Презентация к лабораторной работе №10

Егорова Юлия Владимировна

[1]RUDN University, Moscow, Russian Federation

Выполнение лабораторной работы

Цель работы

Изучить основы программирования в оболочке ОС UNIX/Linux.

Научиться писать небольшие командные файлы.

Выполнение лабораторной работы

1. Вводим команду man zip, получаем информацию о ней:

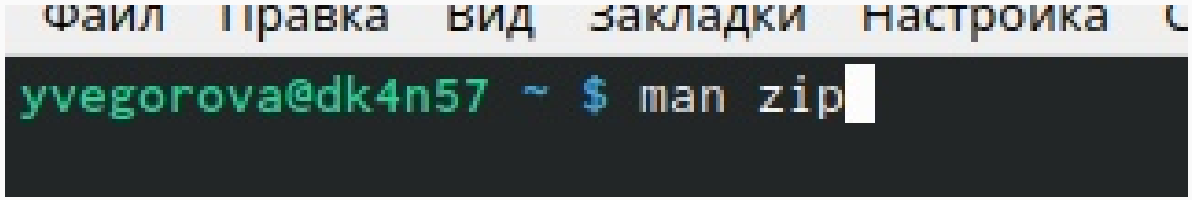


Figure 1: Man zip.

Команда man zip

ZIP(1L)

NAME
zip - package and compress (archive) files

SYNOPSIS
zip [-aABcdDeEfFghjklLmoqrRSTuvVwXyz!@$] [--longoption ...] [-b path] [-n suffixes] [-t date] [-tt date] [zipfile [file ...]] [-xi list]

zipcloak (see separate man page)

zipnote (see separate man page)

zipsplit (see separate man page)

Note: Command line processing in zip has been changed to support long options and handle all options and arguments more consistently. Some old command lines that depend on command line inconsistencies may

DESCRIPTION
zip is a compression and file packaging utility for Unix, VMS, MSDOS, OS/2, Windows 9x/NT/XP, Minix, Atari, Macintosh, Amiga, and Acorn RISC OS. It is analogous to a combination of the Unix commands tar(
and is compatible with PKZIP (Phil Katz's ZIP for MSDOS systems).

A companion program (unzip(1L)) unpacks zip archives. The zip and unzip(1L) programs can work with archives produced by PKZIP (supporting most PKZIP features up to PKZIP version 4.6), and PKZIP and PKUNZIP
chives produced by zip (with some exceptions, notably streamed archives, but recent changes in the zip file standard may facilitate better compatibility). zip version 3.0 is compatible with PKZIP 2.04 and
Zip64 extensions of PKZIP 4.5 which allow archives as well as files to exceed the previous 2 GB limit (4 GB in some cases). zip also now supports bzip2 compression if the bzip2 library is included when
Note that PKUNZIP 1.10 cannot extract files produced by PKZIP 2.04 or zip 3.0. You must use PKUNZIP 2.04g or unzip 5.0p1 (or later versions) to extract them.

See the EXAMPLES section at the bottom of this page for examples of some typical uses of zip.

Large Archives and Zip64. zip automatically uses the Zip64 extensions when files larger than 4 GB are added to an archive, an archive containing Zip64 entries is updated (if the resulting archive still
size of the archive will exceed 4 GB, or when the number of entries in the archive will exceed about 64K. Zip64 is also used for archives streamed from standard input as the size of such archives are not k
but the option -fz- can be used to force zip to create PKZIP 2 compatible archives (as long as Zip64 extensions are not needed). You must use a PKZIP 4.5 compatible unzip, such as unzip 6.0 or later, to e
the Zip64 extensions.

In addition, streamed archives, entries encrypted with standard encryption, or split archives created with the pause option may not be compatible with PKZIP as data descriptors are used and PKZIP at the tim
does not support data descriptors (but recent changes in the PKWare published zip standard now include some support for the data descriptor format zip uses).

Mac OS X. Though previous Mac versions had their own zip port, zip supports Mac OS X as part of the Unix port and most Unix features apply. References to "MacOS" below generally refer to MacOS version
Support for some Mac OS features in the Unix Mac OS X port, such as resource forks, is expected in the next zip release.

For a brief help on zip and unzip, run each without specifying any parameters on the command line.

USE
The program is useful for packaging a set of files for distribution; for archiving files; and for saving disk space by temporarily compressing unused files or directories.

The zip program puts one or more compressed files into a single zip archive, along with information about the files (name, path, date, time of last modification, protection, and check information to verify
An entire directory structure can be packed into a zip archive with a single command. Compression ratios of 2:1 to 3:1 are common for text files. zip has one compression method (deflation) and can also s
compression. (If bzip2 support is added, zip can also compress using bzip2 compression, but such entries require a reasonably modern unzip to decompress. When bzip2 compression is selected, it replaces de
fault method.) zip automatically chooses the better of the two (deflation or store or, if bzip2 is selected, bzip2 or store) for each file to be compressed.

Figure 2: Информация о man zip.

Команда man bzip2

2. Команду man bzip2 и также получаем информацию:

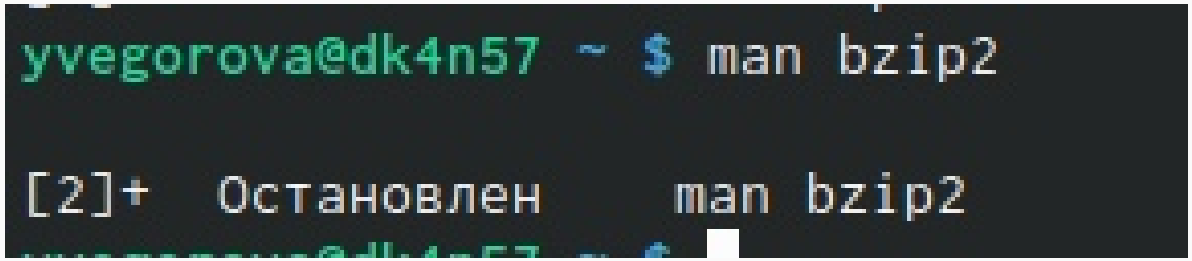
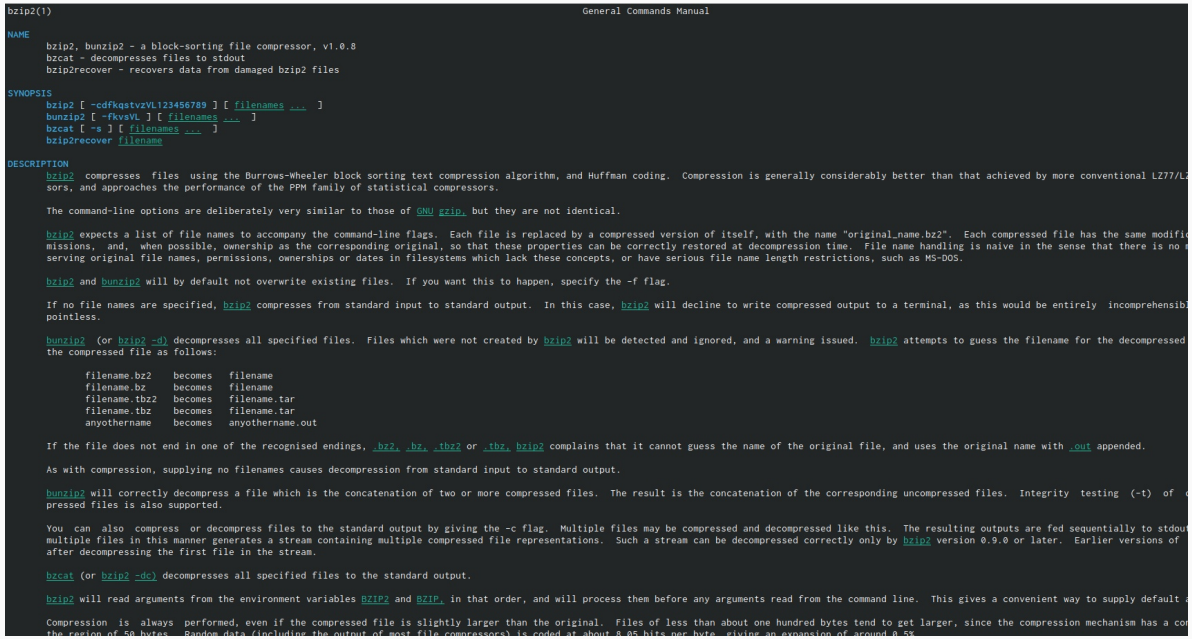


Figure 3: Man bzip2.

Команда man bzip2



bzip2(1) General Commands Manual

NAME

bzip2, bunzip2 - a block-sorting file compressor, v1.0.8
bzcat - decompresses files to stdout
bzip2recover - recovers data from damaged bzip2 files

SYNOPSIS

bzip2 [-cdfkqstvzVL123456789] [filenames ...]
bunzip2 [-fkvsVL] [filenames ...]
bzcat [-s] [filenames ...]
bzip2recover filename

DESCRIPTION

bzip2 compresses files using the Burrows-Wheeler block sorting text compression algorithm, and Huffman coding. Compression is generally considerably better than that achieved by more conventional LZ77/L
sors, and approaches the performance of the PPM family of statistical compressors.

The command-line options are deliberately very similar to those of GNU gzip, but they are not identical.

bzip2 expects a list of file names to accompany the command-line flags. Each file is replaced by a compressed version of itself, with the name "original_name.bz2". Each compressed file has the same modifi
missions, and, when possible, ownership as the corresponding original, so that these properties can be correctly restored at decompression time. File name handling is naive in the sense that there is no
serving original file names, permissions, ownerships or dates in filesystems which lack these concepts, or have serious file name length restrictions, such as MS-DOS.

bzip2 and bunzip2 will by default not overwrite existing files. If you want this to happen, specify the -f flag.

If no file names are specified, bzip2 compresses from standard input to standard output. In this case, bzip2 will decline to write compressed output to a terminal, as this would be entirely incomprehensib
pointless.

bunzip2 (or bzip2 -d) decompresses all specified files. Files which were not created by bzip2 will be detected and ignored, and a warning issued. bzip2 attempts to guess the filename for the decompressed
the compressed file as follows:

filename.bz2 becomes filename
filename.bz becomes filename
filename.tbz2 becomes filename.tar
filename.tbz becomes filename.tar
anyothername becomes anyothername.out

If the file does not end in one of the recognised endings, .bz2, .bz, .tbz2 or .tbz, bzip2 complains that it cannot guess the name of the original file, and uses the original name with .out appended.

As with compression, supplying no filenames causes decompression from standard input to standard output.

bunzip2 will correctly decompress a file which is the concatenation of two or more compressed files. The result is the concatenation of the corresponding uncompressed files. Integrity testing (-t) of
pressed files is also supported.

You can also compress or decompress files to the standard output by giving the -c flag. Multiple files may be compressed and decompressed like this. The resulting outputs are fed sequentially to stdout
multiple files in this manner generates a stream containing multiple compressed file representations. Such a stream can be decompressed correctly only by bzip2 version 0.9.0 or later. Earlier versions of
after decompressing the first file in the stream.

bzcat (or bzip2 -dc) decompresses all specified files to the standard output.

bzip2 will read arguments from the environment variables BZIP2 and BZIP, in that order, and will process them before any arguments read from the command line. This gives a convenient way to supply default

Compression is always performed, even if the compressed file is slightly larger than the original. Files of less than about one hundred bytes tend to get larger, since the compression mechanism has a co
the region of 50 bytes. Random data (including the output of most file compressors) is coded at about 8.05 bits per byte, giving an expansion of around 0.5%.

Figure 4: Информация о man bzip2.

Команда man tar

3. Команду man tar и получаем информацию:

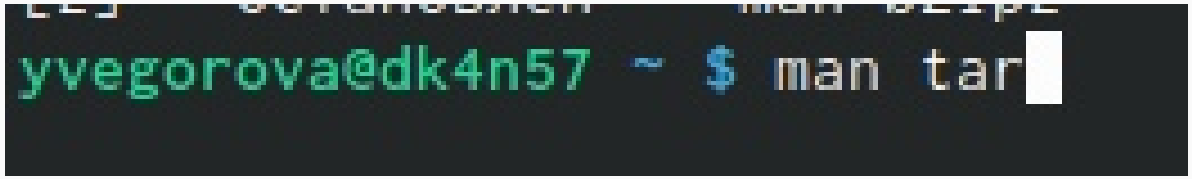
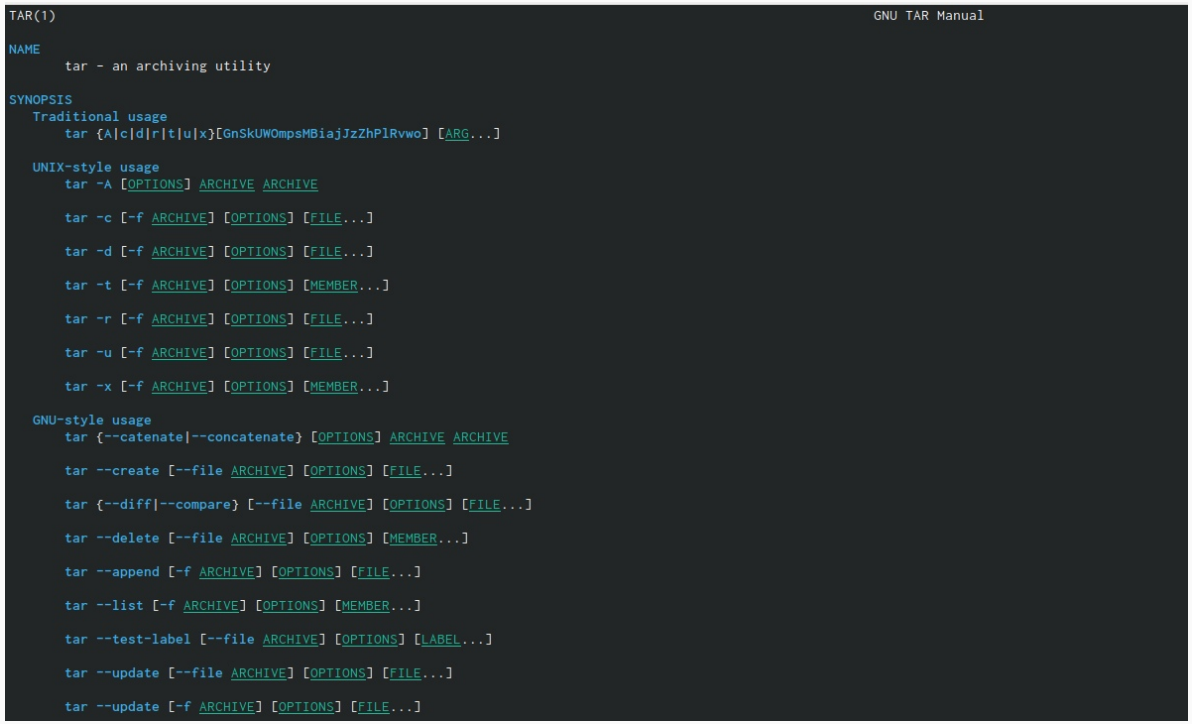


Figure 5: Man tar.

Команда man tar



```
TAR(1)                                                                  GNU TAR Manual

NAME
       tar - an archiving utility

SYNOPSIS
   Traditional usage
       tar {A|c|d|r|t|u|x}[GnSkUWOmpsMBiajJzZhPlRvwo] [ARG...]

   UNIX-style usage
       tar -A [OPTIONS] ARCHIVE ARCHIVE

       tar -c [-f ARCHIVE] [OPTIONS] [FILE...]

       tar -d [-f ARCHIVE] [OPTIONS] [FILE...]

       tar -t [-f ARCHIVE] [OPTIONS] [MEMBER...]

       tar -r [-f ARCHIVE] [OPTIONS] [FILE...]

       tar -u [-f ARCHIVE] [OPTIONS] [FILE...]

       tar -x [-f ARCHIVE] [OPTIONS] [MEMBER...]

   GNU-style usage
       tar {--catenate|--concatenate} [OPTIONS] ARCHIVE ARCHIVE

       tar --create [--file ARCHIVE] [OPTIONS] [FILE...]

       tar {--diff|--compare} [--file ARCHIVE] [OPTIONS] [FILE...]

       tar --delete [--file ARCHIVE] [OPTIONS] [MEMBER...]

       tar --append [-f ARCHIVE] [OPTIONS] [FILE...]

       tar --list [-f ARCHIVE] [OPTIONS] [MEMBER...]

       tar --test-label [--file ARCHIVE] [OPTIONS] [LABEL...]

       tar --update [--file ARCHIVE] [OPTIONS] [FILE...]

       tar --update [-f ARCHIVE] [OPTIONS] [FILE...]
```

Figure 6: Информация о man tar.

Работа с файлом в emacs

4. Создала файл backup.sh:

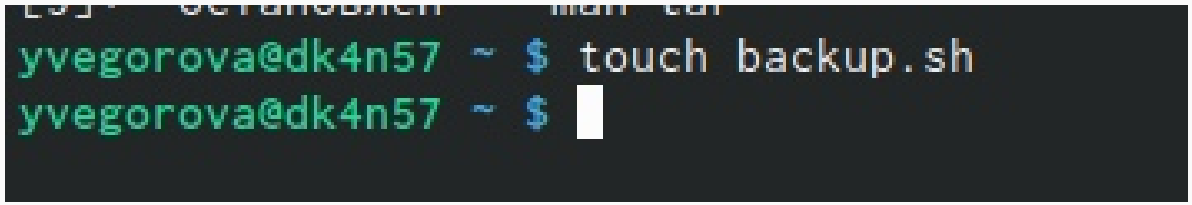


Figure 7: Создание файла.

Работа с файлом в emacs

5. Открыла emacs:

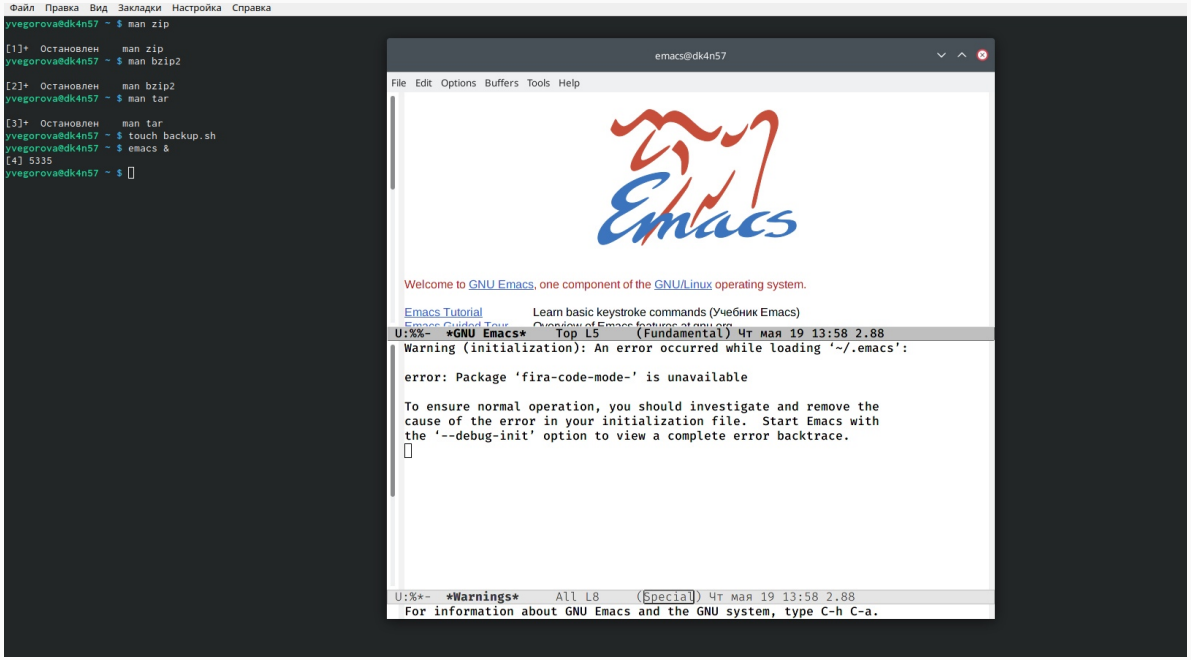
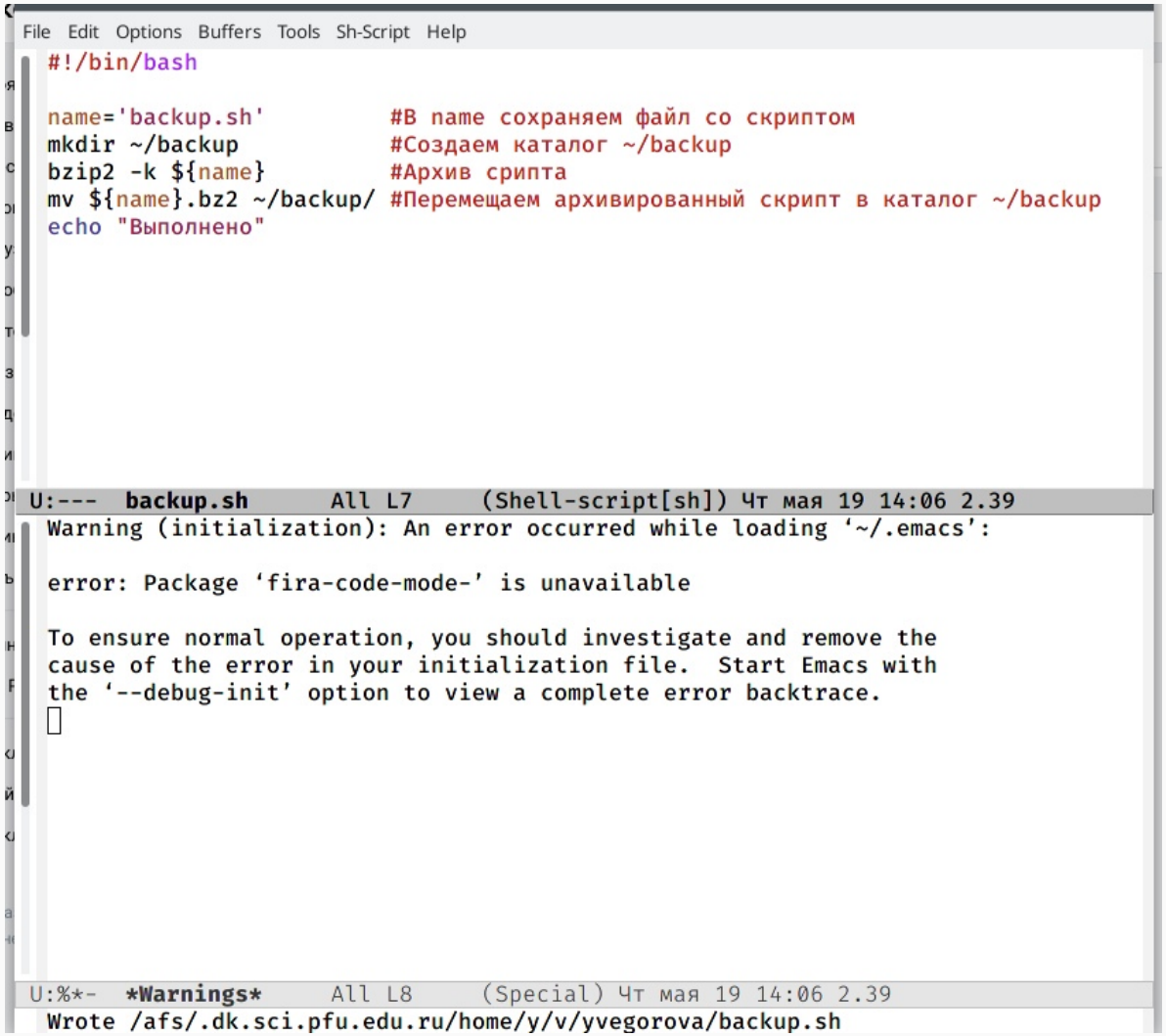


Figure 8: Открытие emacs.

Работа с файлом в emacs

6. По клавишам C-x C-f нашла созданный файл и ввела следующий текст:



Работа с файлом в emacs

7. Через клавиши C-x C-s сохранила наш файл и проверила в консоли через следующие команды:

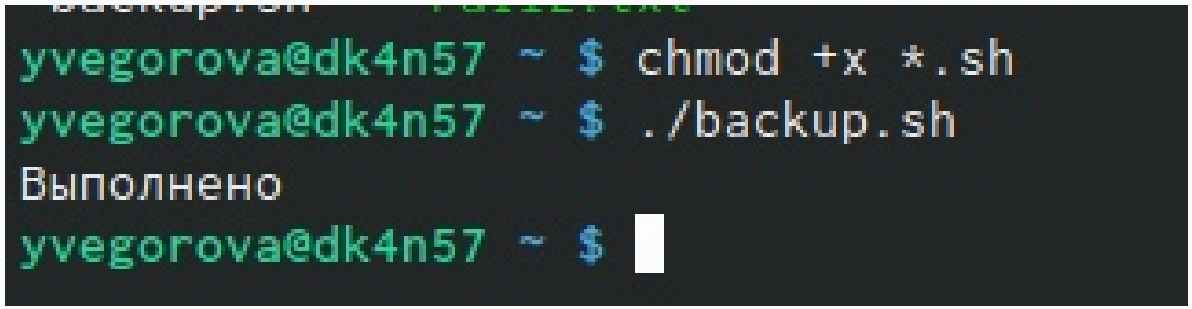


Figure 10: Проверка.

Работа с файлом в emacs

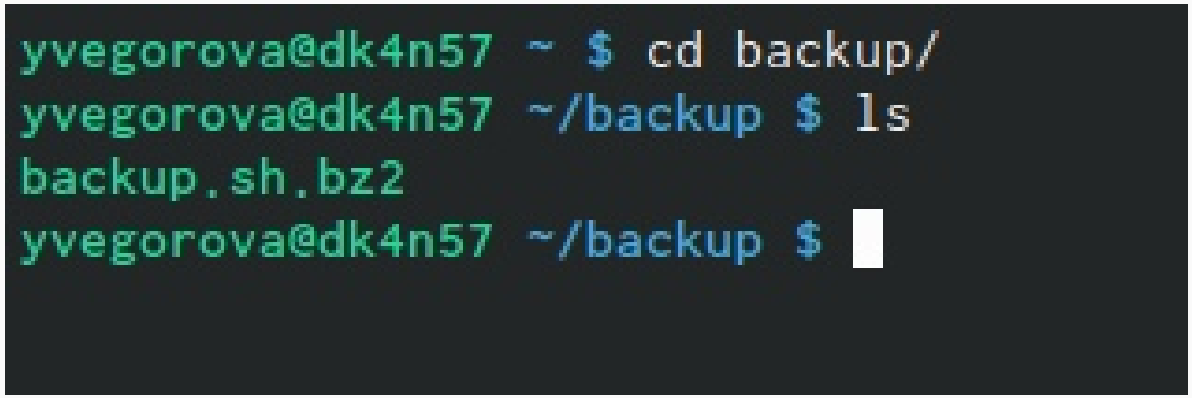


Figure 11: Проверка.

Работа с файлом в emacs

```
yvegorova@dk4n57 ~/backup $ bunzip2 -c backup.sh.bz2
#!/bin/bash

name='backup.sh'          #В name сохраняем файл со скриптом
mkdir ~/backup            #Создаем каталог ~/backup
bzip2 -k ${name}          #Архив срипта
mv ${name}.bz2 ~/backup/ #Перемещаем архивированный скрипт в каталог ~/backup
echo "Выполнено"
yvegorova@dk4n57 ~/backup $
```

Figure 12: Проверка.

Работа с файлом в emacs

8. Создала файл prog2.sh и перешла в emacs:

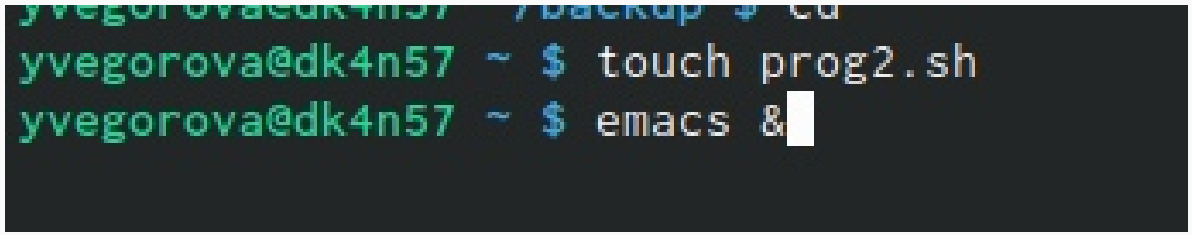


Figure 13: Создание файла и переход в emacs.

9. По клавишам C-x C-f нашла созданный файл и ввела следующий текст:

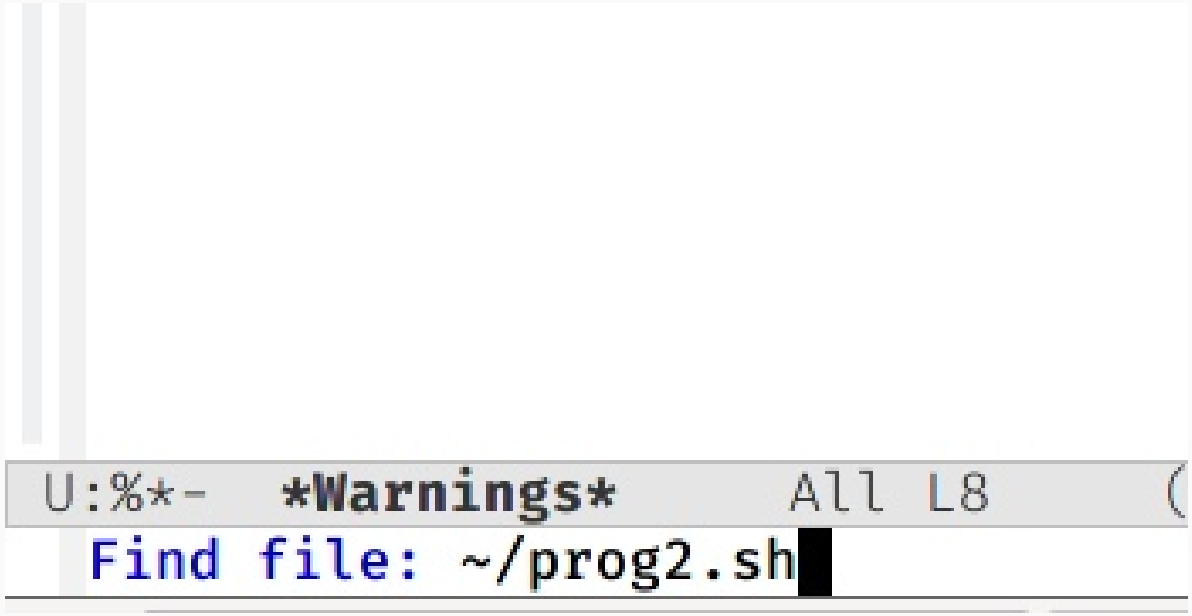


Figure 14: Поиск.

Работа с файлом в emacs

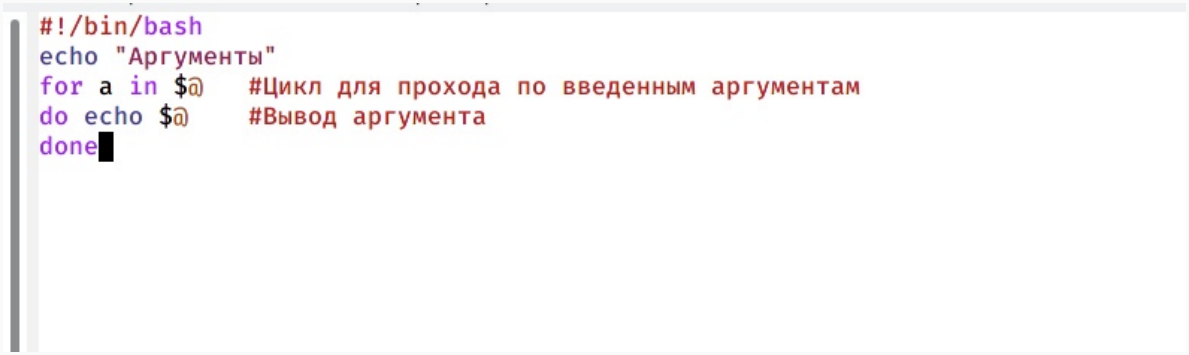


Figure 15: Заполнение.

10. Через клавиши C-x C-s сохранила наш файл и проверила в консоли через команду:

Работа с файлом в emacs

```
yvegorova@dk4n57 ~ $ ./prog2.sh 0 1 2 3 4
Аргументы
0 1 2 3 4
0 1 2 3 4
0 1 2 3 4
0 1 2 3 4
0 1 2 3 4
yvegorova@dk4n57 ~ $
```

Figure 16: Проверка.

Работа с файлом в emacs

11. Создала файл progls.sh и перешла в emacs:

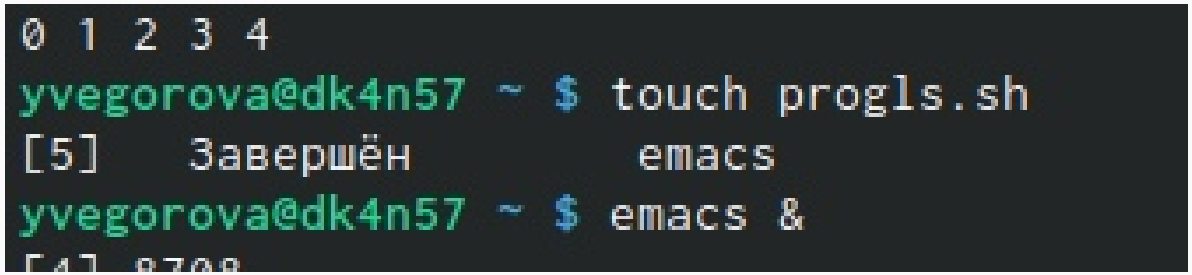


Figure 17: Создание файла и переход в emacs.

12. Ввела текст:

```
#!/bin/bash
a="$1"            #В переменную a сохраняем путь до заданного каталога
for i in ${a}/*   #Цикл, который проходит по всем каталогам и файлам
do
    echo "$i"

    if test -f $i
    then echo "Обычный файл"
    fi

    if test -d $i
    then echo "Каталог"
    fi

    if test -r $i
    then echo "Чтение разрешено"
    fi

    if test -w $i
    then echo "Запись разрешена"
    fi
```

Figure 18: Заполнение файла текстом.

Работа с файлом в emacs

13. Осуществила проверку:

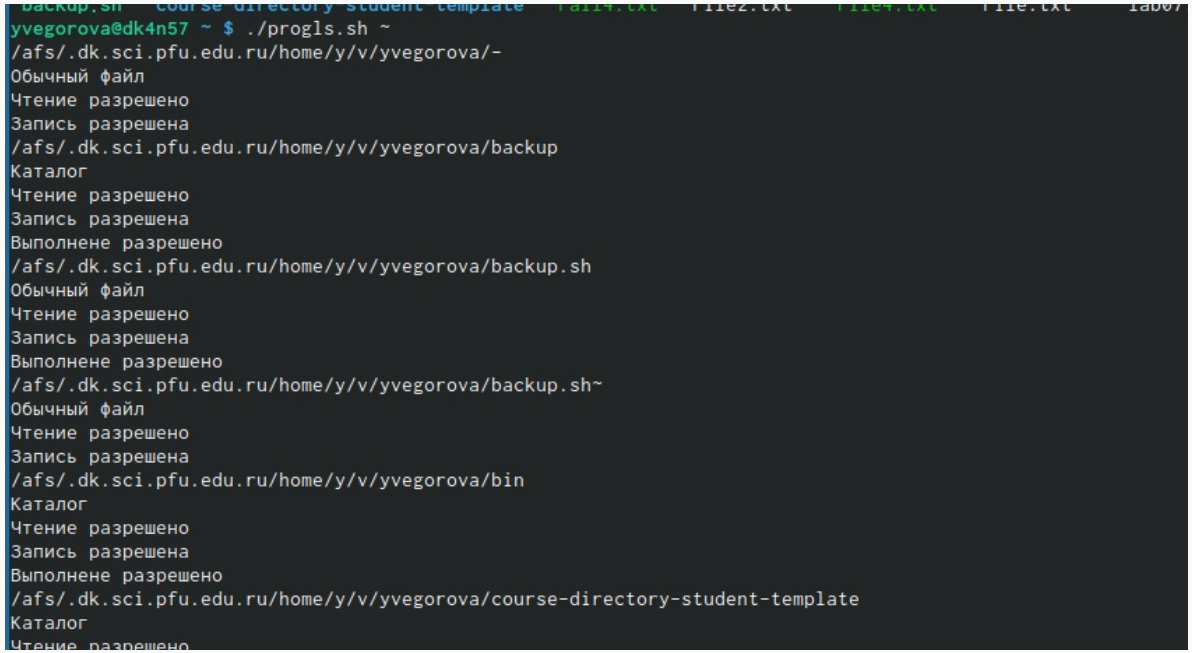


Figure 19: Проверка.

Работа с файлом в emacs

14. Снова создала файл под названием format.sh и перешла в emacs:

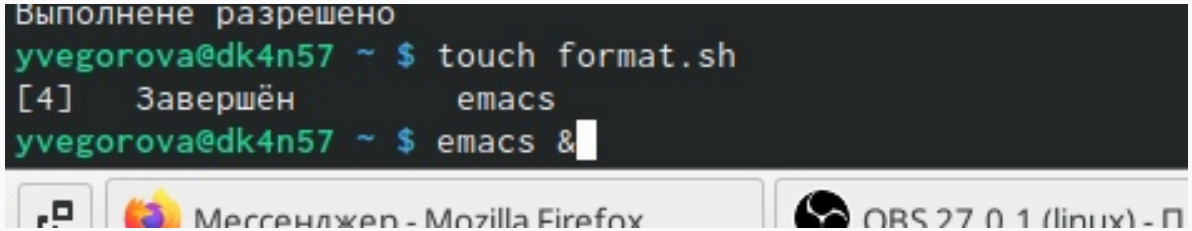


Figure 20: Создание файла и переход в emacs.

Работа с файлом в emacs

15. Нашла созданный файл и ввела текст:

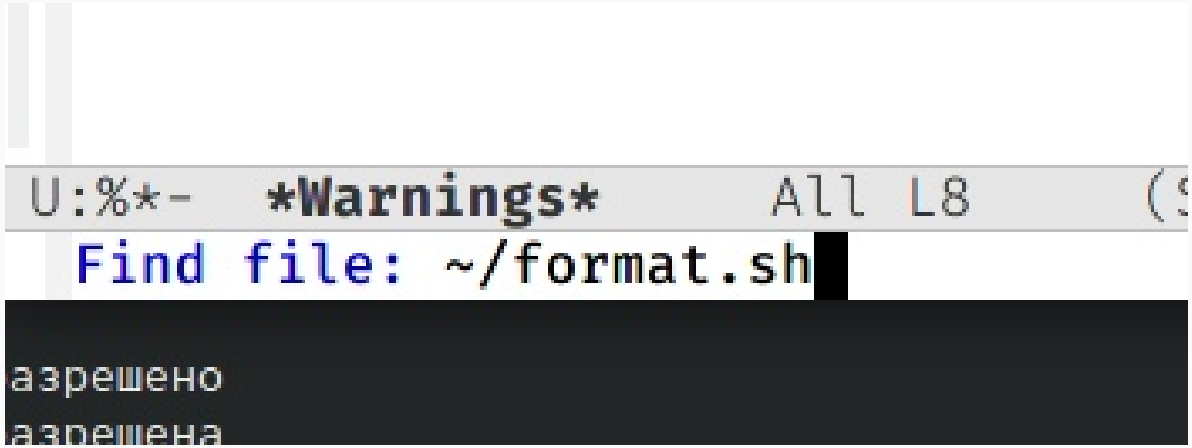


Figure 21: Поиск.

Работа с файлом в emacs

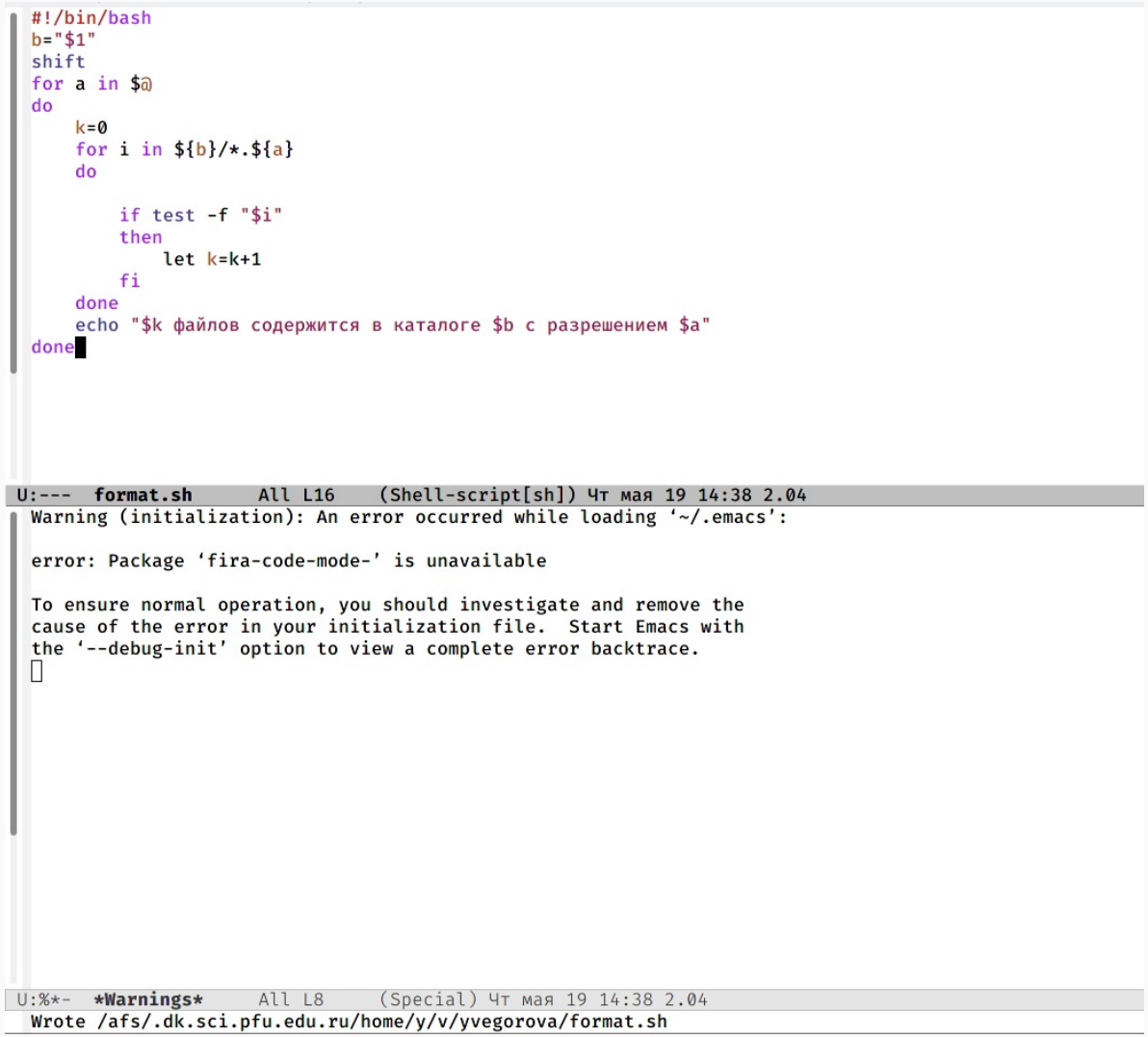


Figure 22: Текст.

Работа с файлом в emacs

16. Ввела следующую команду и выполнила проверку:

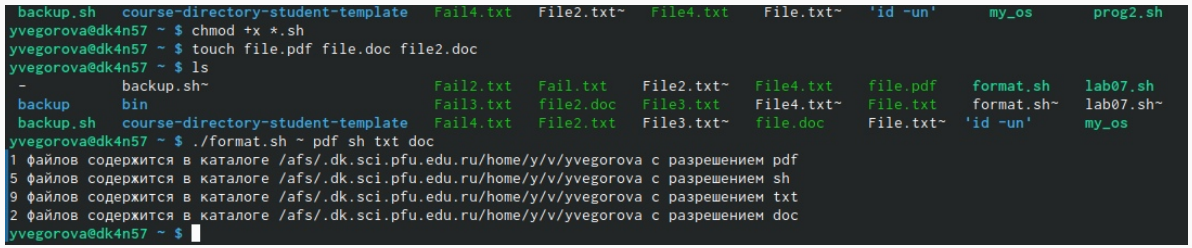


Figure 23: Команда и проверка.

Вывод

В ходе выполнения лабораторной работы я познакомилась с операционной системой Linux. Получила практические навыки работы с редактором vi, установленным по умолчанию практически во всех дистрибутивах