Front matter

lang: ru-RU title: "Программирование в командном процессоре ОС UNIX. Ветвления и циклы: | Койфман Кирилл Дмитриевич institute: | RUDN date: 2022

Formatting

toc; false slide_level: 2 theme: metropolis header-includes: - \metroset\progressbar=frametitle,sectionpage=progressbar,numbering=fraction\} - \makeatletter' - \beamer@ignorenonframefalse' - \makeatother' aspectratio: 43

section-titles: true

Цель работы.

Изучить основы программирования в оболочке ОС UNIX. Научится писать более сложные командные файлы с использованием логических управляющих конструкций и циклов

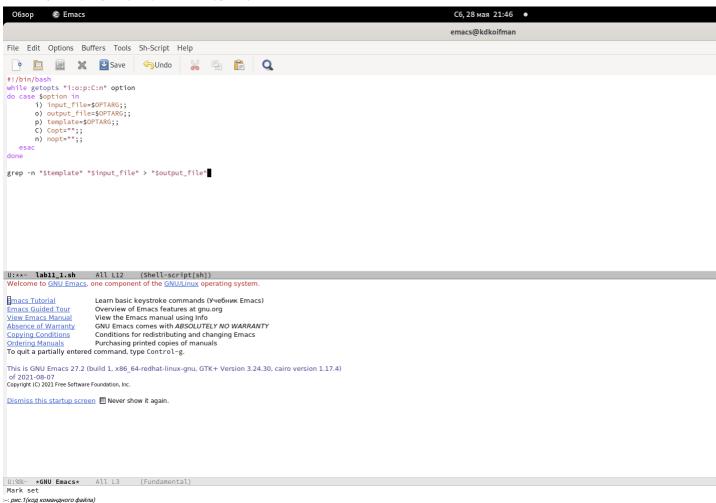
Ход работы.

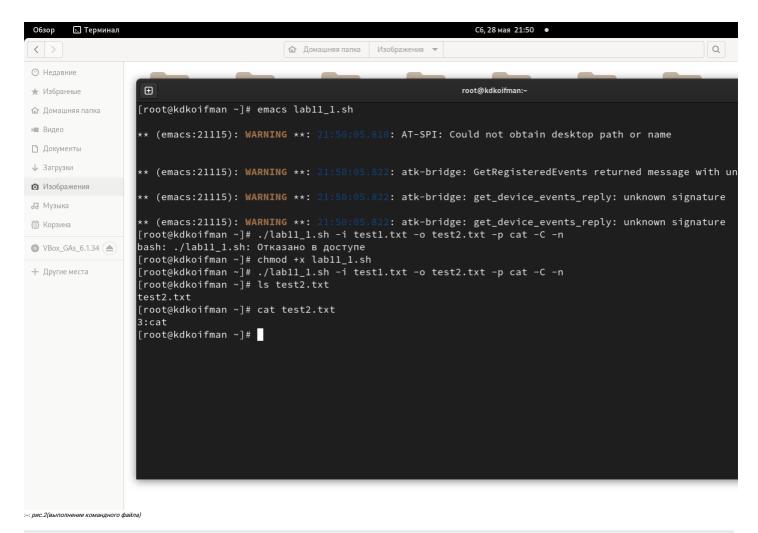
1 задание.

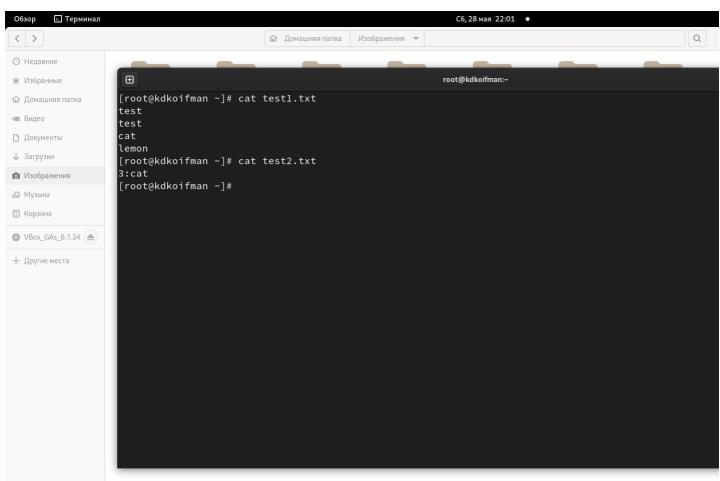
Используя команды getopts grep, напишем командный файл, который анализирует командную строку с ключами:

- -ooutputfile вывести данные в указанный файл;
- -ршаблон указать шаблон для поиска;
- - С различать большие и малые буквы;
- -n выдавать номера строк

А затем ищет в указанном файле нужные строки, определяемые ключом -р (рис.1, 2, 3)



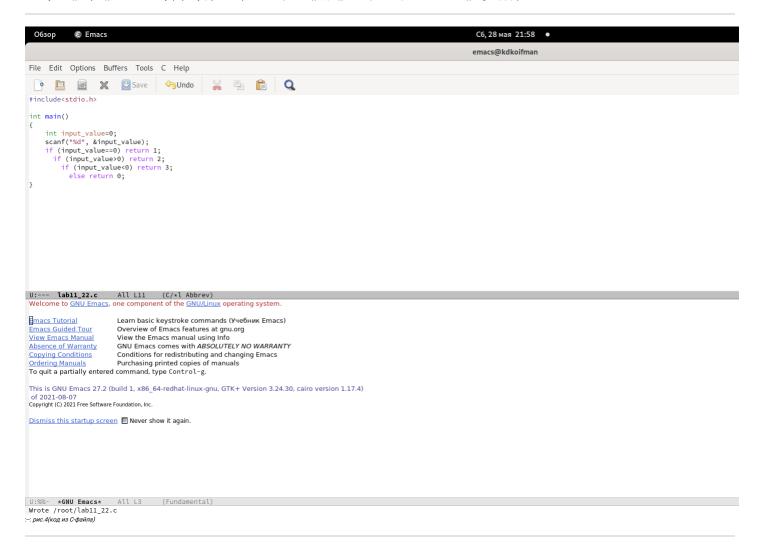


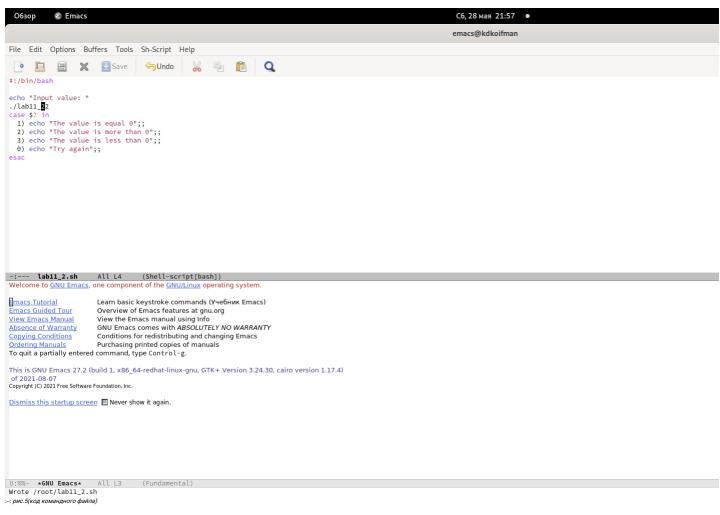


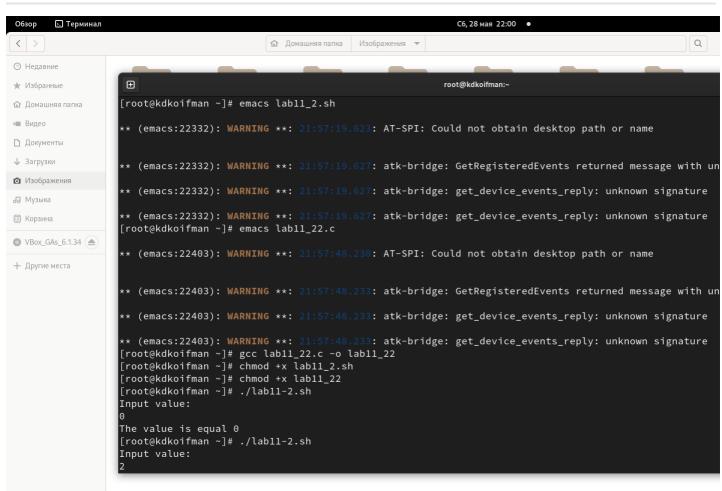
:-: рис.3(содержимое файлов test1.txt и test2.txt)

2 задание.

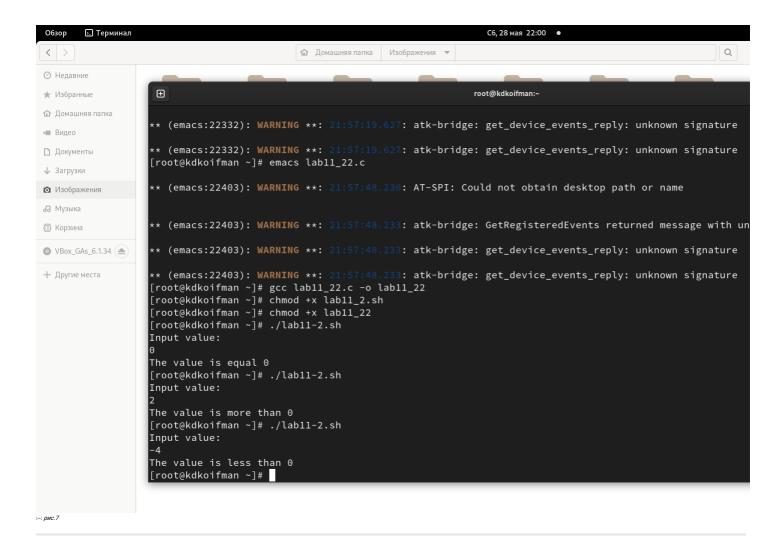
Напишем на языке Си программу, которая вводит число и определяет, является ли оно больше нуля, меньше нуля или равно нулю. Затем программа завершается с помощью функции exit(n), передавая информацию в о коде завершения в оболочку. Командный файл должен вызывать эту программу и, проанализировав с помощью команды \$?, выдать сообщение о том, какое число было введено (рис. 4, 5, 6, 7)







:-: рис.6(выполнение командного файла)



3 задание.

Напишем командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до N (например 1.tmp, 2.tmp, 3.tmp, 4.tmp и т.д.). Число файлов, которые необходимо создать, передаётся в аргументы командной строки. Этот же командный файл должен уметь удалять все созданные им файлы (если они существуют)(рис.8,9)

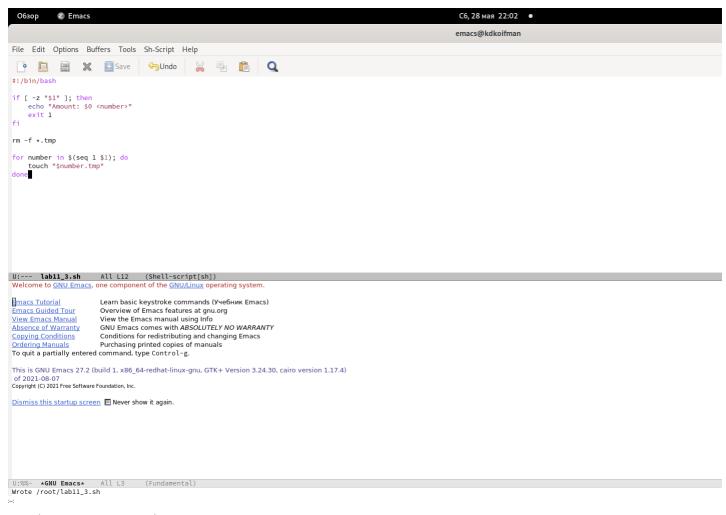
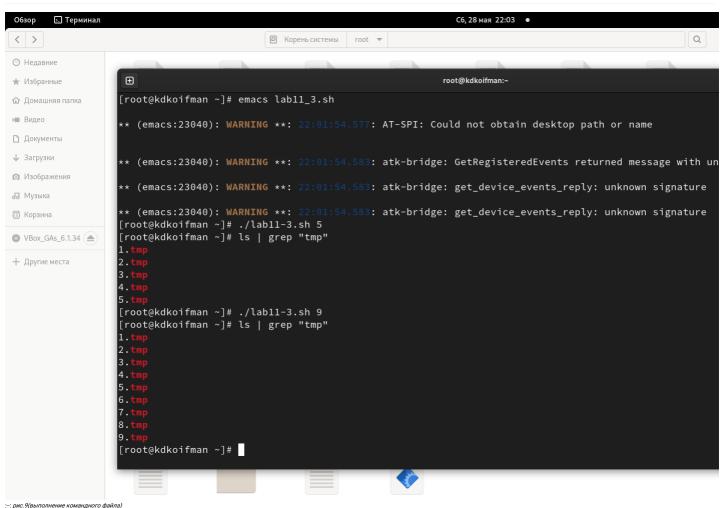
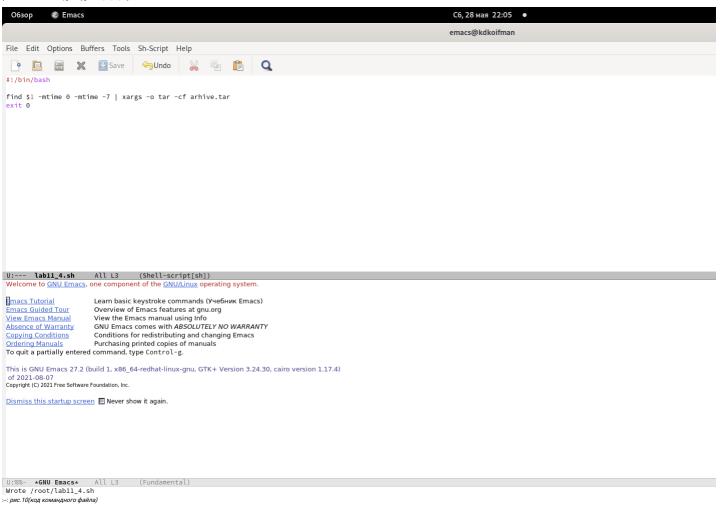


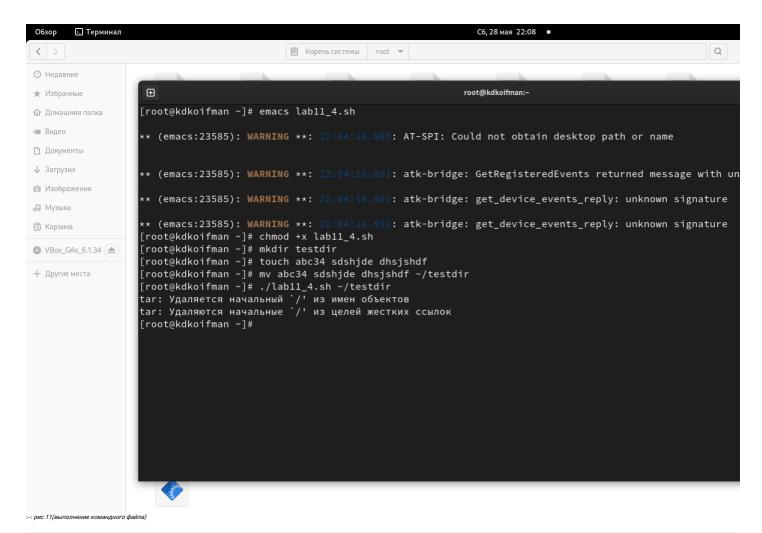
рис.8(код командного файла)



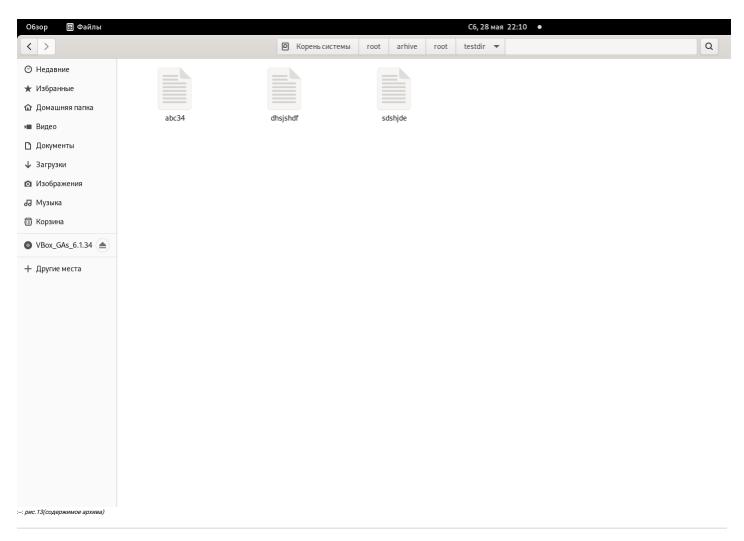
4 задание.

Напишем командный файл, который с помощью команды tar запаковывает в архив все файлы в указанной директории. Модифицируем его так, чтобы запаковывались только те файлы, которые были изменены менее недели тому назад (использовать команду find)(рис.10,11,12,13)





Терминал С6, 28 мая 22:09 О6зор Q ○ Недавние 🖈 Избранные \oplus root@kdkoifman:~ Домашняя папка (emacs:23585): WARNING **: 22:04:16.891: atk-bridge: GetRegisteredEvents returned message with un **Видео** ** (emacs:23585): WARNING **: 22:04:16.891: atk-bridge: get_device_events_reply: unknown signature 🖺 Документы ↓ Загрузки ** (emacs:23585): WARNING **: 22:04:16.891: atk-bridge: get_device_events_reply: unknown signature [root@kdkoifman ~]# chmod +x lab11_4.sh Изображения [root@kdkoifman ~]# mkdir testdir **Л** Музыка [root@kdkoifman ~]# touch abc34 sdshjde dhsjshdf [root@kdkoifman ~]# mv abc34 sdshjde dhsjshdf ~/testdir Корзина [root@kdkoifman ~]# ./labll_4.sh ~/testdir tar: Удаляется начальный `/' из имен объектов tar: Удаляются начальные `/' из целей жестких ссылок [root@kdkoifman ~]# cd ~/testdir + Другие места [root@kdkoifman testdir]# ls abc34 dhsjshdf sdshjde [root@kdkoifman testdir]# cd [root@kdkoifman ~]# ls 1.tmp first lab10-1.sh~ lab11-22 2.tmp first~ lab11_3.sh lab10-2.sh lab12 3.tmp '#foo#' lab11_22.c Makefi 4.tmp foo lab11-22.cpp lab11-22.cpp~ foo~ lab10-3.sh~ 5.tmp cfile.cpp fourth lab11_22.cpp~ conf.txt lab11-2.sh lab12-1.sh 6.tmp lab11-2.sh~ 7.tmp equiplist2 lab11-1.sh lab12-2 8.tmp feathers lab07.sh lab12-2~ second file.cpp lab11_2.sh~ lab12-2.sh 9.tmp lab07.sh~ file.cpp.save anaconda-ks.cfg lab10-1.sh lab11_1.sh~ lab11-3.sh lab12-2.sh~ test1 root@kdkoifman ~]#



Вывод.

В ходе выполнения лабораторной работы мной были усвоены основные навыки программирования в оболочке ОС UNIX, а также написания комплексных командных файлов с использованием логических управляющих конструкций и циклов.