**Министр науки и высшего образования Российской̆ Федерации**

**Федеральное государственное автономное образовательное учреждение высшего образования**

**«Национальный исследовательский университет ИТМО»**

Факультет информационных технологий и программирования

Лабораторная работа № 3

Установка программного обеспечения

Выполнил студент группы № M3311

Ершова Мария

**Задачи:**

1. Научиться получать информацию об установленных и доступных пакетах и устанавливать пакеты программного обеспечения из репозиториев в Linux.
2. Научиться устанавливать программные продукты из исходных кодов в Linux.
3. Научится конфигурировать менеджер пакетов и создавать собственные репозитории.

**Задание:**

Выполните указанные ниже действия и запишите все использовавшиеся команды с параметрами и изменения в файлах в отдельный файл протокола. Для удобства выполнения работы рекомендуется настроить доступ к виртуальной машине по ssh.

**Порядок действий:**

1. Выведите список всех подключенных репозитариев:

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Для каждого репозитория в системе должен быть конфигурационный файл. Эти файлы располагаются в каталоге: /etc/apt/sources.list.d/ для Debian или в общем файле /etc/apt/sources.list – поэтому мы и пишем cat /etc/apt/sources.list

1. Обновите локальные индексы пакетов в менеджере пакетов:

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Ура, ура, все сделала команда apt update!

1. Выведите информацию о метапакете build-essential:

apt show build-essential

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

1. Установите метапакет build-essential, при этом определите какие компоненты будут установлены, а какие обновлены:

apt install build-essential – ура, оно все устанавливает:

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Ну фотка это так, показать, что все рил установилось.

1. Найдите пакет, в описании которого присутствует строка «clone with a bastard algorithm»:

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

В итоге нашли пакет bastet.

--full выводит всю инфу так сказать.

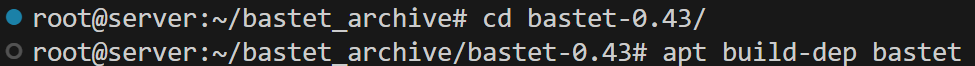
1. Скачайте в отдельную директорию в домашнем каталоге архив с исходными кодами найденного в п.5 пакета:

Изображение выглядит как текст, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.

Для начала была создана отдельная директория /bastet\_archive, после чего туда мы скачали сам bastet с помощью apt source bastet.

1. Установите пакет из исходных кодов, скаченных в п.6:



Для начала заходим в bastet-0.43, и билдим все зависимости.

Ну и теперь пишем make.

Делаем ./bastet:

Изображение выглядит как текст, снимок экрана, диаграмма, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Вот такой прикол, ну по логике вещей надо было дальше сделать make install, но что-то там кому-то не нравится и вот. В итоге было сделано cat INSTALL, прописана оттуда строчка cp bastet /usr/local/bin и теперь мы можем где угодно писать bastet и играть в тетрис.

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

1. Если в конфигурационном файле пакета нет параметров установки пакета в систему, то добавьте их, так, чтобы пакет устанавливался в /usr/local/bin и на него назначались права ---rwxr-xr-x. Проверьте выполнения этих директив:

Для выполнения этого пункта надо зайти в файл Makefike и прописать:

install:

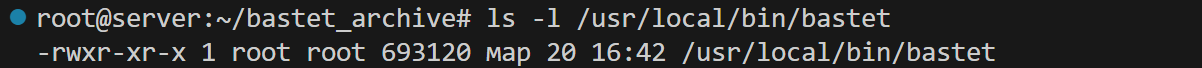
install -m 0755 bastet /usr/local/bin

Изображение выглядит как текст, снимок экрана, программное обеспечение, Операционная система

Контент, сгенерированный ИИ, может содержать ошибки.

1. Проверьте, что любой пользователь может запускать установленный пакет, но не тратьте на это более 5 минут:

ls -l /usr/local/bin/bastet сделает нам вайбик и покажет права:



1. Создайте файл task10.log, в который выведите список всех установленных пакетов:

Изображение выглядит как текст, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.

Создаем в /root директорию Lab3, переходим в неё и пишем:

dpkg --get-selections > task10.log

1. Создайте файл task11.log, в который выведите список всех пакетов (зависимостей), необходимых для установки и работы компилятора gcc:

Изображение выглядит как текст, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.

Плачем и хнычем. Я так есть хочу, жуть просто, я так мало поела, ужас. А apt depends gcc делает все что нам надо.

1. Создайте файл task12.log, в который выведите список всех пакетов (зависимостей), установка которых требует установленного пакета libgpm2:

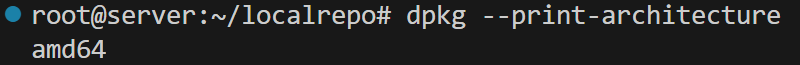
Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Ну крайне схожая тема что и в предыдущем пункте – наслаждаемся!

1. Создайте каталог localrepo в домашнем каталоге пользователя root и скопируйте в него c сайта http://snapshot.debian.org/package/htop/ пять разных версий пакета htop. Это можно сделать с помощью wget или просто передав файлы на виртуальную машину используя протокол ssh и утилиту scp:

Ну первым делом проверим какая у нас архитектура системы (ну понятно, что amd64, но я хочу – я буду):



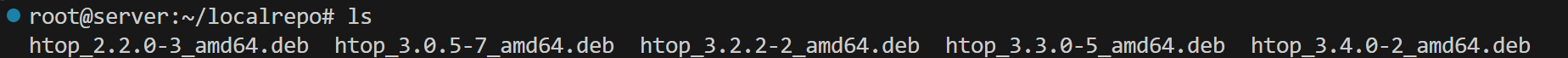
Ну и очев, что устанавливаем версии для amd64.

Ну все понятно: создали localrepo, зашли в неё и с помощью wget копируем 5 разных версий пакета htop:

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Тут пример как все выглядит в итоге:



1. Сгенерируйте в каталоге репозитория файл Packages, который будет содержать информацию о доступных пакетах в репозитории и файл. Создайте файл Release, содержащий описание репозитория. Например:

Origin: My Local Repo

Label: My Local Repo

Suite: stable

Version: 1.0

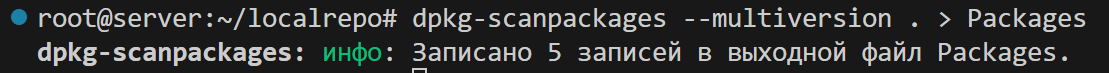
Codename: myrepo

Architectures: amd64

Components: main

Description: My local APT repository

И так, создаем файл Packages:



Посмотрим, что записалось в Packages:

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Теперь генерируем файл Release, который описывает репозиторий:

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Отредактируем этот файл:

nano Release и копипастим штуку из примера, сохраняем и радуемся. Только теперь надо изменить список репозиториев, это можно сделать с помощью apt edit-sources:

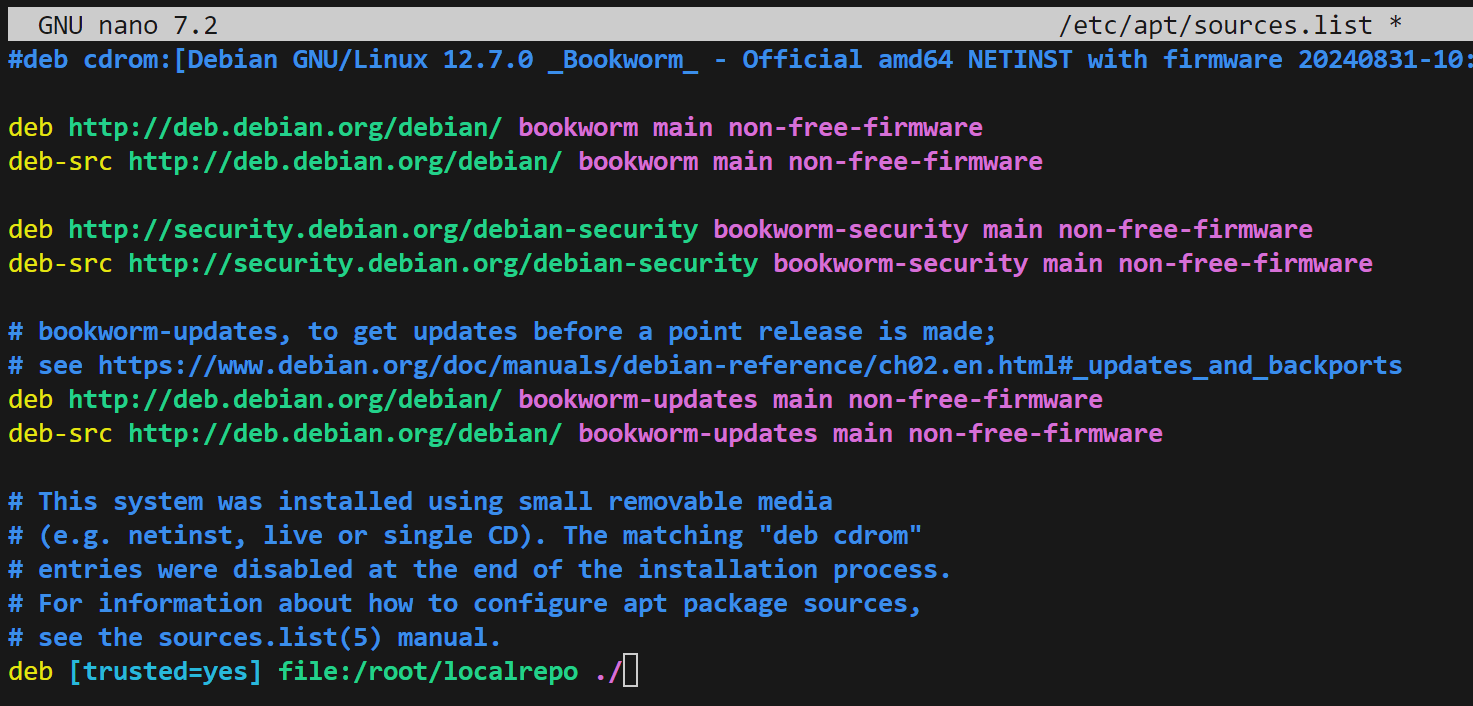
**Изображение выглядит как текст, электроника, снимок экрана, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.**

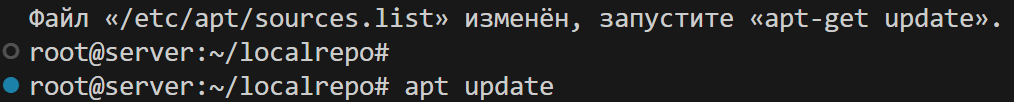
Открылась такая цветастая шняга, значит время добавить строчку:

deb [trusted=yes] file:/root/localrepo ./

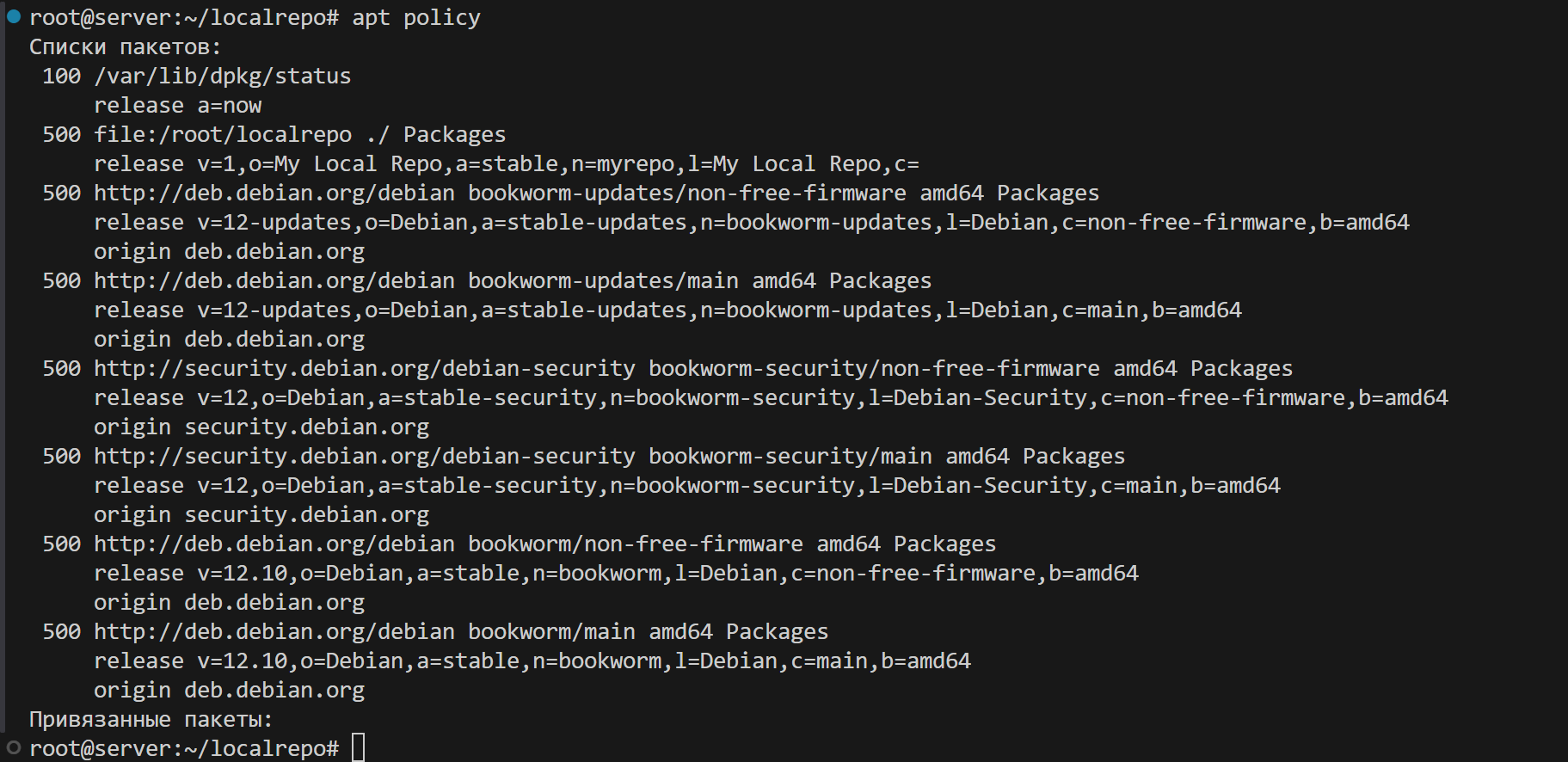
Благодаря этой строке APT увидит пакеты из локального репозитория (ура семпай заметил нас):



Нам уже прямым текстом говорят, что надо сделать, а значит делаем:



Ждем конца выполнения и проверяем (apt policy в помощь):



Ура, ура, успех!

1. Обновите кэш apt

apt update – ну думаю результат не нужен.

1. Выведите список подключенных репозитариях и краткую информацию о них.

cat /etc/apt/sources.list:

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Можно так или снова через apt policy:

Изображение выглядит как текст, снимок экрана, Шрифт, меню

Контент, сгенерированный ИИ, может содержать ошибки.

1. Создайте файл task16.log в который выведите список всех доступных версий htop

apt-cache policy htop > task16.log

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Пипец…

У меня почему-то apt update сносит права для \_apt, решается так:

chown -R root:nogroup /root/localrepo

chmod -R 755 /root/localrepo

1. Установите предпоследнюю версию пакета

Я очень стараюсь установить эту фигню, но вот:

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Я пыталась это фиксить и в итоге скачала с сайта самую последнюю версию:

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Но потом я получила это:

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Я не готова короче рисковать, так что скачаю другое htop (3.3.0-1) :3

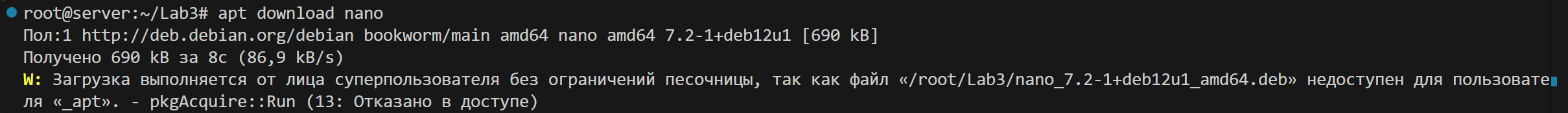
Изображение выглядит как текст, снимок экрана, Шрифт

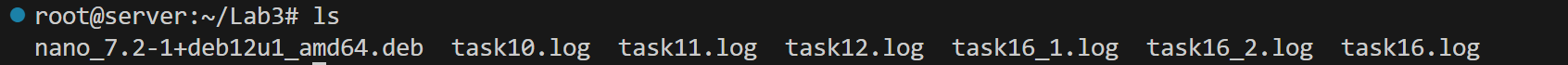
Контент, сгенерированный ИИ, может содержать ошибки.

Пришлось проделать все последние пункты заново.

1. Скачайте из сетевого репозитория пакет nano. Пересоберите пакет таким образом, чтобы после его установки, появлялась возможность запустить редактор nano из любого каталога, введя команду newnano. Для работы с пакетом следует использовать dpkg-deb, а для установки dpkg. В файле протокола работы опишите использованные команды.

Делаем apt download nano:





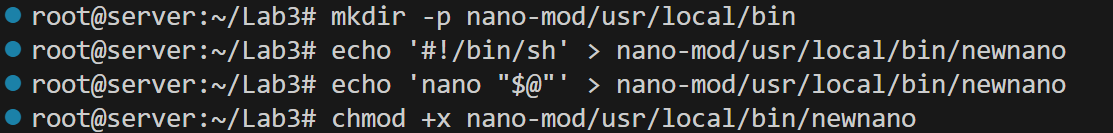
Не спрашивайте, что за странные файлы :3

Теперь делаем временную директорию nano-mod, в которой распакуем пакет:

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Так, создаем директорию nano-mod/usr/local/bin, в ней создаем файл newnano и указываем что это shell скрипт. Потом даем echo 'nano "$@" ' > nano-mod/usr/local/bin/newnano ($@ передаёт все аргументы в nano) и делаем newnano исполняемым файлом:



Пересобираем пакет и теперь у нас есть пересобранный .deb пакет nano-newnano.deb, содержащий newnano ура ура ура ура ура!

А, ну и устанавливаем наш пакет и радуемся жизни:

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

1. Бонусный вопрос с подвохом - что есть в APT?

**Душа**