 3 1.Сделайте отчёт по предыдущей лабораторной работе в формате Markdown. 2.В качестве отчёта предоставить отчёты в 3 форматах: pdf,docx и md

Выполнение лабораторной работы

Часть 1. Установка и настройка Виртуальной машины VirtualBox

1)Переходим на официальный сайт VirtualBox и скачиваем файл. Устанавливаем VirtualBox на компьютер. Я часто пользуюсь Linux, поэтому VirtualBox уже был установлен на компьютере.

VirtualBox

Начальная

Download VirtualBox

Here you will find links to VirtualBox binaries and its source code.

VirtualBox binaries

By downloading, you agree to the terms and conditions of the respective license.

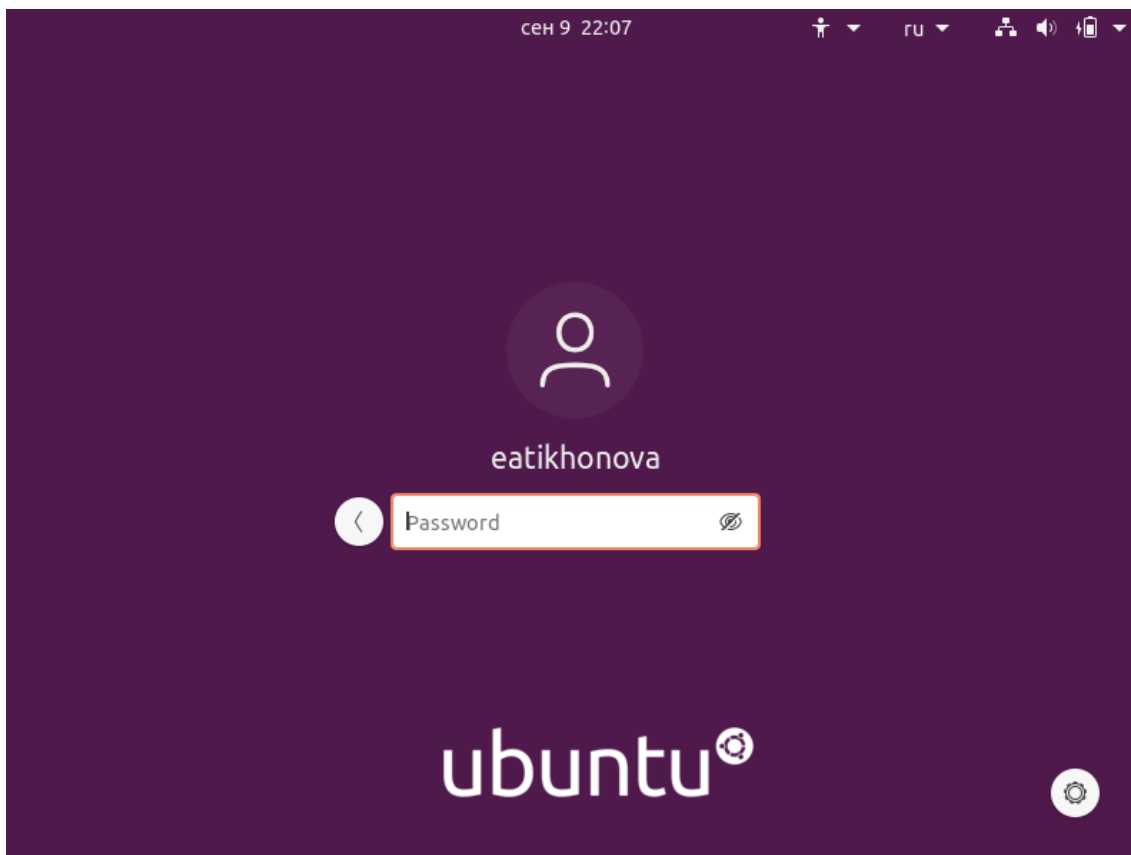
If you're looking for the latest VirtualBox 6.1 packages, see [VirtualBox 6.1 builds](#). Version 6.1 will remain supported until December 2023.

VirtualBox 7.0.10 platform packages

- [Windows hosts](#)
- [macOS / Intel hosts](#)
- [Linux distributions](#)
- [Solaris hosts](#)
- [Solaris 11 IPS hosts](#)

Официальный сайт VirtualBox

2) Создаём виртуальную машину. Для этого нажимаем на “Создать”. Имя нашей виртуальной машины - eatikhonova. Выбираем наш образ диска Rocky Linux. Чтобы он распознал его как образ, предварительно скачиваем UltraISO. Автоматически выбирается Linux версии RedHat.



Пример установки:

Создать виртуальную машину

Имя и операционная системы виртуальной машины

Пожалуйста укажите имя и местоположение новой виртуальной машины. Заданное Вами имя будет использоваться для идентификации данной машины. Кроме того, вы можете выбрать ISO образ для установки операционной системы.

Имя: ✓

Папка:

Образ ISO:

Редакция:

Тип: 64

Версия:

☒ Пропустить автоматическую установку

Вы выбрали пропустить автоматическую установку гостевой ОС, гостевая ОС должна быть установлена вручную.

[Справка](#) [Экспертный режим](#) [Назад](#) [Далее](#) [Отмена](#)

Создание виртуальной машины

3) Виртуальной машине требуется ОЗУ. Выделяем 4096 МБ, что является половиной основного ОЗУ. Также выделяем ей 4 ЦП.

Создать виртуальную машину

Оборудование

Вы можете настроить оборудование виртуальной машины, изменяя размер ОЗУ и количество виртуальных процессоров. Также возможна активация EFI.

Основная память: 4 МБ 8192 МБ

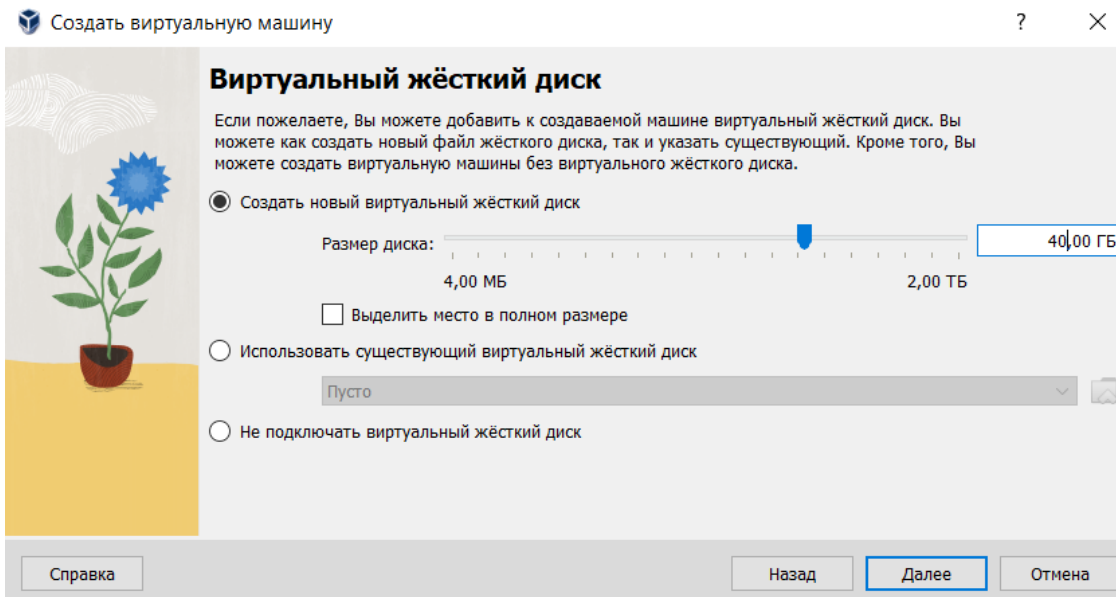
Процессоры: 1 ЦП 12 ЦП

☐ Включить EFI (только специальные ОС)

[Справка](#) [Назад](#) [Далее](#) [Отмена](#)

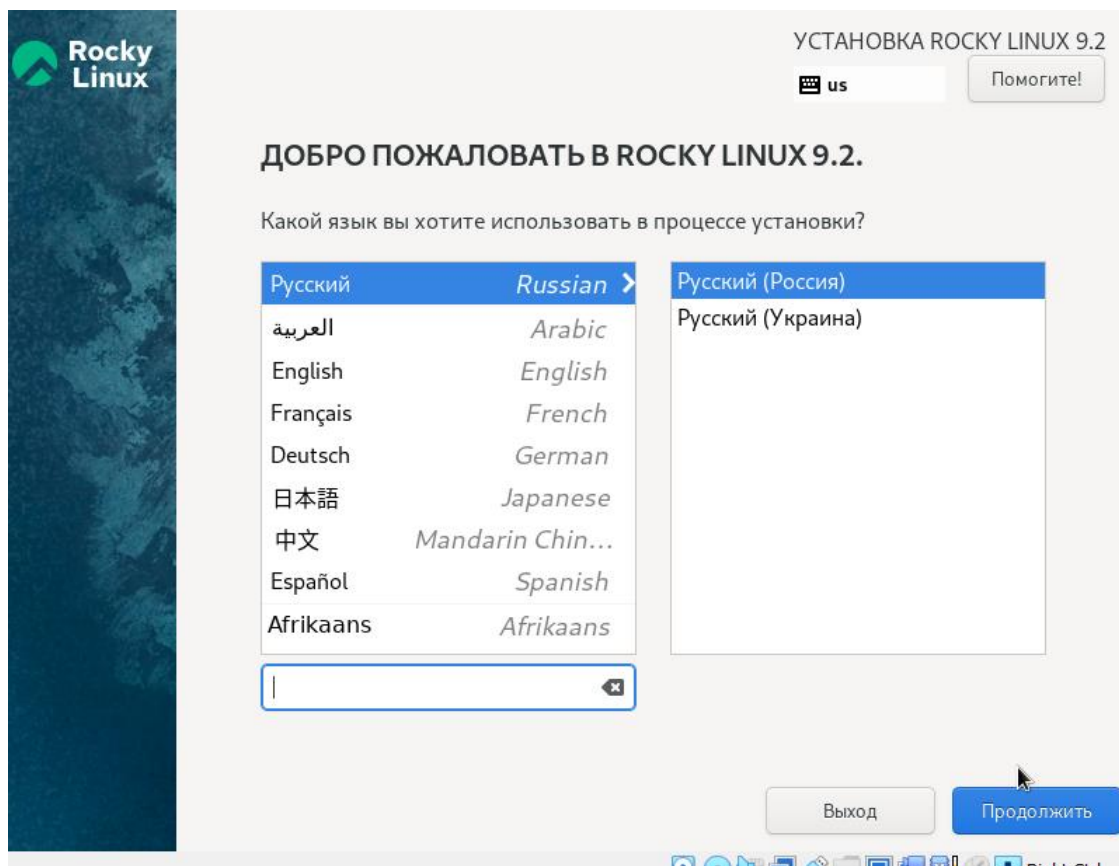
Память и процессоры

4) Выделяем 40 ГБ памяти на виртуальную машину, думаю, этого достаточно.



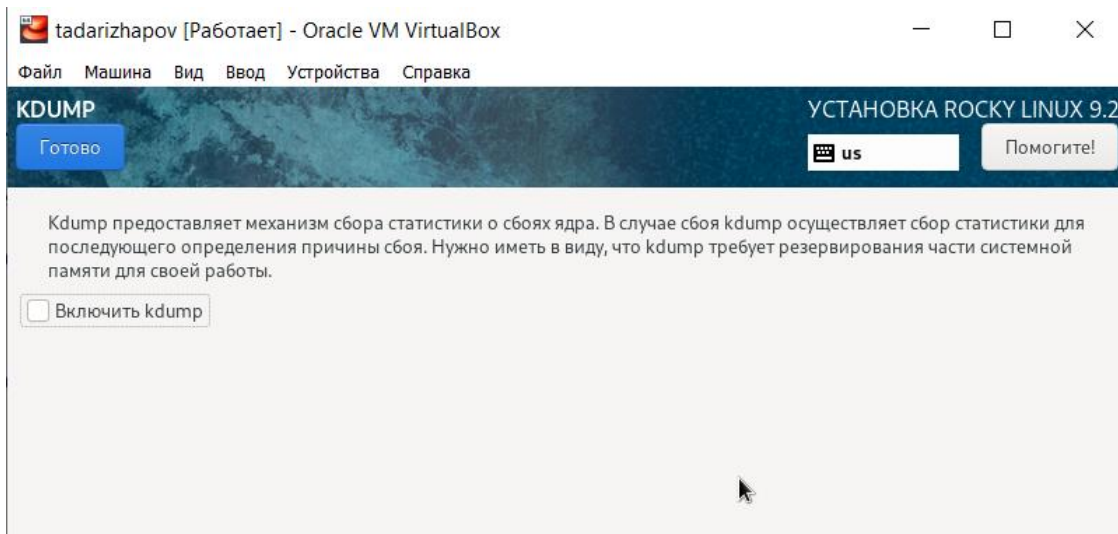
Основная память

5) Включаем виртуальную машину, нас встречает выбор языка.



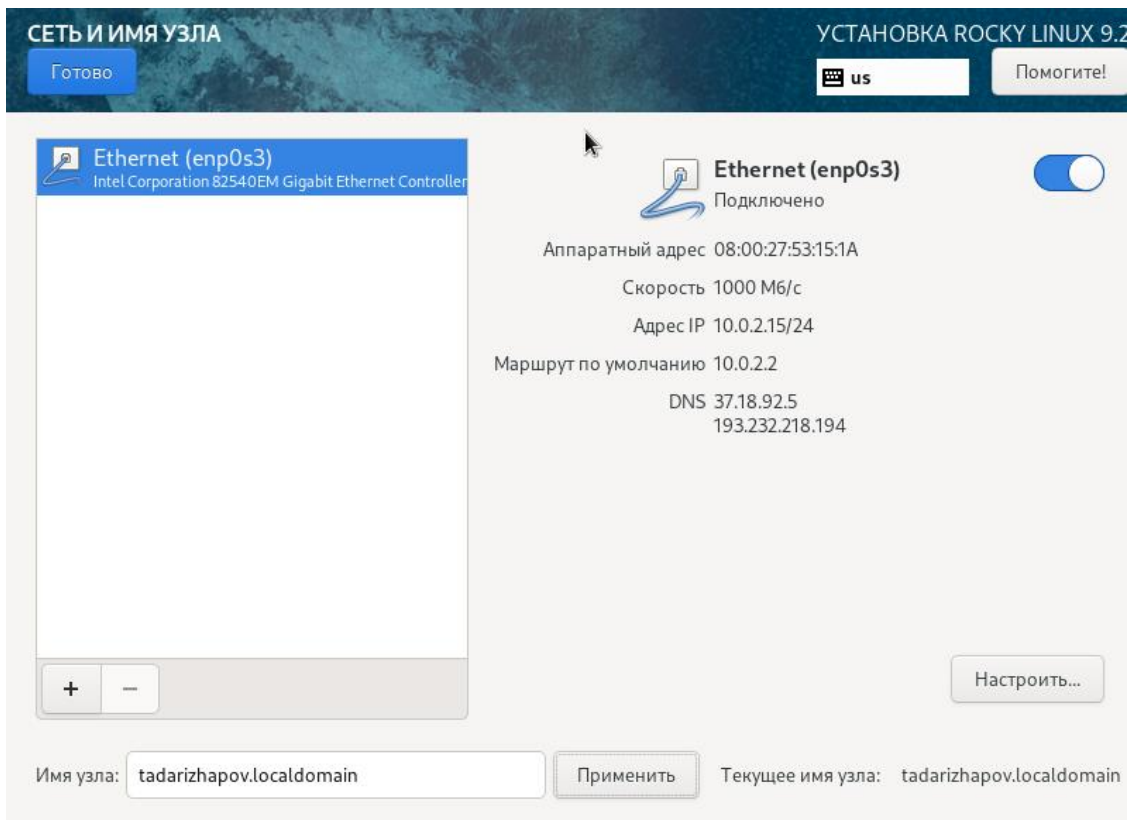
Выбор языка

6) Отключаем KDUMP.



Отключение KDUMP

7) Настраиваем сеть и имя узла



Настройка интернета

8) Базовое окружение выбираем Сервер с GUI. В дополнительном окружении выбираем средства разработки.

ВЫБОР ПРОГРАММ

Готово

УСТАНОВКА ROCKY LINUX 9.2

us

Помогите!

Базовое окружение

- ☒ **Сервер с GUI**
Интегрированный, простой в управлении сервер с графическим интерфейсом.
- ☐ **Server**
Интегрированный, простой в управлении сервер.
- ☐ **Минимальная установка**
Базовая функциональность.
- ☐ **Рабочая станция**
Рабочая станция - это удобная для пользователя настольная система для ноутбуков и ПК.
- ☐ **Пользовательская операционная система**
Базовый строительный блок для индивидуальной системы Rocky Linux.
- ☐ **Хост виртуализации**
Минимальный комплект хоста виртуализации.

Дополнительное программное обеспечение для выбранной среды

- ☐ **Совместимость с устаревшими функциями UNIX**
Программы совместимости для миграции или работы с устаревшими окружениями UNIX.
- ☐ **Консольные средства Интернета**
Консольные средства доступа к Интернету, обычно используемые администраторами.
- ☐ **Управление контейнерами**
Инструменты для управления контейнерами Linux
- ☒ **Средства разработки**
Стандартная среда разработки.
- ☐ **.NET Development**
Tools to develop and/or run .NET applications
- ☐ **Графические средства администрирования**
Графические программы управления системными компонентами.
- ☐ **Безголовое управление**
Инструменты для управления системой без подключенной графической консоли.
- ☐ **Инструменты разработки оборотов оборотов**
Tools used for building RPMs, such as rpmbuild.
- ☐ **Инженерные инструменты**
Средства для математических и научных вычислений и преобразований, а также параллельных вычислений.

Выбор программ

9) Устанавливаем пароль для root пользователя.

ПАРОЛЬ ROOT

Готово

УСТАНОВКА ROCKY LINUX 9

us

Помогите!

Учетная запись администратора (root) предназначена для управления системой. Введите пароль root.

Пароль root:

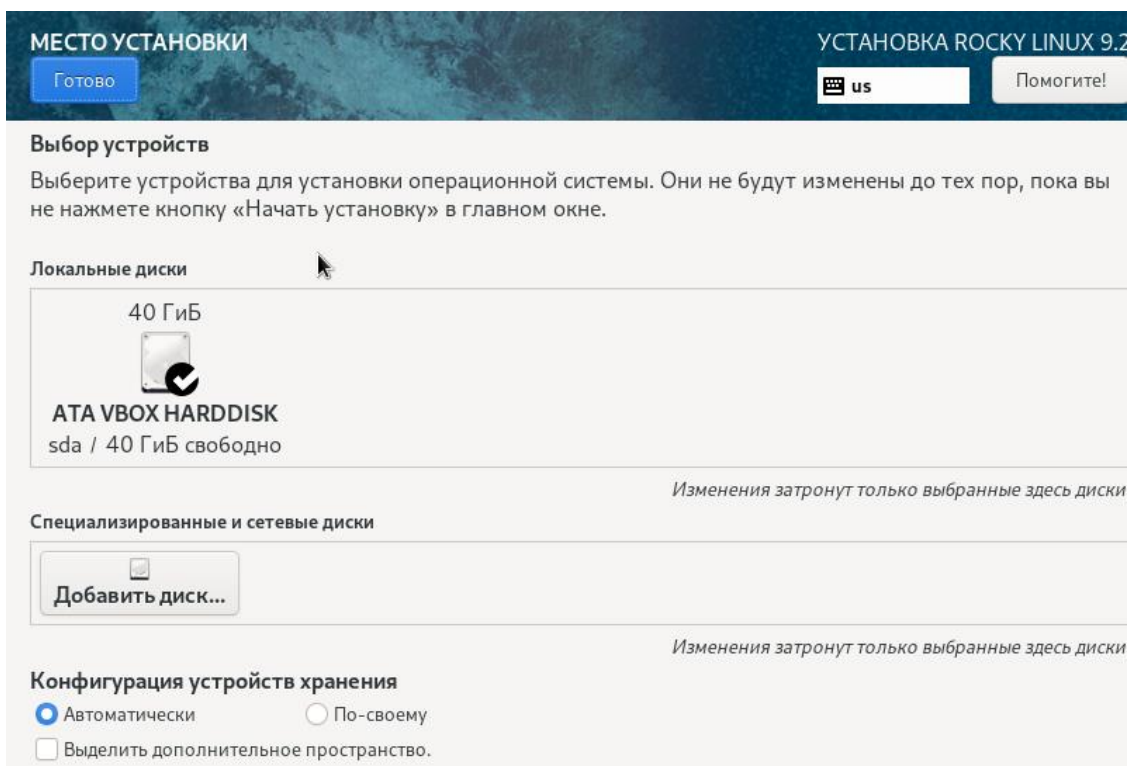
Подтверждение:

☐ Заблокировать учётную запись root

☐ Разрешить вход пользователем root с паролем через SSH

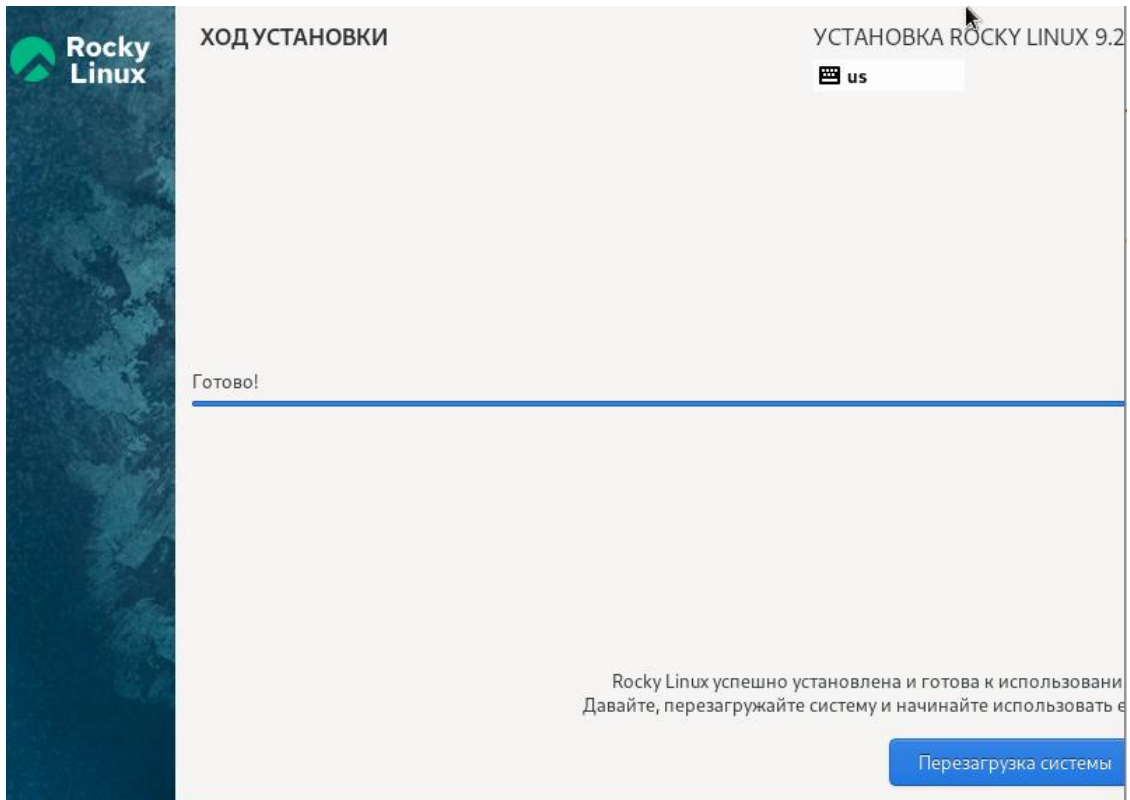
Пароль

10) В выборе места установки ставим наш диск.



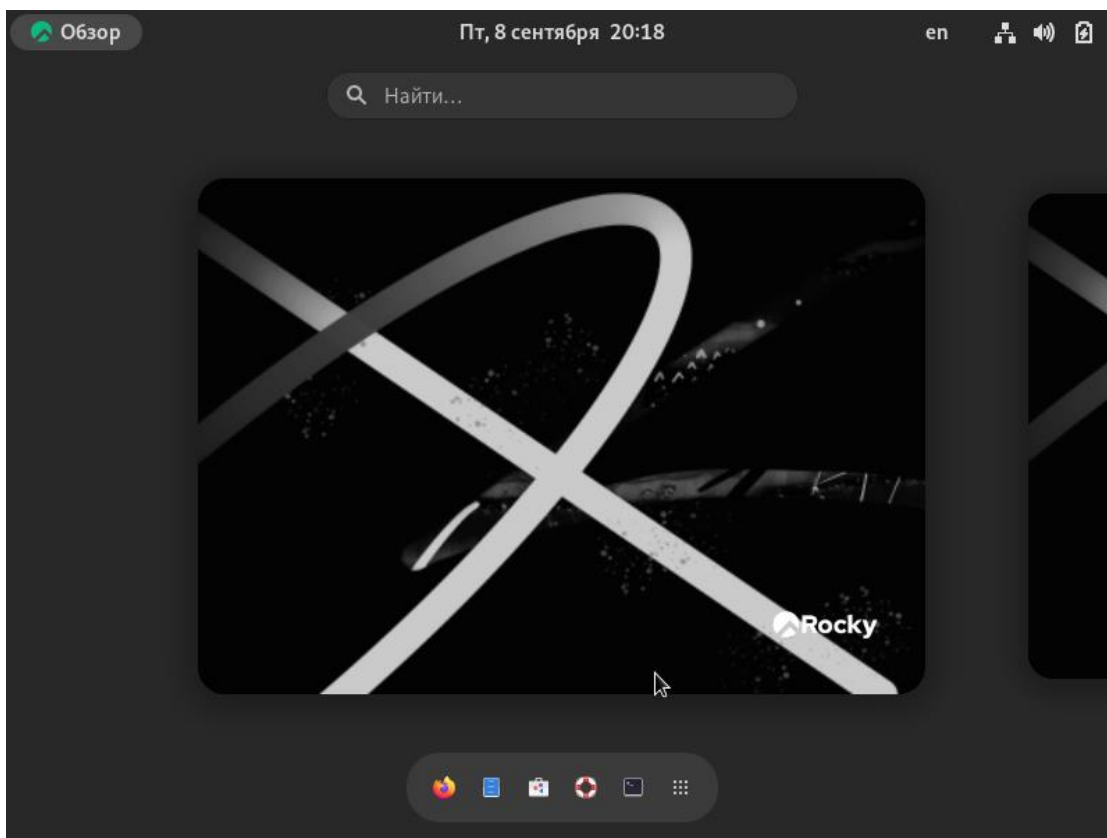
Место установки

11) Установка закончена. Перезагружаем.



Перезагрузка

12) Нас встречает система Rocky Linux. Установка завершена! Осталось подключить образ диска дополнений гостевой ОС.



Rocky Linux

ДОМАШНЕЕ ЗАДАНИЕ №1:

1) В окне терминала проанализировала последовательность загрузки системы, выполнив команду `dmesg`. Можно просто просмотреть вывод этой команды: `dmesg | less`


```
[ 0.000000] Linux version 5.14.0-284.11.1.el9_2.x86_64 (mockbuild@iadi-prod-build001.bld.equ.rockylinux.org) (gcc (GCC) 11.3.1 20221121 (Red Hat 11.3.1-4), GNU ld version 2.35.2-37.el9) #1 SMP PREEMPT_DYNAMIC Tue May 9 17:09:15 UTC 2023
[ 0.000000] The list of certified hardware and cloud instances for Enterprise Linux 9 can be viewed at the Red Hat Ecosystem Catalog, https://catalog.redhat.com.
[ 0.000000] Command line: BOOT_IMAGE=(hd0,msdos1)/vmlinuz-5.14.0-284.11.1.el9_2.x86_64 root=/dev/mapper/rl-root ro resume=/dev/mapper/rl-swap rd.lvm.lv=rl/root rd.lvm.lv=rl/swap rhgb quiet
[ 0.000000] x86/fpu: Supporting XSAVE feature 0x001: 'x87 floating point registers'
[ 0.000000] x86/fpu: Supporting XSAVE feature 0x002: 'SSE registers'
[ 0.000000] x86/fpu: Supporting XSAVE feature 0x004: 'AVX registers'
[ 0.000000] x86/fpu: xstate_offset[2]: 576, xstate_sizes[2]: 256
[ 0.000000] x86/fpu: Enabled xstate features 0x7, context size is 832 bytes, using 'standard' format.
[ 0.000000] signal: max sigframe size: 1776
[ 0.000000] BIOS-provided physical RAM map:
[ 0.000000] BIOS-e820: [mem 0x0000000000000000-0x000000000009fbff] usable
[ 0.000000] BIOS-e820: [mem 0x000000000009fc00-0x000000000009ffff] reserved
[ 0.000000] BIOS-e820: [mem 0x00000000000a0000-0x00000000000fffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000000100000-0x00000000000dffff] usable
[ 0.000000] BIOS-e820: [mem 0x00000000000dffff0000-0x00000000000dffffffffff] ACPI data
[ 0.000000] BIOS-e820: [mem 0x000000000fec000000-0x000000000fec00ffff] reserved
[ 0.000000] BIOS-e820: [mem 0x000000000fee000000-0x000000000fee00ffff] reserved
[ 0.000000] BIOS-e820: [mem 0x000000000ffc000000-0x000000000fffffffff] reserved
[ 0.000000] BIOS-e820: [mem 0x000000001000000000-0x00000000011fffffffff] usable
[ 0.000000] NX (Execute Disable) protection: active
[ 0.000000] SMBIOS 2.5 present.
[ 0.000000] DMI: innotek GmbH VirtualBox/VirtualBox, BIOS VirtualBox 12/01/2006
[ 0.000000] Hypervisor detected: KVM
[ 0.000000] kvm-clock: Using mrs 4b564d01 and 4b564d00
[ 0.000002] kvm-clock: using sched offset of 4944891169 cycles
[ 0.000003] clocksource: kvm-clock: mask: 0xffffffffffffff max_cycles: 0x1cd42e4dffb, max_idle_ns: 881590591483 ns
[ 0.000005] tsc: Detected 2688.002 MHz processor
[ 0.000690] e820: update [mem 0x000000000-0x000000fff] usable ==> reserved
[ 0.000701] e820: remove [mem 0x000a00000-0x0000fffff] usable
[ 0.000704] last_pfn = 0x120000 max_arch_pfn = 0x400000000
[ 0.000712] Disabled
[ 0.000713] x86/PAT: MTRRs disabled, skipping PAT initialization too.
[ 0.000714] CPU MTRRs all blank - virtualized system.
[ 0.000716] x86/PAT: Configuration [0-7]: WB WT UC- UC WB WT UC- UC
[ 0.000719] last_pfn = 0xdffff max_arch_pfn = 0x400000000
[ 0.000764] found SMP MP-table at [mem 0x0009ffff0-0x0009fffff]
[ 0.01049] RAMDISK: [mem 0x30fa2000-0x347c8fff]
[ 0.01054] ACPI: Early table checksum verification disabled
[ 0.01055] ACPI: RSDP 0x0000000000005000 00000004 (v00 VBOX -)
```

dmesg

Можно использовать поиск с помощью `grep:dmesg | grep -i "то, что ищем"` Команда `grep` используется для поиска данных. а. Версия ядра Linux (Linux version).

```
eatikhonova@eatikhonova-VirtualBox: ~
eatikhonova@eatikhonova-VirtualBox:~$ drep:dmesg|grep-i
grep-i: command not found
drep:dmesg: command not found
eatikhonova@eatikhonova-VirtualBox:~$ ^C
eatikhonova@eatikhonova-VirtualBox:~$ ^C
eatikhonova@eatikhonova-VirtualBox:~$ grep:dmesg|grep -i
Usage: grep [OPTION]... PATTERNS [FILE]...
Try 'grep --help' for more information.
grep:dmesg: command not found
eatikhonova@eatikhonova-VirtualBox:~$ dmesg | grep -i "Linux version"
[ 0.000000] Linux version 5.8.0-55-generic (buildd@lgw01-amd64-050) (gcc (Ubuntu 9.3.0-17ubuntu1-20.04) 9.3.0, GNU ld (GNU Binutils for Ubuntu) 2.34) #62~20.04.1-Ubuntu SMP Wed Jun 2 08:55:04 UTC 2021 (Ubuntu 5.8.0-55.62-20.04.1-generic 5.8.18)
eatikhonova@eatikhonova-VirtualBox:~$
```

Версия ядра Linux

b. Частота процессора (Detected Mhz processor).

```
eatikhonova@eatikhonova-VirtualBox:~$ dmesg | grep -i "MHz"
[ 0.000020] tsc: Detected 1800.003 MHz processor
[ 3.012384] e1000 0000:00:03:00 eth0: (PCI:33MHz:32-bit) 08:00:27:44:3b:ef
eatikhonova@eatikhonova-VirtualBox:~$
```

Частота процессора

c. Модель процессора (CPU0)

```
eatikhonova@eatikhonova-VirtualBox:~$ dmesg | grep -i "CPU0"
[ 0.544423] smpboot: CPU0: Intel(R) Core(TM) i5-8250U CPU @ 1.60GHz (family: 0x6, model: 0x8e, stepping: 0xa)
eatikhonova@eatikhonova-VirtualBox:~$
```

Модель процессора (CPU0)

d. Объем доступной оперативной памяти (Memory available)

```
eatikhonova@eatikhonova-VirtualBox:~$ dmesg | grep -i "Memory"
[ 0.265578] ACPI: Reserving FACP table memory at [mem 0x7fff00f0-0x7fff01e3]
[ 0.265579] ACPI: Reserving DSDT table memory at [mem 0x7fff0470-0x7fff2794]
[ 0.265581] ACPI: Reserving FACS table memory at [mem 0x7fff0200-0x7fff023f]
[ 0.265582] ACPI: Reserving FACS table memory at [mem 0x7fff0200-0x7fff023f]
[ 0.265583] ACPI: Reserving APIC table memory at [mem 0x7fff0240-0x7fff0293]
[ 0.265585] ACPI: Reserving SSDT table memory at [mem 0x7fff02a0-0x7fff046b]
[ 0.265595] check: Scanning 1 areas for low memory corruption
[ 0.267156] Early memory node ranges
[ 0.276702] PM: hibernation: Registered nosave memory: [mem 0x00000000-0x00000fff]
[ 0.276704] PM: hibernation: Registered nosave memory: [mem 0x0009f000-0x0009ffff]
[ 0.276705] PM: hibernation: Registered nosave memory: [mem 0x000a0000-0x000effff]
[ 0.276706] PM: hibernation: Registered nosave memory: [mem 0x000f0000-0x000fffff]
[ 0.291368] Memory: 1975824K/2096696K available (14339K kernel code, 2537K rdata, 5456K rodata, 2648K init, 4912K bss, 120872K reserved, 0K cma-reserved)
[ 0.425596] Freeing SMP alternatives memory: 40K
[ 0.552979] x86/mm: Memory block size: 128MB
[ 1.555584] Freeing initrd memory: 51684K
[ 1.555775] check: Scanning for low memory corruption every 60 seconds
[ 1.789794] Freeing unused decrypted memory: 2040K
[ 1.790751] Freeing unused kernel image (initmem) memory: 2648K
[ 1.791981] Freeing unused kernel image (text/rodata gap) memory: 2044K
[ 1.792257] Freeing unused kernel image (rodata/data gap) memory: 688K
[ 8.123906] [drm] Max dedicated hypervisor surface memory is 507904 kiB
[ 8.123907] [drm] Maximum display memory size is 16384 kiB
[ 8.168638] [TTM] Zone kernel: Available graphics memory: 1017484 KiB
eatikhonova@eatikhonova-VirtualBox:~$
```

Объем доступной оперативной памяти

e. Тип обнаруженного гипервизора (Hypervisor detected).

```
eatikhonova@eatikhonova-VirtualBox:~$ dmesg | grep -i "Hypervisor detected"
[ 0.000000] Hypervisor detected: KVM
eatikhonova@eatikhonova-VirtualBox:~$
```

Объем доступной оперативной памяти

f. Тип файловой системы корневого раздела. Последовательность монтирования файловых систем

```
eatikhonova@eatikhonova-VirtualBox:~$ dmesg | grep -i "Mount"
[ 0.400574] Mount-cache hash table entries: 4096 (order: 3, 32768 bytes, linear)
[ 0.400578] Mountpoint-cache hash table entries: 4096 (order: 3, 32768 bytes, linear)
[ 3.783918] EXT4-fs (sda5): mounted filesystem with ordered data mode. Opts: (null)
[ 4.622473] systemd[1]: Set up automount Arbitrary Executable File Formats File System Automount Point.
[ 4.625131] systemd[1]: Mounting Huge Pages File System...
[ 4.626333] systemd[1]: Mounting POSIX Message Queue File System...
[ 4.627525] systemd[1]: Mounting Kernel Debug File System...
[ 4.630253] systemd[1]: Mounting Kernel Trace File System...
[ 4.723308] systemd[1]: Starting Remount Root and Kernel File Systems...
[ 4.838052] systemd[1]: Mounted Huge Pages File System.
[ 4.840271] systemd[1]: Mounted POSIX Message Queue File System.
[ 4.845175] EXT4-fs (sda5): re-mounted. Opts: errors=remount-ro
[ 4.858640] systemd[1]: Mounted Kernel Debug File System.
[ 4.858976] systemd[1]: Mounted Kernel Trace File System.
[ 4.881671] systemd[1]: Finished Remount Root and Kernel File Systems.
[ 5.008997] systemd[1]: Mounting FUSE Control File System...
[ 5.060809] systemd[1]: Mounting Kernel Configuration File System...
[ 5.178122] systemd[1]: Mounted FUSE Control File System.
[ 5.379472] systemd[1]: Mounted Kernel Configuration File System.
[ 5.737301] systemd[1]: Mounting Mount unit for core18, revision 1988...
[ 5.738966] systemd[1]: Mounting Mount unit for core18, revision 2066...
[ 5.762441] systemd[1]: Mounting Mount unit for gnome-3-34-1804, revision 66...
[ 5.777295] systemd[1]: Mounting Mount unit for gnome-3-34-1804, revision 72...
[ 5.821127] systemd[1]: Mounting Mount unit for gtk-common-themes, revision 1514...
[ 5.839934] systemd[1]: Mounting Mount unit for gtk-common-themes, revision 1515...
[ 5.858708] systemd[1]: Mounting Mount unit for snap-store, revision 542...
[ 5.898432] systemd[1]: Mounting Mount unit for snap-store, revision 547...
[ 5.919002] systemd[1]: Mounting Mount unit for snapd, revision 12057...
[ 5.957840] systemd[1]: Mounting Mount unit for snapd, revision 12159...
eatikhonova@eatikhonova-VirtualBox:~$
```

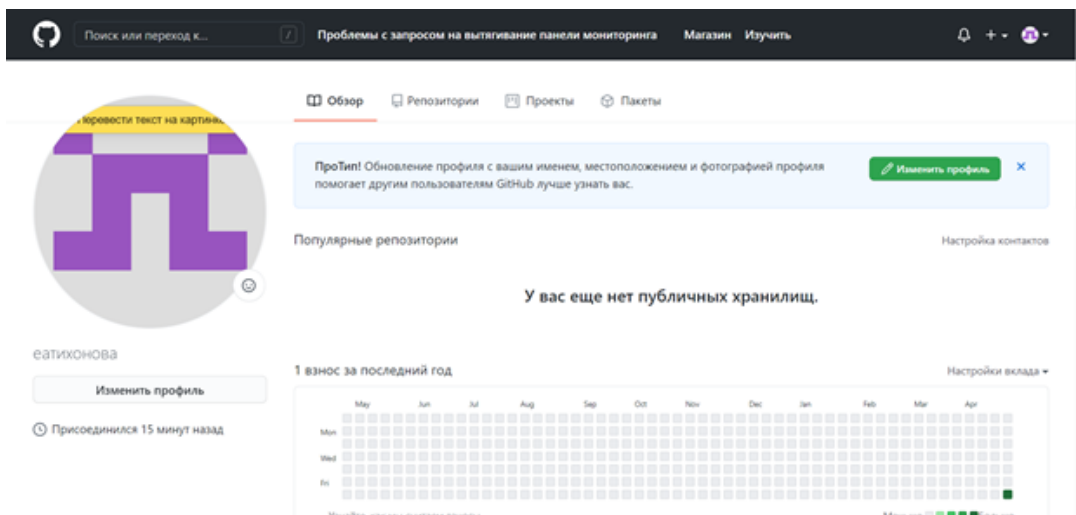
Объем доступной оперативной памяти

КОНТРОЛЬНЫЕ ВОПРОСЫ №1

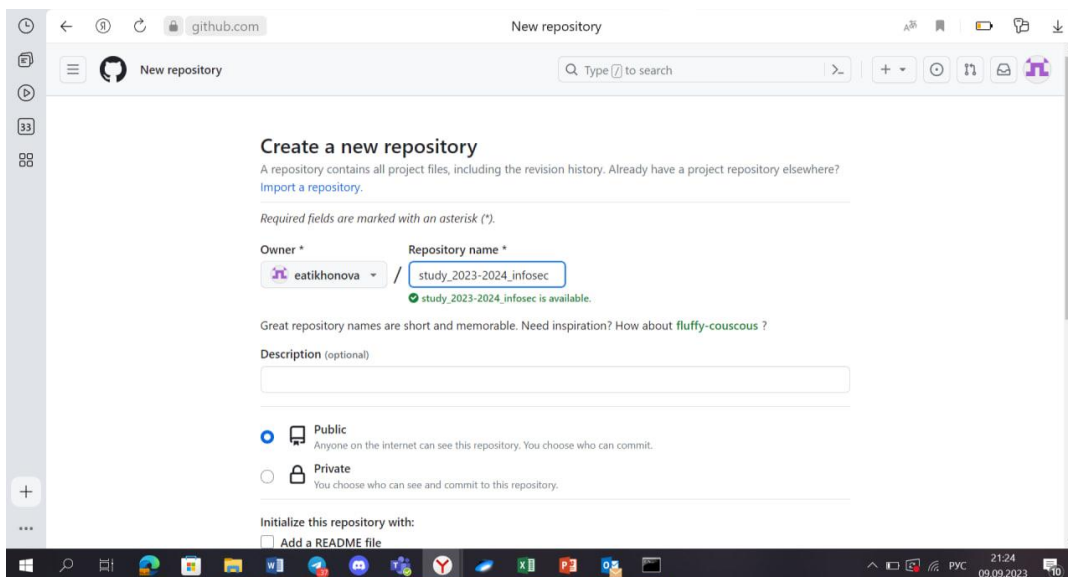
1) Учётная запись пользователя содержит сведения, необходимые для идентификации пользователя при подключении к системе, такие как имя пользователя, имя хоста и пароль. 2) Команды терминала: а. Для получения справки используется ключ `-help` или команда `man`. Например, `ls -help` или `man ls`. б. Для перемещения по файловой системе используется команда `cd`. Например `cd ~`. в. Для просмотра содержимого каталога используется команда `ls`. Например `ls ~/work`. г. Для определения объёма каталога используется команда `du`. д. Для создания каталогов используется `mkdir`, для удаления пустых каталогов используется `rmdir`. Для создания файлов используется `touch`, для удаления файлов и каталогов используется `rm`. е. Для задания прав используется команда `chmod`. Например, `chmod u-w test.txt`. ж. Для просмотра истории команд используется команда `history`. 3) Файловая система — часть ОС, которая обеспечивает чтение и запись файлов на дисковых носителях информации. а. Ext2 — расширенная файловая система. Данные сначала кэшируются и только потом записываются на диск. б. Ext3 и Ext4 — журналируемые файловые системы. Осуществляется хранение в виде журнала со списком изменений, что помогает сохранить целостность при сбоях. в. XFS — высокопроизводительная журналируемая файловая система, рассчитанная для работы на дисках большого объёма. 4) Для просмотра подмонтированных в ОС файловых систем необходимо использовать команду `findmnt`. 5) Для удаления зависшего процесса используется команда `kill PID` или `killall название`.

Часть 2. Работа с GitHub

1) Первым делом, мы регистрируемся на сайте github.com. Так как мы не в первый раз имеем дело с github, то аккаунт уже готов.



Создаем репозиторий:



Знакомство с github

2) Первоначальная настройка для работы с github.

в) Вводим имя репозитория. Нажимаем на галочку на create readme file. Рис5

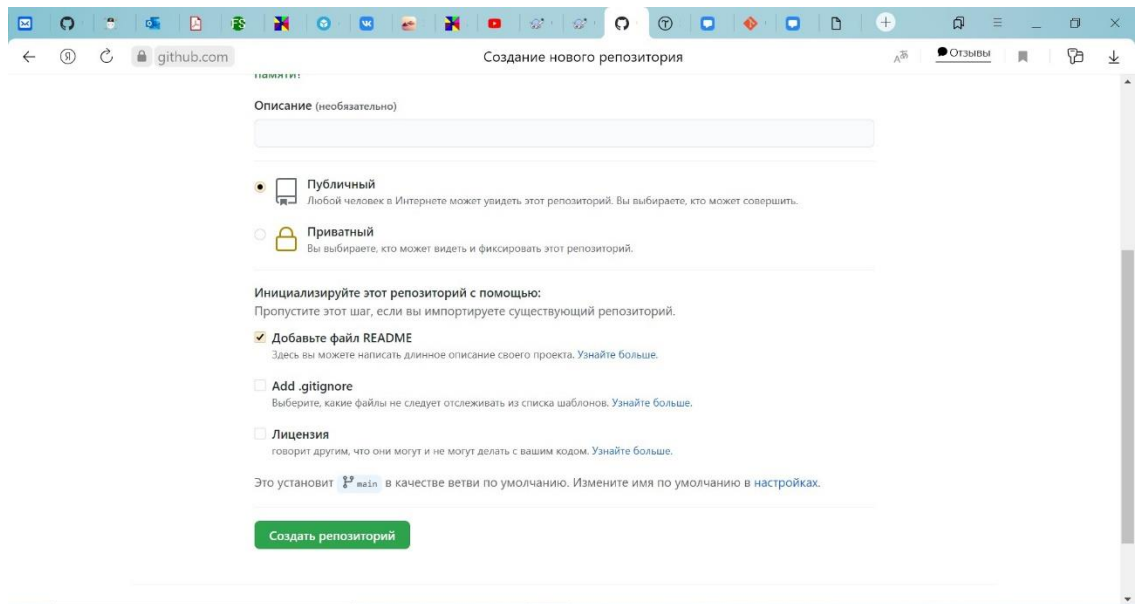


рис5

- 4) Связываем репозитории на ПК и на гитхабе
а) Нажимаем ПКМ. выбираем на git clone. рис6

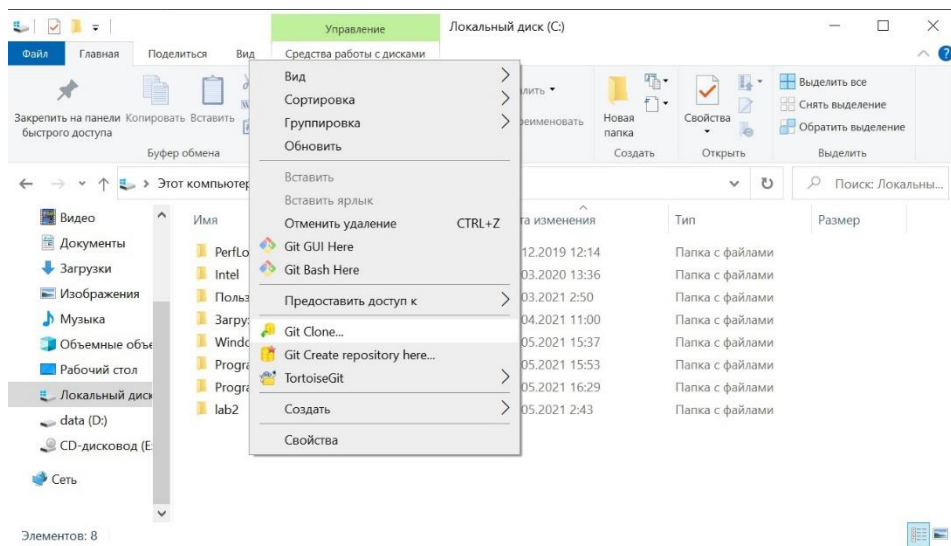


Рис6

- б) В строчку url вставляем ссылку на репозиторий. Рис7

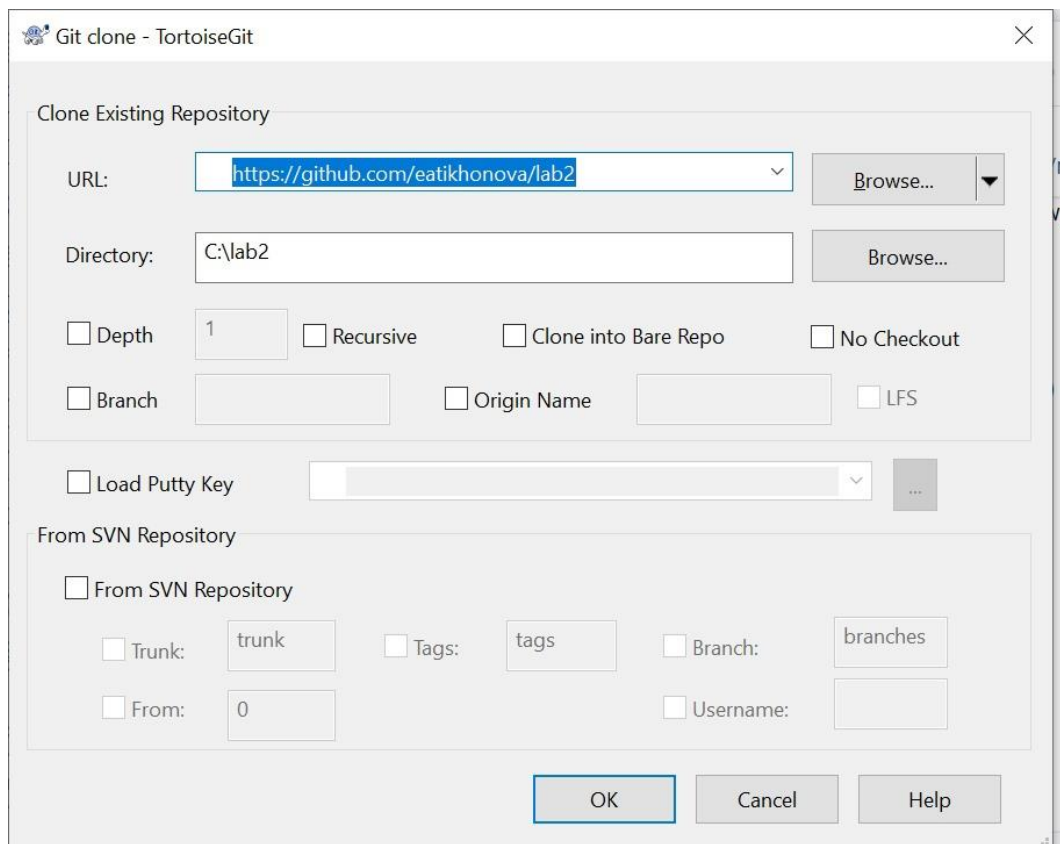


рис7

в) На компьютер скачался репозиторий с гитхаба. Рис8

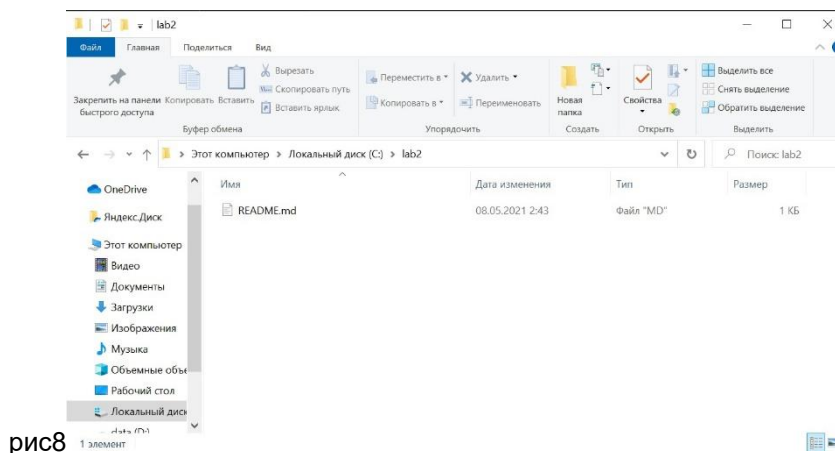


рис8

5) Чтобы файлы с компьютера отображались в репозитории делаем следующее:
а) Нажимаем пкм на репозиторий, выбираем git commit. рис9

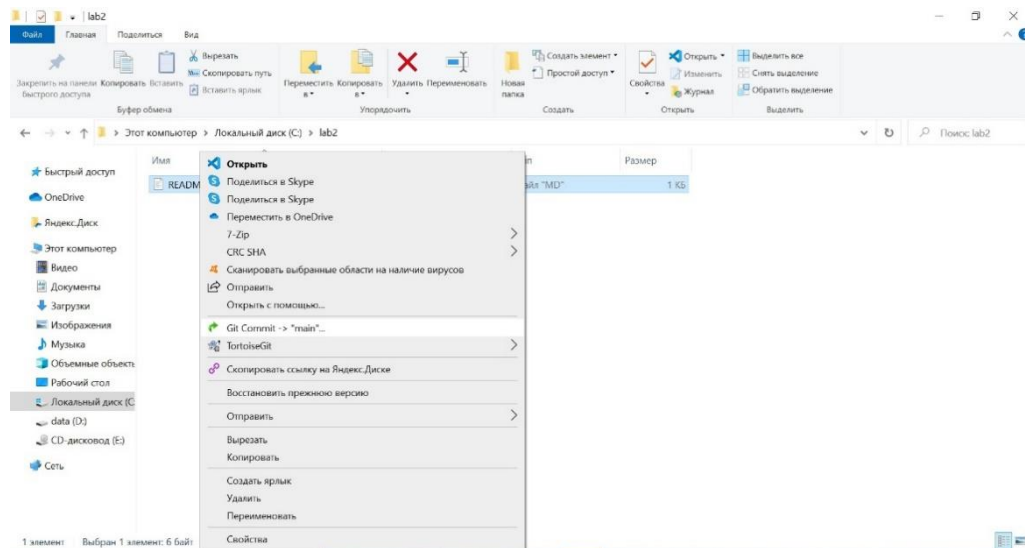


Рис9

б) Отмечаем галочкой все файлы снизу, жмем на кнопку commit. Рис10

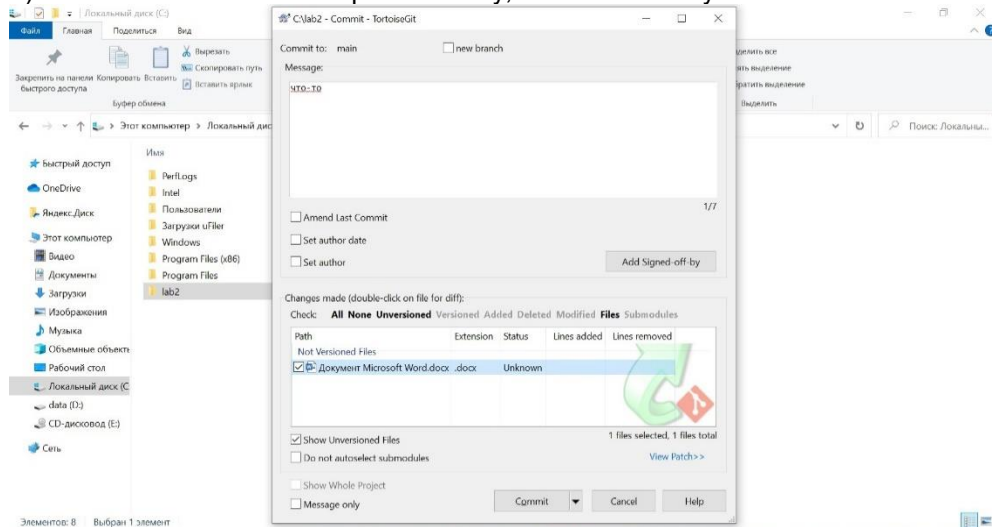


рис10

в) Нажимаем на кнопку Push и файлы появляются на гитхаб.

Коммит

Вторая часть закончена. Мы связали наш репозиторий с репозиторием на github.com.

КОНТРОЛЬНЫЕ ВОПРОСЫ №2:

1) Система контроля версий Git представляет собой набор программ командной строки. Доступ к ним можно получить из терминала посредством ввода команды `git` 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 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) У Git две основные 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 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 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`

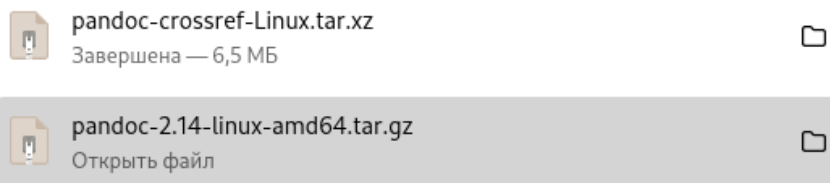
8)Использования `git` при работе с локальными репозиториями (добавления текстового документа в локальный репозиторий): `git add hello.txt git commit -am 'Новый файл'`

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 `ewrvisov`. Для этого сначала нужно получить список имеющихся 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`

Часть 3. Создание Markdown файла

1)Для того, чтобы из Markdown получить PDF и DOCX файлы, требуется скачать `pandoc` и `pandoc-crossref`. Это непростая задача, пришлось скачивать их с `github.com` старой версии 2.14. Переместить исполняемые файлы в папку `~/bin`. Сперва скачаем zip-файлы из гитхабов `jgm` и `lierdakil`. Смотрим ветки и выбираем всё для `pandoc 2.14`.



pandoc

2)Таким образом, как на скриншоте, мы перемещаем в корневую папку `pandoc`. Аналогично с `pandoc-crossref`.

```
~]$ cd pandoc-2.14-linux-amd64/
```

```
]$ cd pandoc-2.14/
```

```
]$ sudo cp -R pandoc /bin
```

Установка pandoc

3) В консольной строке пишем make и получаем наши файлы. Примечание: компиляцию файлов я делал в Ubuntu, так как в Rocky это делается немножко криво. Нужно не забыть написать команду `sudo apt-get install texlive-full`, чтобы все нужные шрифты были готовы.

Либо для конвертации в pdf заходим в Markdown в меню extensions (оно справа). В поиске пишем Markdown PDF. Скачиваем.

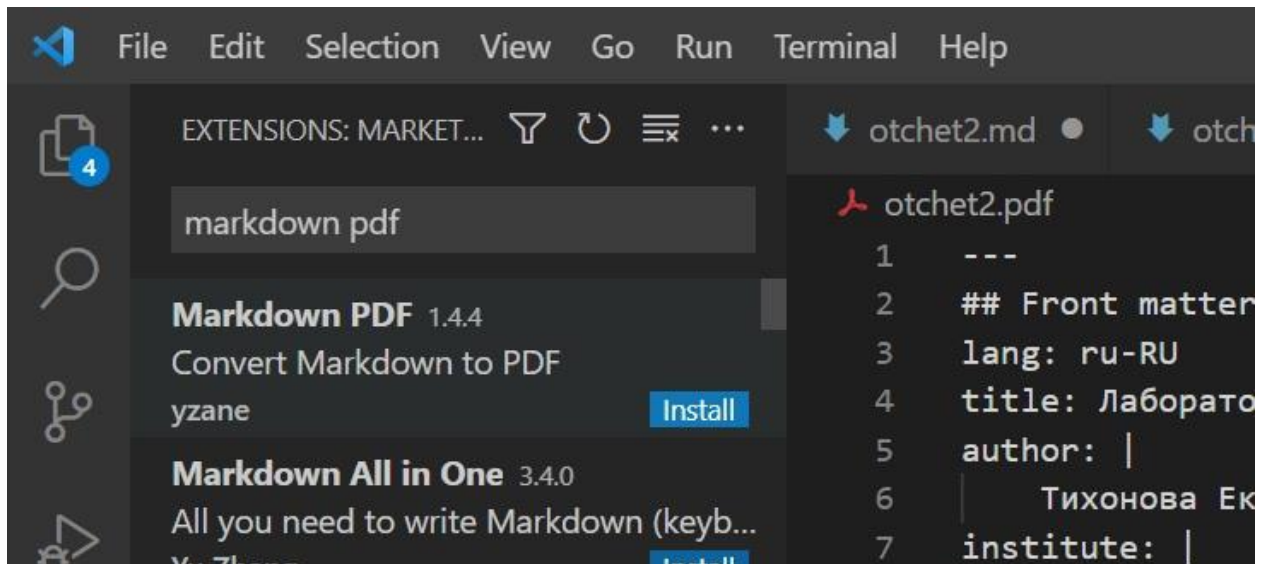


рис6

7) Пишем текст лабораторной работы. Когда он готов, нажимаем `f1`, в появившемся окошечке пишем `export`, выбираем `export to pdf`. рис7

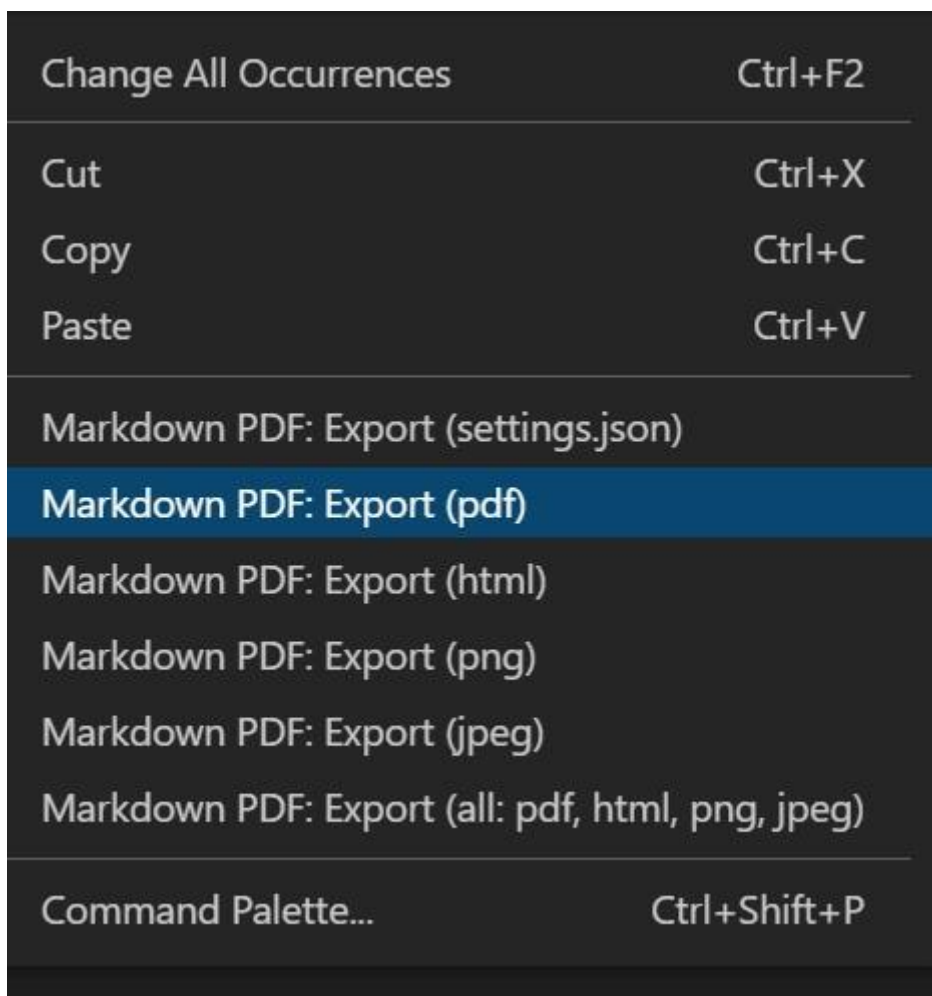


рис7

Выводы

Я установила VirtualBox, поставила на виртуальную машину Rocky Linux. Разобралась с системой контроля версий и github. Скомпилировала файлы из формата md в pdf и docx.