ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №9 ОБРАБОТКА СТРОК (РАБОТА С ТЕКСТОВЫМИ ДАННЫМИ)

Цель работы - практическое знакомство со способами эффективной обработки текста при помощи интерфейса командной строки и набора стандартных утилит.

МЕТОДИКА ВЫПОЛНЕНИЯ

1. Используя утилиты hexdump и strings, вывести на экран содержимое одного из перечисленных ниже файлов из каталога /bin. Позиция файла для распечатки определяется номером бригады. Имена файлов для выполнения задания 1: tar, sort, sed, ping, vi, unlink, uname, touch, sleep, sty.

Командой **hexdump -C /bin/sleep** выводится содержимое файла **sleep** в шестнадцатеричном виде. В результате мы видим слева номер строки, в середине байты, справа — те же байты как символы. Непечатаемые символы заменяются точками.

Готовый результат:

```
vboxuser@tusabusa:~/Desktop$ hexdump -C /bin/sleep
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  80 2b 00 00 00 00 00 00  |..>......+......|
00000020  40 00 00 00 00 00 00 00  48 82 00 00 00 00 00 00  |@.......H.......|
00000030  00 00 00 00 40 00 38 00  0d 00 40 00 1f 00 1e 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  c0 12 00 00 00 00 00 00  c0 12 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 20 00 00 00 00 00 00  00 20 00 00 00 00 00 00  |. ....... ......|
00000100  00 20 00 00 00 00 00 00  b1 36 00 00 00 00 00 00  |. .......6......|
00000110  b1 36 00 00 00 00 00 00  00 10 00 00 00 00 00 00  |.6..............|
00000120  01 00 00 00 04 00 00 00  00 60 00 00 00 00 00 00  |.........`......|
00000130  00 60 00 00 00 00 00 00  00 60 00 00 00 00 00 00  |.`.......`......|
00000140  00 0e 00 00 00 00 00 00  00 0e 00 00 00 00 00 00  |................|
00000150  00 10 00 00 00 00 00 00  01 00 00 00 06 00 00 00  |................|
00000160  d0 7b 00 00 00 00 00 00  d0 7b 00 00 00 00 00 00  |.{.......{......|
```

Далее командой **strings -n 10 /bin/sort** показывают только те фрагменты файла, которые могут быть интерпретированы как текстовые строки. Ключ **-n 10** означает, что выводятся строки длиной не меньше 10 символов.

Готовый результат:

```
vboxuser@tusabusa:~/Desktop$ strings -n10 /bin/sleep
/lib64/ld-linux-x86-64.so.2
__libc_start_main
__cxa_finalize
__cxa_atexit
__fprintf_chk
__printf_chk
fputs_unlocked
nl_langinfo
__freading
__errno_location
__fpending
getopt_long
fputc_unlocked
__stack_chk_fail
__ctype_get_mb_cur_max
__ctype_b_loc
__memset_chk
program_invocation_name
__progname_full
bindtextdomain
program_invocation_short_name
__progname
```

2. Подсчитать общее количество файлов (каталогов) в одном из перечисленных ниже каталогов. Каталог для подсчета количества определяется номером бригады. Имена каталогов для выполнения задания 2: /bin, /etc, /lib, /proc, /usr, /var, /dev, /sbin, /sys, /root

С помощью команды **ls -1 /sbin** выводим список файлов по одному в каждой строке. Затем с помощью символа | отправляем этот вывод в команду **wc -l**, которая просто подсчитывает количество строк. В результате получаем число — это и есть количество файлов в каталоге **sbin.**

Готовый результат:

```
vboxuser@tusabusa:~/Desktop$ ls -1 /sbin | wc -l
543
```

3. Найти общее количество процессов, выполняющихся в системе в данный момент.

С помощью команды **ps aux** выводим полный список всех работающих процессов. Через конвейер | этот список передаём в команду **wc -l**, которая считает количество строк. В результате получаем число – общее количество запущенных процессов.

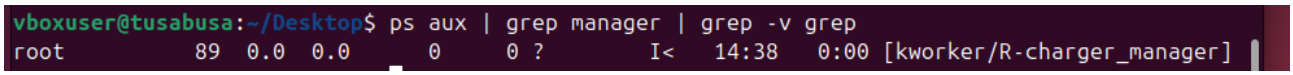
Готовый результат:

```
vboxuser@tusabusa:~/Desktop$ ps aux | wc -l
187
```

4. Вывести список выполняющихся процессов, в именах которых присутствует слово manager и отсутствует слово grep.

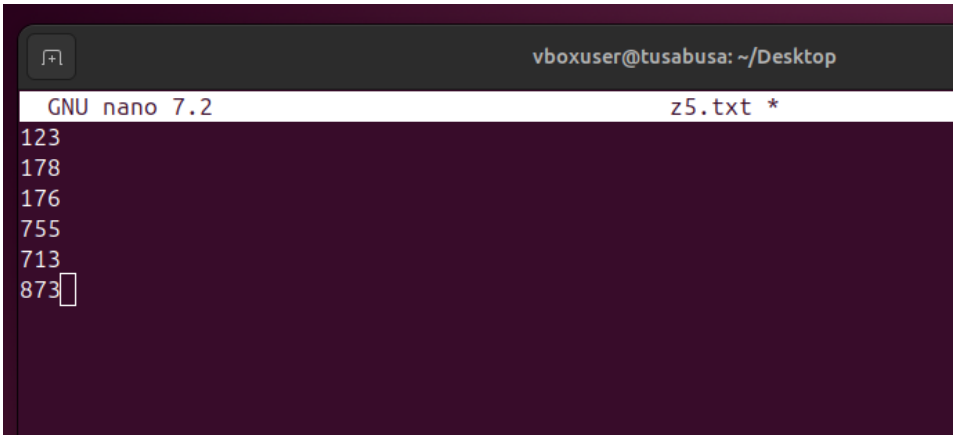
С помощью команды **ps aux** выводим все процессы. Далее через | передаём этот список в **grep manager**, чтобы оставить только те строки, где встречается слово **manager**. Затем ещё раз используем конвейер, где **grep -v grep** убирает из результата строку с самой командой **grep**. В итоге на экране остаются только нужные процессы с **manager** в имени.

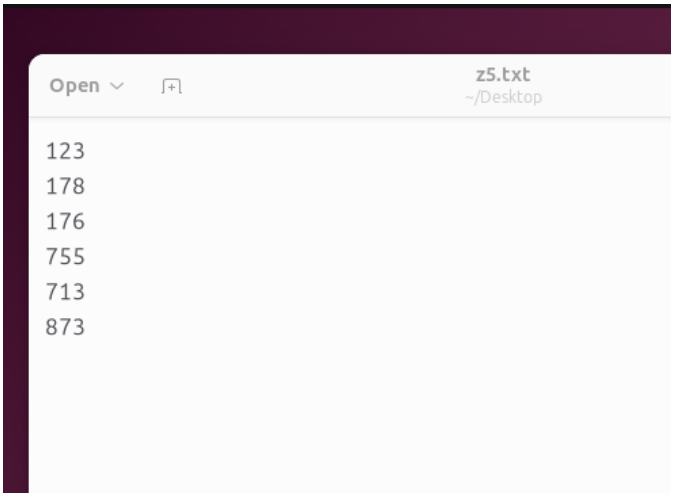
Готовый результат:



5. Создать текстовый файл, содержащий набор строк вида: 123, 178, 176, 755, 713, 873. С помощью утилиты grep найти строки, в которых есть цифра 7, после которой находится одна из цифр — 1, 3 или 5.

Чтобы создать текстовый файл открываем текстовый редактор командой **nano z5.txt** и вписываем туда необходимые числа. После ввода сохраняем файл и выходим из **nano** сочетанием клавиш **Ctrl+X**.





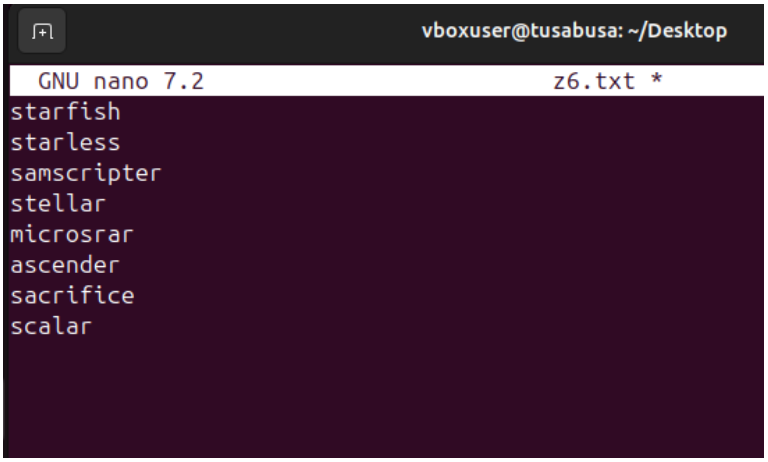
Затем используем команду **grep "7[1|3|5]" z5.txt.** Она просматривает файл и выводит только те строки, где есть цифра 7, а сразу после неё стоит 1, 3 или 5.

Готовый результат:

```
vboxuser@tusabusa:~/Desktop$ grep "7[1|3|5]" z5.txt
755
713
873
```

6. Создать текстовый файл, содержащий набор строк вида: starfish, starless, samscripter, stellar, microsrar, ascender, sacrifice, scalar. С помощью утилиты grep найти строки, начинающиеся на букву s и заканчивающиеся на букву r.

С помощью редактора **nano** создаём текстовый файл **z6.txt** и вписываем в него нужные слова. Сохраняем файл и выходим из редактора (Ctrl+X).



Далее используем команду **grep "\b[Ss]\w*[Rr]\b" z6.txt.** Эта команда ищет в файле строки, где слово начинается на букву s или S и заканчивается на r или R.
\b означает границу слова (начало/конец), **[Ss]** — первую букву s в любом регистре, **\w*** — любое количество любых символов внутри слова, а **[Rr]** — последнюю букву r в любом регистре. В результате на экран выводятся только подходящие слова из файла.

Готовый результат:

```
vboxuser@tusabusa:~/Desktop$ grep "\b[Ss]\w*[Rr]\b" z6.txt
samscripter
stellar
scalar
```

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.

Сначала создаём файл с разными строками, в том числе с e-mail-адресами, и смотрим его содержимое командой **cat z7.txt.**

```
vboxuser@tusabusa:~/Desktop$ cat z7.txt
abcd@gmail.com
fasol_em@yandex.ru
12@.sous
suzuki2000@nail.com
```

С помощью команды **grep** из файла вытаскиваем определенные e-mail-адреса. Ключ **-E** включает расширенные регулярные выражения, **-o** выводит только совпавшие части строки. Шаблон после **grep** описывает адрес. В начале стоит имя пользователя с допустимыми символами, затем знак @, потом доменное имя, точка и домен верхнего уровня длиной от 2 до 6 букв. В итоге на экран попадают только правильно записанные адреса из файла.

Готовый результат:

```
vboxuser@tusabusa:~/Desktop$ grep -E -o '\b[A-Za-z0-9._%?^+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,6}\b' z7.txt
abcd@gmail.com
fasol_em@yandex.ru
suzuki2000@nail.com
```

8. На произвольном примере продемонстрировать работу утилиты tr. Создать текстовый файл, содержащий допустимые и недопустимые IP-адреса, например 127.0.0.1; 255.255.255.255; 12.34.56; 123.256.0.0; 1.23.099.255; 0.79.378.111. С помощью утилиты grep и руководства man найти строки, содержащие допустимые четырехбайтовые IP адреса.

Создаём файл **z8.txt** со строками, где есть и правильные, и неправильные IP-адреса, и смотрим его через команду **cat**. Затем командой **grep -E '^(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]?)){3}$' z8.txt** оставляем только те строки, где все четыре числа в адресе лежат в диапазоне **0–255**, то есть это корректные IPv4-адреса. Внутренний блок **(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)** описывает одно число **от 0 до 255** и повторяется 4 раза, разделённый точками.

```
vboxuser@tusabusa:~/Desktop$ nano z8.txt
vboxuser@tusabusa:~/Desktop$ cat z8.txt
127.0.0.1
255.255.255.245
12.34.551
123.678.0.0.
1.23.456.789
0.87.654.333
```

Готовый результат:

```
vboxuser@tusabusa:~/Desktop$ grep -E '(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]?)' z8.txt

127.0.0.1
255.255.255.245
```

Ниже показано, как с помощью команды **tr** и ключа **-s** убрать повторяющиеся символы. Команда **echo "bicccycle" | tr -s "c"** сжимает подряд идущие **c**, и на выходе получается корректное слово.

Готовый результат:

```
vboxuser@tusabusa:~/Desktop$ echo "bicccycle" |tr -s "c"
bicycle
```

9. Создать текстовый файл, содержащий корректные и некорректные номера телефонов ведомственной АТС объемом 399 номеров, номера с 000 до 399 – корректные, 0, 400, 900 –некорректные. С помощью утилиты grep и руководства man найти строки, содержащие допустимые номера телефонов.

Создаем текстовый файл **z9.txt** с корректными и некорректными номерами ведомственной АТС, где правильные номера лежат в диапазоне **от 000 до 399**, а 0, 400, 900 и т.п. считаются неверными.

```
vboxuser@tusabusa:~/Desktop$ cat z9.txt
000
399
400
900
555
1
35
```

Затем с помощью команды **grep -E** и регулярного выражения, в котором задаются ограничения для каждой из трёх цифр, из файла выбираются только допустимые номера из диапазона 000–399 и они выводятся на экран.

Готовый результат:

```
vboxuser@tusabusa:~/Desktop$ grep -E "(0[0-9][0-9]|[1-2][0-9][0-9]|3[0-9][0-9])" z9.txt
000
399
```

Лукьянчикова Дарья, группа 2-МВ-4