Лабораторная работа №9

1. Используя утилиты hexdump, strings вывести на экран содержимое файла из каталога /bin

   Для начала устанавливаем binutils

```
vboxuser@Linux:~$ sudo apt-get install binutils -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  binutils-common binutils-x86-64-linux-gnu libbinutils li
  libgprofng0 libsframe1
Suggested packages:
  binutils-doc gprofng-gui
```

   Strings - извлечения читаемых текстовых строк из бинарных файлов

   Bin- системный каталог

   Ping- проверяет доступность узла

   -n10 – минимальная длинна строки

```
vboxuser@Linux:~$ strings -n10 /bin/ping
/lib64/ld-linux-x86-64.so.2
_ITM_deregisterTMCloneTable
__gmon_start__
_ITM_registerTMCloneTable
__libc_start_main
__cxa_finalize
__cxa_atexit
__errno_location
__stack_chk_fail
__printf_chk
getaddrinfo
__memcpy_chk
freeaddrinfo
gai_strerror
setsockopt
__isoc23_strtol
__isoc23_strtoul
cap_get_proc
cap_get_flag
cap_set_flag
cap_set_proc
```

Далее hexdump – просмотр содержимое файлов в шестнадцатиричном формате

```
vboxuser@Linux:~$ hexdump -C /bin/ping
00000000  7f 45 4c 46 02 01 01 00  00 00 00 00 00 00 00 00  |.ELF............|
00000010  03 00 3e 00 01 00 00 00  90 5d 00 00 00 00 00 00  |..>......]......|
00000020  40 00 00 00 00 00 00 00  88 57 01 00 00 00 00 00  |@........W......|
00000030  00 00 00 00 40 00 38 00  0d 00 40 00 1d 00 1c 00  |....@.8...@.....|
00000040  06 00 00 00 04 00 00 00  40 00 00 00 00 00 00 00  |........@.......|
00000050  40 00 00 00 00 00 00 00  40 00 00 00 00 00 00 00  |@.......@.......|
00000060  d8 02 00 00 00 00 00 00  d8 02 00 00 00 00 00 00  |................|
00000070  08 00 00 00 00 00 00 00  03 00 00 00 04 00 00 00  |................|
00000080  18 03 00 00 00 00 00 00  18 03 00 00 00 00 00 00  |................|
00000090  18 03 00 00 00 00 00 00  1c 00 00 00 00 00 00 00  |................|
000000a0  1c 00 00 00 00 00 00 00  01 00 00 00 00 00 00 00  |................|
000000b0  01 00 00 00 04 00 00 00  00 00 00 00 00 00 00 00  |................|
000000c0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
000000d0  90 20 00 00 00 00 00 00  90 20 00 00 00 00 00 00  |. ....... ......|
000000e0  00 10 00 00 00 00 00 00  01 00 00 00 05 00 00 00  |................|
000000f0  00 30 00 00 00 00 00 00  00 30 00 00 00 00 00 00  |.0.......0......|
00000100  00 30 00 00 00 00 00 00  71 ac 00 00 00 00 00 00  |.0......q.......|
00000110  71 ac 00 00 00 00 00 00  00 10 00 00 00 00 00 00  |q...............|
00000120  01 00 00 00 04 00 00 00  00 e0 00 00 00 00 00 00  |................|
00000130  00 e0 00 00 00 00 00 00  00 e0 00 00 00 00 00 00  |................|
00000140  48 32 00 00 00 00 00 00  48 32 00 00 00 00 00 00  |H2......H2......|
00000150  00 10 00 00 00 00 00 00  01 00 00 00 06 00 00 00  |................|
```

2. Подсчитываем общее кол-во файлов в одном из каталогов.

```
vboxuser@Linux:~$ ls /etc | wc -l
234
```

Ls /etc – выводит список элементов каталога

Wc –l – считает кол-во строк

3. Найти кол-во процессов, выполняющихся в системе на данный момент

```
vboxuser@Linux:~$ ps -e | wc -l
244
```

Wc – выводит все процессы в системе

-l – считает строки

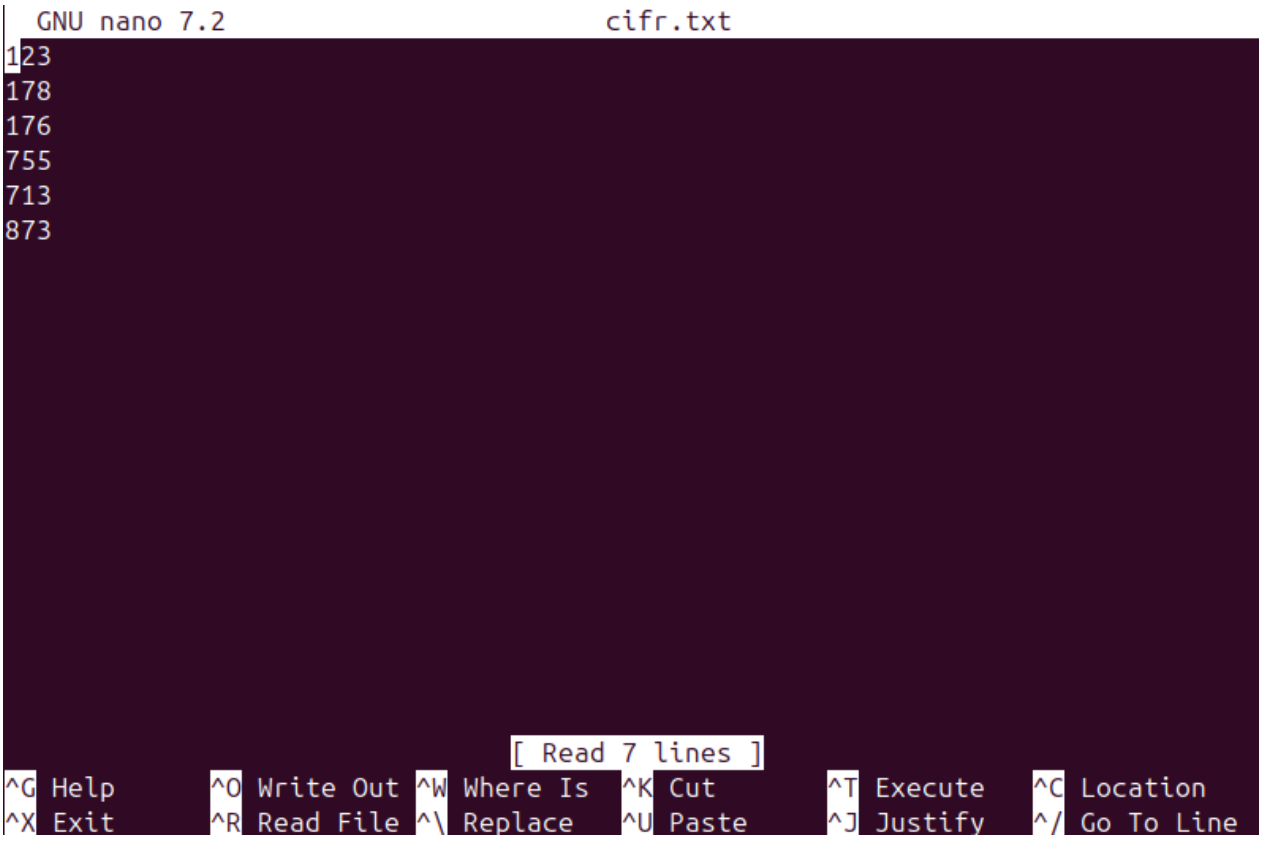
4. Выводим список процессов, в имени которых есть manager и нет grep

```
vboxuser@Linux:~$ ps -aux | grep manager | grep -v grep
root         125  0.0  0.0      0     0 ?        I<   07:59   0:00 [kworker/R-ch
arger_manager]
```

- aux – a – процессы всех юзеров, u – подробный формат, x – фоновые процессы
Grep manager – фильтрует оставляя только строки с подстроками
Grep –v grep исключает из результата строки grep – чтобы не уйти в бесконечность

5. Создаем текстовый файл с содержимым

123
178
176
755
713
873



[…] - является фильтром для поиска 135 идущих после 7

```
vboxuser@Linux:~$ grep '7[1|3|5]' cifr.txt
755
713
873
```

6. Создаем текстовый файл содержащий набор строк

starfish

starless

samscripter

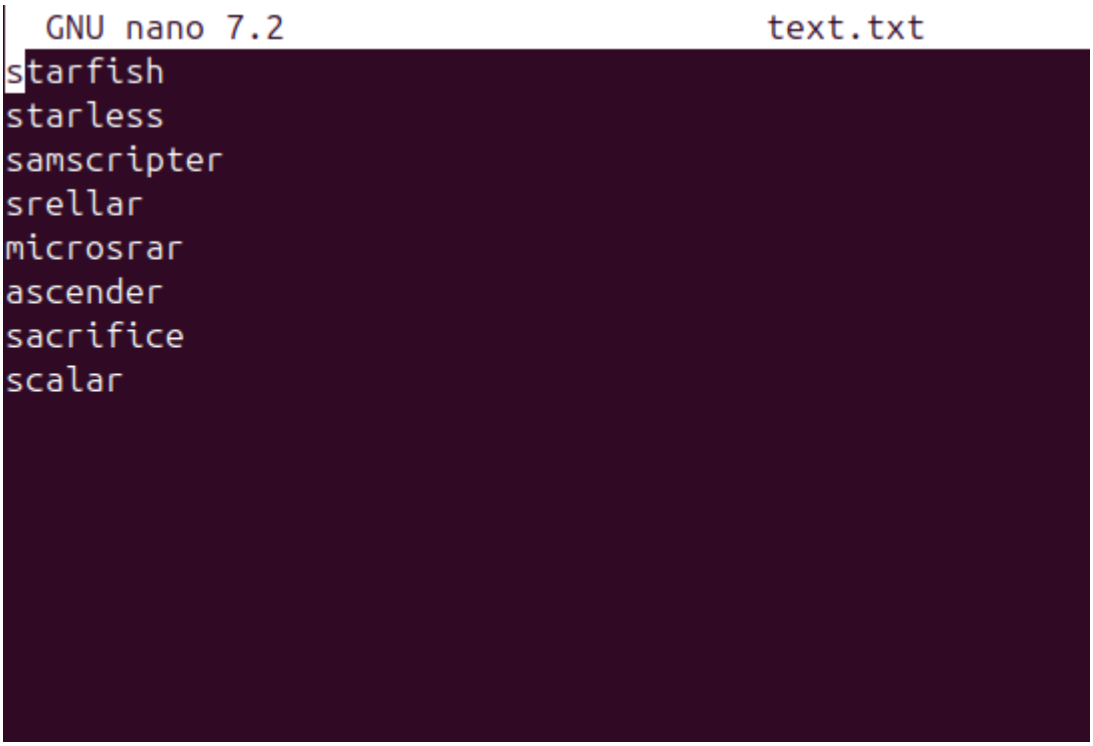
stellar

microsrar

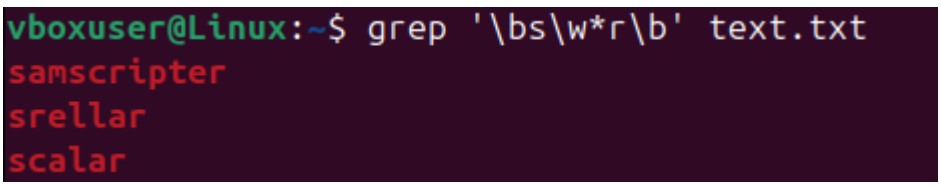
ascender

sacrifice

scalar



Ищем слова с буквы s и конец на r по аналогии с предыдущем заданием



7. Создать текстовый файл, содержащий простейшие адреса электронной почты вида username@website.com

   С помощью утилиты grep найти строки, содержащие правильные простейшие адреса. Проверить возможность использования более сложного регулярного выражения для распознавания адресов, содержащих другие допустимые символы.

```
  GNU nano 7.2                                email.
sokol.game2016@yandex.ru
sukhopar.03@gamil.com
uandexplus.com
SPBGD.ru
ptiven@mail.ru
shalom@pravolav
```

В скобках все возможные символы в правильных почтах - -E -o '\b[A-Za-z0-9._%=?^+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,6}\b'

```
vboxuser@Linux:~$ grep -E -o '\b[A-Za-z0-9._%=?^+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,
6}\b' email.txt
sokol.game2016@yandex.ru
sukhopar.03@gamil.com
ptiven@mail.ru
```

8. Создаем файл, с айпи адресами, продемонстрировать работу утилиты tr

   127.0.0.1

   255.255.255.255

   12.34.56

   123.256.0.0

   1.23.099.255

   0.79.378.111

```
  GNU nano 7.2
127.0.0.1
255.255.255.255
12.34.56
123.256.0.0
1.23.099.255
0.79.378.111
```

При помощи man находим строки с четырехбайтовыми ip

В скобках все допустимые числовые значения

'(25[0-5]2[0-4][0-9]|[01]?[0-9][0-9]?)\.(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.(25[0-5]|2[0 4][0-9]|[01]?[0-9][0-9]?)\.(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)'

```
vboxuser@Linux:~$ grep -E -o '(25[0-5]2[0-4][0-9]|[01]?[0-9][0-9]?)\.(25[0-5]|2[0-4][0-9]|[01]
04][0-9]|[01]?[0-9][0-9]?)\.(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)' ip.txt
127.0.0.1
55.255.255.255
1.23.099.255
```

9. .Создаем текстовый файл, содержащий корректные и некорректные номера телефонов ведомственной АТС объемом 399 номеров, номера с 000 до 399 – корректные, 0, 400, 900 некорректные.

Создаем цикл для создания номеров

```
#!/bin/bash
echo -e "0\n" > phone.txt
for i in {000..399}; do
  echo "$i" >> phone.txt   (000-399)
done
echo -e "400\n900" >> phone.txt
```

В скобках условия для поиска '(39[0-9|2[0-4][0-9]|[01]?[0-9][0-9]?)'

```
vboxuser@Linux:~$ grep -E -o '(39[0-9|2[0-4][0-9]|[01]?[0-9][0-9]?)' phone.txt
0
000
39
9
000
39
9
40
0
90
0
```